SESSION

ACCREDITATION + ASSESSMENT STRATEGIES + ACADEMIC REVIEWS

Chair(s)

TBA

Assessing Student Learning in Computer Science – A Case Study

K.B. Lakshmanan[†], Sandeep R. Mitra[†], and T.M. Rao[†]
[†]Department of Computer Science
The College at Brockport, State University of New York
Brockport, NY 14420, USA

Abstract – *Student learning outcomes are statements that describe what students are expected to know and be able to do by the time of graduation. In order to assess the extent to which an outcome is met, it is necessary to define an outcome in terms of measurable performance indicators. Rubrics allow collection of relevant data and their consistent interpretation. Using just one outcome as an example, this paper presents the approach used by the College at Brockport, State University of New York to define appropriate performance indicators, construct a curriculum map, develop holistic rubrics, collect data, evaluate, and use the findings for continuous program improvement.*

Keywords: ABET, accreditation, assessment, outcomes, performance indicators, rubrics

1 Introduction

Accreditation is a review process that educational institutions undergo periodically for improvement of academic quality and for public accountability [5]. ABET is a recognized accreditor of college and university programs in applied science, computing, engineering, and technology [1]. The Computing Accreditation Commission (CAC) of ABET accredits computing programs, in particular Computer Science, Information Systems, and Information Technology. The Engineering Accreditation Commission (EAC) of ABET accredits Computer Engineering and Software Engineering programs.

The College at Brockport, State University of New York is a co-educational, state assisted, comprehensive liberal arts college in Western New York, offering bachelor's, master's, and post-master's certificate programs. The Department of Computer Science is located in the School of Science and Mathematics. The Advanced Computing Track of the Computer Science Major was first accredited by CSAB [4] in 1994. With the integration of CSAB as a member society of ABET, the program is now accredited by the Computing Accreditation Commission (CAC) of ABET. The undergraduate program is a small one, graduating about 10 students per year. Moreover, since other unaccredited computing programs exist, students pursuing this rigorous accredited program are highly motivated academic achievers. The department does not offer a graduate program in computer science.

The Department of Computer Science at The College at Brockport, State University of New York has a long tradition of continuous program improvement through assessment of student learning outcomes. The department undertook a major revision of the outcomes and the assessment process following the release of the outcomes based assessment criteria by the Computing Accreditation Commission of ABET in 2006. In preparation for this revision, almost all faculty members of the department attended ABET-sponsored workshops on assessment. In this paper, we focus on just one outcome as an example, and discuss the approach we used to develop an assessment methodology that has been used successfully for ABET accreditation since then.

2 Outcomes

ABET requires each computing program seeking accreditation to develop a clear set of student learning outcomes, collect assessment data, determine the extent to which the outcomes are attained, and use the results of the evaluation to improve the program. In ABET terminology the student learning outcomes were previously known as program outcomes. But they are now known simply as student outcomes. The ABET web site has extensive material related to assessment [2].

Criterion 3 (Student Outcomes) of the ABET Criteria for Accrediting Computing Programs lists a set of characteristics any computing program must enable a student to attain. The list is augmented by additional characteristics under specific program criteria for computer science, information systems and information technology. Specifically, for computer science, the criteria list eleven characteristics. An institution may adopt this list (a)-(k) itself as the set of student learning outcomes for its computer science program. However, at Brockport, we chose to develop a smaller set of eight outcomes that incorporated the essence of these eleven characteristics:

A. An ability to apply fundamental principles of computing and mathematics as appropriate to the discipline of computer science.

B. An ability to analyze a problem and model it as a computing system using appropriate methodologies, and to identify the computing requirements necessary to meet the desired needs.

C. An ability to design, implement and test a computing system, and to evaluate and compare the efficiencies of alternative solutions.

D. An ability to use current techniques, skills, and tools appropriate for immediate employment in computing technology development fields.

E. An ability to function effectively on teams to accomplish a common goal.

F. An ability to communicate effectively, both orally and in writing, using accepted standards of the profession.

G. An ability to analyze the social and human context of computing as it impacts individuals, organizations, and society, including ethical, legal, security, and global policy issues.

H. An ability to work and learn independently and an appreciation of the importance of continuing education and professional growth over the course of a lifetime.

The eight outcomes relate to fundamentals of the discipline, problem analysis and logical design, physical design and implementation, technical skills, team work, communication skills, ethics and other issues, and continuing education. Our outcomes are well-documented and widely disseminated through print literature and on the web to a variety of stakeholders such as potential and current students, alumni, and employers. ABET criteria require that there be a documented and effective process for the periodic review and revision of these outcomes to ensure that they are consistent with the published program educational objectives, which are broad statements that describe what graduates are expected to attain within a few years of graduation.

3 Performance Indicators

ABET criteria provides considerable leeway as to how the assessment of student learning outcomes is carried out. The criteria merely states that “effective assessment uses relevant direct, indirect, quantitative and qualitative measures as appropriate to the outcome being measured.” However, with growing maturity in the assessment process, programs are expected to place greater emphasis on *direct* assessment of student learning, which is based on student demonstrated work, rather than *indirect* assessment, which is based on student, alumni, or employer surveys. In this paper, we will just focus on Outcome A, as an example, and describe our assessment methodology. We will also focus only on the direct assessment of the outcome through evaluation of student work. As can be seen, Outcome A is a minor restatement of the student characteristic (a) in the published ABET criteria. Clearly, this outcome is a fairly broad statement encompassing fundamentals of computer science and mathematics appropriate to computer science. Thus, the assessment of the same requires breaking it down to a number of simple measurable aspects that allow one to determine the extent to which the outcome is met. In ABET literature these measurable aspects are known as performance indicators or performance criteria. Performance indicators are written with action verbs such as define, demonstrate, discriminate, evaluate, and interpret. Performance indicators also spell out specific subject contents [3]. In our case, we chose the following six performance indicators to define the Outcome A:

A1. Demonstrate an understanding of basic data structures and algorithms

A2. Demonstrate an understanding of a high-level object-oriented programming language and software design

A3. Demonstrate an understanding of number systems and digital logic

A4. Demonstrate an understanding of computer organization and architecture

A5. Demonstrate an understanding of analysis of algorithms

A6. Demonstrate an understanding of models of computation

The above list was chosen after careful evaluation of several performance indicators proposed by faculty members. Note that we have chosen two performance indicators to assess each of the software, hardware and theory aspects of computer science. Also the six performance indicators together cover the five foundational areas of computer science mentioned in the criteria: algorithms, data structures, software design, concepts of programming languages, and computer organization and architecture. Thus, collectively, the student performances measured by these indicators provide a reasonable basis for stating claims on the extent to which the Outcome A is attained. Similar sets of performance indicators have been developed for all other outcomes, but will not be presented or discussed in this paper.

4 Curriculum Map

Once the performance indicators have been identified, the next step for us was to determine for each performance indicator a set of courses or educational practices in the program through which the relevant skills, knowledge, and behaviors are acquired. If carried out carefully, this step in itself can help an institution identify some weaknesses in the program. For each performance indicator, the direct assessment data may be gathered from course material and student performances in classes through a variety of sources such as assignments, laboratory and written examinations, portfolios, programming projects, and oral examinations [6]. It is important to note that ABET expects *summative* assessment data, because the student learning outcomes specify what students are expected know and be able to do *at the time of graduation*. In other words, it is not necessary to gather data regarding each performance indicator at various stages in the curriculum. An institution may choose to develop *formative* assessment data for better control of their program, but ABET criteria do not require this. Further, some institutions may require a senior capstone project course. Assessment data for several performance indicators may be gathered from student demonstrated work in this course alone. But, in order to even out the workload on the faculty, it may be desirable to spread the assessment data collection over several courses. At Brockport, we collect assessment data for performance indicators in upper-level classes, as close to the time of graduation as possible. Table 1 presents performance indicators developed for the Outcome A, where the associated knowledge and skills are developed, where they are assessed, and how the assessment data is collected.

The following is a partial list of required courses in the computer science curriculum that appear in Table 1:
CSC 203: Fundamentals of Computer Science I (CS1)
CSC 205: Fundamentals of Computer Science II (CS2)
CSC 303: Digital Logic and Computer Design
CSC 311: Comp. Organization and Software Interface
CSC 401: Programming Languages
CSC 406: Algorithms and Data Structures
CSC 411: Computer Architecture
CSC 483: Theory of Computation

The complete curriculum and detailed descriptions of these courses may be obtained from the departmental web site at http://www.brockport.edu/cs/. Notice that if the knowledge or skill relevant to a particular performance indicator is developed over a sequence of courses, we collect assessment data in the last course in the sequence, that is, as close to the time of graduation as possible.

5 Rubrics

For further discussion let us focus on the Performance Indicator A2, which requires a student to demonstrate an understanding of a high-level object-oriented programming language and software design. Our goal in assessment is to categorize a student based on his or her performance relative to this performance indicator in one of four levels – beginning, developing, competent, and accomplished. But, exactly what aspects of programing languages and software design should be considered and exactly what data should be collected from course material? Further, how can we assure that different instructors at different times will characterize student performances consistently? To answer these questions, we have developed a *holistic rubric* associated with this performance indicator. Table 2 provides our rubric. Similar rubrics have been developed for all performance indicators associated with all student learning outcomes.

6 Data Collection

Until recently, we assessed all outcomes on a two-year cycle. We have recently moved to a three-year cycle. In any case, all instructors are informed of the performance indicators for which they need to collect data at the beginning of each academic year. Thus, an instructor for CSC 205, who is required to collect assessment data regarding the performance indicator A2 will be able to use the rubric for guidance and embed appropriate questions in course examinations or embed specific components in course projects and laboratory exercises that map to the knowledge and skills articulated in the rubric. The instructor will then be able to extract performance data and use them to categorize students in one of four levels indicated before.

At the end of the data collection period, which could be one semester or one academic year depending on the course, the instructor responsible for data collection is expected to provide the assessment coordinator the following:

- head counts of students in each of the four levels of performance
- mapping of course assignments, examinations, projects, etc., to rubric topics
- specific course assignments, projects, and examinations used in the analysis, and
- a reflective statement that could explain the distribution of students in the various performance levels

Table 3 is an example of head counts and the mapping of course material to rubric topics reported by an instructor.

7 Evaluation and Reporting

At the end of an academic year, the assessment coordinator of the department will tabulate all the head counts provided by various instructors and present them to the assessment committee for evaluation. Table 4 presents the compilation of student performances in regard to Outcome A during the 2006-08 assessment cycle. Chart 1 provides a visual presentation of the same data. The assessment committee is responsible for evaluating the data and arriving at suitable recommendations. We had set an a priori target level for performance as follows: an outcome is considered achieved if the percentage of the students rated competent or accomplished is 70% or higher. However, the assessment committee will still consider each and every one of the performance indicators individually and recommend actions for improvement even if the outcome is considered achieved. In this context, the reflective statements submitted by instructors along with assessment data plays a significant role.

8 Program Improvement

Table 5 provides an example of a typical recommendation for program improvement based on the assessment of Outcome A. Observe that the Outcome A is considered achieved as the percentage of students rated competent or accomplished is above 70%. However, close inspection of the individual performance indicators show that for both A3 and A4, the percentages of students rated competent or accomplished is below 70%, more so in the case of A4. Both performance indicators relate to hardware aspects of computing covered in the sequence of courses CSC 303, 311, and 411. The recommendation in this case took into consideration the recent changes in the curriculum and suggested only minor fine-tuning of course contents. In subsequent cycles of assessment, particular attention will be paid to recommendations of previous cycles.

9 Concluding Remarks

Using just one outcome as an example in this paper, we have presented the approach used by the College at Brockport, State University of New York to define appropriate performance indicators, construct a curriculum map, develop holistic rubrics, collect data, evaluate, and use the findings for continuous program improvement. Some assessment data have been modified to protect the privacy of instructors and to make this brief presentation more illustrative. The assessment methodology has been used for more than two cycles. With that experience, we have following observations to make.

- Student learning outcomes: We chose not to use the CAC list of student characteristics (a)-(k) as our outcomes, under the assumption that having fewer outcomes would lead to less work on the faculty. But, we now believe that there is no particular advantage to have developed our own set of student learning outcomes, since assessment data collection workload is really dependent on the total number of performance indicators we have.
- Performance indicators: When we first designed our assessment scheme, we tried to keep the performance indicators *collectively exhaustive* of all aspects of the outcome assessed. As a result, in some cases, we ended up with as many as ten performance indicators, leading to considerable workload for our faculty. We now believe that about six performance indicators per outcome are appropriate. Having fewer than six performance indicators may not provide sufficient information for program improvement.
- Holistic rubrics: Our rubrics show varying levels of detail as several faculty members participated in the development. We now believe that we need to bring some uniformity as these rubrics play a significant role in data collection and reducing faculty bias in scoring.

10 Acknowledgment

The authors acknowledge the contributions of other faculty members in the Department of Computer Science, The College at Brockport of the State University of New York in the development of the student learning outcomes, the performance indicators, and the holistic rubrics.

11 References

[1] ABET, Inc., http://www.abet.org/

[2] ABET Assessment Planning, http://www.abet.org/assessment-planning/

[3] M. Besterfield-Sacre, et. al., “Defining the Outcomes: A Framework for EC 2000”, *IEEE Trans. On Education*, Vol. 43, No. 2, May 2000, pp. 100-110.

[4] CSAB, Inc., http://www.csab.org/

[5] Council for Higher Education Accreditation (CHEA) At a Glance, http://www.chea.org/

[6] K.E. Sanders and R. McCartney, “Collected Wisdom: Assessment Tools for Computer Science Programs”, *Computer Science Education*, Vol. 14, No. 3, 2004, pp. 183-203.

Table 1. Performance indicators, curriculum map, and assessment method

Outcome A. An ability to apply fundamental principles of computing and mathematics as appropriate to the discipline of computer science.

Performance Indicators	**Curriculum Map (Where Developed)**	**Where Assessed**	**Assessment Method**
A1. Demonstrate an understanding of basic data structures and algorithms	CSC 203, 205, 406	CSC 406	Selected questions extracted from course examinations
A2. Demonstrate an understanding of a high-level object-oriented programming language and software design	CSC 203, 205	CSC 205	Selected questions extracted from course examinations; selected components of course projects
A3. Demonstrate an understanding of number systems and digital logic	CSC 303	CSC 303	Selected questions extracted from course examinations; selected components of course projects
A4. Demonstrate an understanding of computer organization and architecture	CSC 303, 311, 411	CSC 411	Selected questions extracted from course examinations
A5. Demonstrate an understanding of analysis of algorithms	CSC 205, 406	CSC 406	Selected questions extracted from course examinations
A6. Demonstrate an understanding of models of computation	CSC 401, 483	CSC 483	Selected questions extracted from course examinations

Table 2. Rubric for Performance Indicator A2

Performance Indicator A2. Demonstrate an understanding of a high-level object-oriented programming language and software design.

Beginning (Fails to meet)	**Developing (Close to Meeting)**	**Competent (Meets, Satisfactory)**	**Accomplished (Exceeds, Exemplary)**
Cannot determine appropriate data representation or design algorithms. Cannot implement solutions using simple control structures in a high-level language. Cannot analyze problems, identify classes, assign responsibilities, or design solutions. Cannot create analysis/design artifacts such as use cases and class diagrams.	Can determine a data representation and design an algorithm for simple problems. Can implement solutions using basic control structures (if, switch, and looping) to solve simple problems in a high-level language. Can analyze problems, identify classes, assign responsibilities, and design solutions for simple cases. Can create some analysis/ design artifacts such as use cases and class diagrams.	Can determine an appropriate data representation and design correct algorithms. Can implement solutions using control structures (if, switch, looping, and recursion) to solve reasonable sized problems in a high-level language. Can analyze problems, identify classes, assign responsibilities, and design solutions for reasonably complex cases. Can create analysis/design artifacts such as use cases and class diagrams for reasonably complex cases.	Can determine the most appropriate data representation and design efficient algorithms. Can implement solutions using control structures (if, switch, looping, and recursion) to solve large problems in a high-level language. Can analyze problems, identify classes, assign responsibilities, and design solutions for fairly complex cases. Can create analysis/design artifacts such as use cases and class diagrams for fairly complex cases.

Table 3. Reporting of head counts and mapping of course material to rubric topics*

Performance Indicator A2. Demonstrate an understanding of a high-level object-oriented programming language and software design.

Beginning **(Fails to meet)**	**Developing** **(Close to Meeting)**	**Competent** **(Meets, Satisfactory)**	**Accomplished** **(Exceeds, Exemplary)**
0	2	4	6

Rubric Topic	**Course Topic**	**Assignment, Exam, and Projects**
Data representation and design of algorithms	Collection classes: stacks, queues, linked lists, binary trees, etc.	Exam II: Question 2; Project 2
Implementation in a high-level language	Java implementation, manipulation of data structures, wrapper classes, recursion	Exam I: Question 3, 4; Laboratory Assignments1, 2
Problem analysis, identification of classes and assignment of responsibilities	Abstract classes, interfaces	Final examination Questions 7 and 8; Project 3
Creation of analysis/design artifacts	Use cases/CRC cards	Exam II: Question 6; Laboratory Assignment 4; Project 3

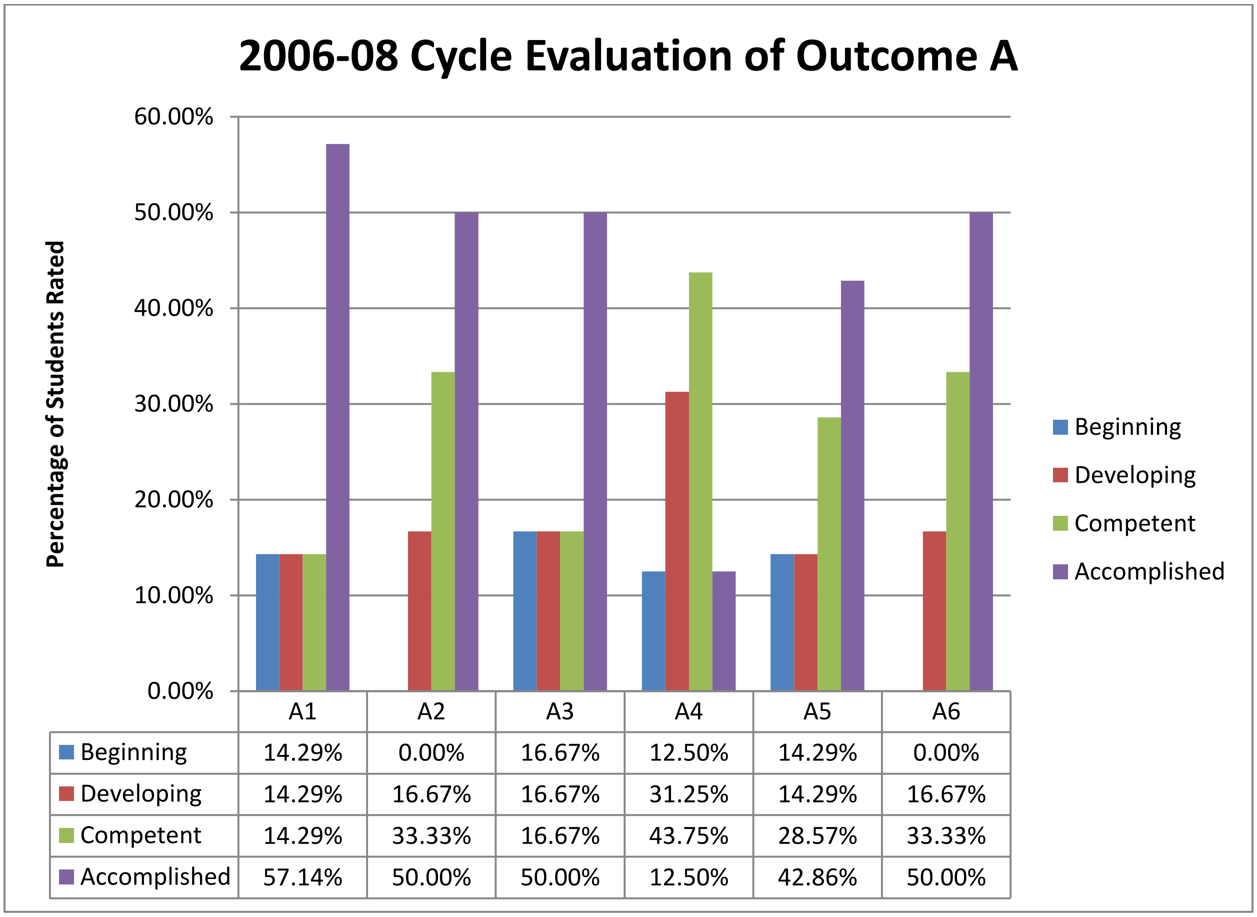
Table 4. Compilation of assessment data for Outcome A during 2006-08 cycle*

Outcome A. An ability to apply fundamental principles of computing and mathematics as appropriate to the discipline of computer science.

Performance Indicators	**Where/When Assessed**	**Head Count of Students Rated**				
		Beginning	**Developing**	**Competent**	**Accomplished**	**Total**
A1. Demonstrate an understanding of basic data structures and algorithms	CSC 406 Spring 2007	2	2	2	8	**14**
A2. Demonstrate an understanding of a high-level object-oriented programming language and software design	CSC 205 2006-07	0	2	4	6	**12**
A3. Demonstrate an understanding of number systems and digital logic	CSC 303 2006-07	2	2	2	6	**12**
A4. Demonstrate an understanding of computer organization and architecture	CSC 411 Spring 2007	2	5	7	2	**16**
A5. Demonstrate an understanding of analysis of algorithms	CSC 406 Spring 2007	2	2	4	6	**14**
A6. Demonstrate an understanding of models of computation	CSC 483 Spring 2007	0	1	2	3	**6**
Total		**8**	**14**	**21**	**31**	**74**

*Data presented here are for illustrative purposes only and are not actual

Chart 1. Visual presentation of assessment data for Outcome A



	A1	A2	A3	A4	A5	A6
🟦 Beginning	14.29%	0.00%	16.67%	12.50%	14.29%	0.00%
🟥 Developing	14.29%	16.67%	16.67%	31.25%	14.29%	16.67%
🟩 Competent	14.29%	33.33%	16.67%	43.75%	28.57%	33.33%
🟪 Accomplished	57.14%	50.00%	50.00%	12.50%	42.86%	50.00%

Table 5. Recommendations for program improvement

Assessment Cycle 2006-08

Outcome A. An ability to apply fundamental principles of computing and mathematics as appropriate to the discipline of computer science.

Percentage of students rated competent or accomplished: 52/74= 70.27%

Status of the outcome: Achieved. (The percentage of students rated competent or accomplished is above 70%)

Recommendation(s): Performances under indicators A3 and A4 are of concern. The percentages of students rated competent or accomplished are below 70% for both. We have revised the contents of CSC 311: Computer Organization and Software Interface effective Fall 2006. We have moved from Intel to MIPS architecture. We also require programming in “C” language in this course now. Hence, no immediate course correction is recommended. The instructors teaching hardware-related courses (CSC 303, 311 and 411) should continue to monitor the situation, with small fine-tuning of contents, if found necessary.

Time frame for implementation: Starting from the next offerings of CSC 303, 311, and 411.

Using A Learning Management System to Facilitate Program Accreditation

Harry N. Keeling, Ph.D.
Howard University
Department of Systems and Computer Science
College of Engineering, Architecture and Computer Sciences
2300 6th Street NW, Washington, DC 20059
(202) 806-4830

ABSTRACT

In this paper we introduce AssessTrack, a web-based learning management system (LMS) that ***Assess****es and* ***Track****s key elements of engineering education. It is designed to (1) facilitate student learning, (2) ease the rigors of course management and (3) address the daunting task of collecting assessment data for engineering program accreditation. AssessTrack provides an individualized, tailored study guide generated by AssessTrack's intelligent tutor. For educators, the system provides a paperless environment where they can easily post all the key components of the instructional process (course material, lectures, assessments, tutorials, surveys and grades). AssessTrack gives engineering program administrators and accreditation agencies (in our case, the Accreditation Board for Engineering and Technology, ABET) the ability to track the attainment of program objectives and course educational outcomes, both formatively and summatively. AssessTrack is accessible from personal computers, tablets and even smart phones.*

Keywords

LMS (learning management system), assessment, accreditation, computer science education, ITS (intelligent tutoring systems)

1. INTRODUCTION

Currently there are hundreds of computer-based learning and performance tools [1]. For many years, university instructors have employed selected subsets of this software to evaluate and analyze the educational process [2, 3]. However, most of this research has been directed at measuring the acceptance and usability of these technological advancements [4]. Some studies have focused on pedagogical considerations, while others analyze user perceptions to provide insights into how to design and utilize these educational tools [5]. Though there has been some study of the relationship of online courses to learning outcomes [6], there has been little or no research conducted into using web-based learning systems to support the program accreditation process.

Program accreditation is the assurance that a college or university program meets the quality standards established by the profession for which it prepares its students. For example, an accredited computer science program must meet the quality standards set by the computing profession. So, to receive ABET accreditation a program must regularly use processes for assessing and evaluating the extent to which both program and student outcomes are being attained. [7]

To prepare for the program accreditation review process, accreditation agencies, like ABET, require that an educational program conduct a self-study. The self-study documents how well the program is meeting the established accreditation criteria in multiple areas, such as their students, curriculum, faculty, administration, facilities, and institutional support. Assess track is designed to print many of the required self-study reports as well as allowing the external program evaluators access to these reports online. We have not found any other software that provides similar features.

2. TOOL DEVELOPMENT

In 2008, we began researching computer-based assessment tools to assist in our self-study in preparation for an upcoming ABET visit. We found no tools that could meet our needs. So, we began developing prototypes and eventually ended up with a tool that we called the Course Management System (CMS). CMS was used for 2 years and evolved over time. We then decided to discontinue CMS and develop a new, more robust system that could handle the needs of all students, any instructor, and every educational unit of the university.

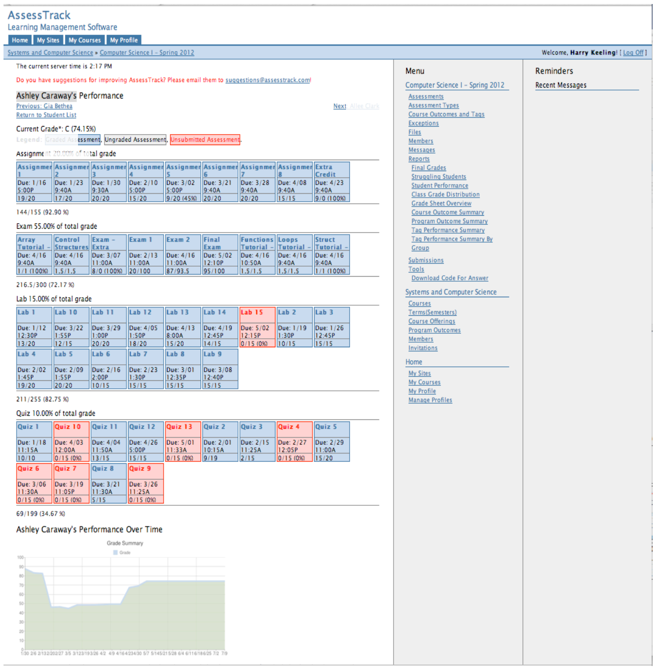
The features of AssessTrack were added incrementally employing a methodology best described as "evolutionary prototyping". The prototyping effort began with an application that would allow Computer Science I students to take a quiz on lab computers. We wanted to start taking steps towards running a paperless course and to find an easier way to generate the reports about student performance required by ABET. The next prototypes were individual desktop applications. The students would answer the questions and the system would save them to the computer's hard drive. We would later retrieve the files and use the data to score the answers. After several iterations, the current version of AssessTrack is a web-based .NET application that uses the Model View Controller (MVC) architecture. It uses a central SQL database to store the many tables needed to support a variety of functions. We have recently deployed a cloud version of AssessTrack to address throughput and capacity issues.

3. KEY FUNCTIONS

Within AssessTrack there are many noteworthy functions that serve students, instructors, teaching assistants, administrators and accreditation agencies. Students can perform all of their course work within the AssessTrack LMS with the exception of compiling their computer programs. The system has an embedded C++ compiler, but its use is reserved for online grading of programming assignments, quizzes and exams by teaching assistants and instructors. AssessTrack allows for use of collusion detection software (JPlag) that uncovers students who may have copied others students' submissions.

3.1 Student Online Notebooks

The key function for a student is access to their individual online notebook. AssessTrack gives students the ability to see all the work they have submitted, their detailed grades, and their overall grade in the course at any point in time.



AssessTrack
Learning Management Software

Systems and Computer Science » Computer Science I – Spring 2012

Welcome, **Harry Keeling**! [Log Off]

The current server time is 2:17 PM

Do you have suggestions for improving AssessTrack? Please email them to suggestions@assesstrack.com!

Ashley Caraway's Performance

Previous: Gia Bethea

Next: Allee Clark

Return to Student List

Current Grade*: C (74.15%)

Legend: Graded Assessment, Ungraded Assessment, Unsubmitted Assessment.

Assignment 20.00% of total grade

Assignment 1	Assignment 2	Assignment 3	Assignment 4	Assignment 5	Assignment 6	Assignment 7	Assignment 8	Extra Credit
Due: 1/16 5:00P	Due: 1/23 9:40A	Due: 1/30 9:30A	Due: 2/10 5:00P	Due: 3/02 5:00P	Due: 3/21 9:40A	Due: 3/28 9:40A	Due: 4/08 9:40A	Due: 4/23 9:40A
19/20	17/20	20/20	15/20	9/20 (45%)	20/20	20/20	15/15	9/0 (100%)

144/155 (92.90 %)

Exam 55.00% of total grade

Array Tutorial –	Control Structures	Exam – Extra	Exam 1	Exam 2	Final Exam	Functions Tutorial –	Loops Tutorial –	Struct Tutorial –
Due: 4/16 9:40A	Due: 4/16 9:40A	Due: 3/07 11:00A	Due: 2/13 11:00A	Due: 4/16 11:00A	Due: 5/02 12:10P	Due: 4/16 10:50A	Due: 4/16 9:40A	Due: 4/16 9:40A
1/1 (100%)	1.5/1.5	8/0 (100%)	20/100	87/93.5	95/100	1.5/1.5	1.5/1.5	1/1 (100%)

216.5/300 (72.17 %)

Lab 15.00% of total grade

Lab 1	Lab 10	Lab 11	Lab 12	Lab 13	Lab 14	Lab 15	Lab 2	Lab 3
Due: 1/12 12:30P	Due: 3/22 1:55P	Due: 3/29 1:00P	Due: 4/05 1:50P	Due: 4/13 8:00A	Due: 4/19 12:45P	Due: 5/02 12:15P	Due: 1/19 1:30P	Due: 1/26 12:45P
13/20	12/15	20/20	18/20	15/20	14/15	0/15 (0%)	10/15	15/15
Lab 4	Lab 5	Lab 6	Lab 7	Lab 8	Lab 9			
Due: 2/02 1:45P	Due: 2/09 1:55P	Due: 2/16 2:00P	Due: 2/23 1:30P	Due: 3/01 12:35P	Due: 3/08 12:40P			
19/20	20/20	10/15	15/15	15/15	15/15			

211/255 (82.75 %)

Quiz 10.00% of total grade

Quiz 1	Quiz 10	Quiz 11	Quiz 12	Quiz 13	Quiz 2	Quiz 3	Quiz 4	Quiz 5
Due: 1/18 11:15A	Due: 4/03 12:00A	Due: 4/04 11:50A	Due: 4/26 5:00P	Due: 5/01 11:33A	Due: 2/01 10:15A	Due: 2/15 11:25A	Due: 2/27 12:05P	Due: 2/29 11:00A
10/10	0/15 (0%)	13/15	15/15	0/15 (0%)	9/19	2/15	0/15 (0%)	15/20
Quiz 6	Quiz 7	Quiz 8	Quiz 9					
Due: 3/06 11:30A	Due: 3/19 11:05P	Due: 3/21 11:30A	Due: 3/26 11:25A					
0/15 (0%)	0/15 (0%)	5/15	0/15 (0%)					

69/199 (34.67 %)

Ashley Caraway's Performance Over Time

Menu

Computer Science I – Spring 2012
- Assessments
- Assessment Types
- Course Outcomes and Tags
- Exceptions
- Files
- Members
- Messages
- Reports
  - Final Grades
  - Struggling Students
  - Student Performance
  - Class Grade Distribution
  - Grade Sheet Overview
  - Course Outcome Summary
  - Program Outcome Summary
  - Tag Performance Summary
  - Tag Performance Summary By Group
- Submissions
- Tools
  - Download Code For Answer

Systems and Computer Science
- Courses
- Terms(Semesters)
- Course Offerings
- Program Outcomes
- Members
- Invitations

Home
- My Sites
- My Courses
- My Profile
- Manage Profiles

Reminders

Recent Messages

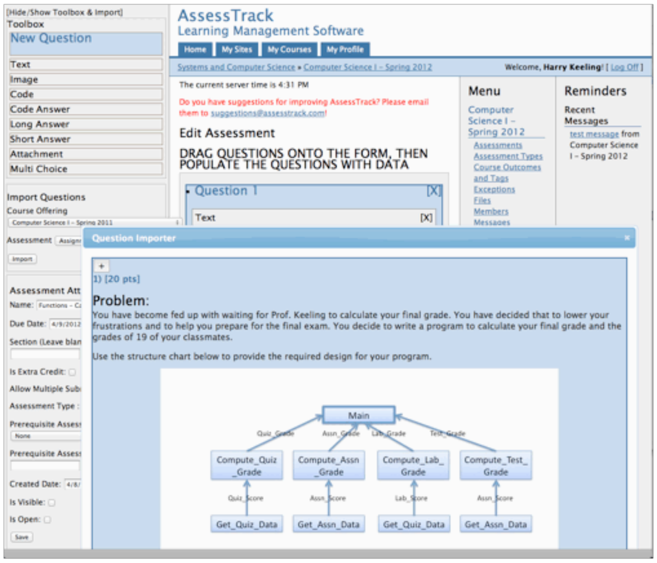
Figure 1: Student "My Grades" Screen

Figure 1 shows a student's "My Grades" screen that only they can see after inputting their username and password.

This screen shows students all of their assessments, their grades for each assessment, an aggregate measure of performance for each assessment type and their overall grade for the course based on the work that has been submitted. A student can click on any submitted assessment (blue square) and drill down to see the content on the assessment, their answers, the correct answers, and their grades on each assessment component (question). This screen can be thought of as an electronic notebook containing all of their work during the semester. Any work that has not been submitted and is overdue will appear in red and any submitted work that has not been graded will show up in gray. The graph at the bottom of Figure 1 shows the student's overall performance over time. This student started off good, had a stump for about 2 and a half weeks before mid-term, but brought their grades up afterwards to finally earn a "C" in the course. Students have commented that this screen helps them determine their direction and helps to motivate them to keep making progress. Students no longer have to ask the instructor "How am I doing in your class?" The answer is always available to them, even on their smart phones.

3.2 Drag and Drop Assessment Creation

For instructors, assessments are an important part of a course offering. Assessments can be created for any gradable material that would normally be given to students. This key feature is called the Quiz Builder Tool (Figure 2).



[Hide/Show Toolbox & Import]

Toolbox

Import Questions

Course Offering: Computer Science I – Spring 2011

Assessment: Assign

Assessment Att

Name: Functions – C

Due Date: 4/9/2012

Section (Leave blan

Is Extra Credit: □

Allow Multiple Sub

Assessment Type :

Prerequisite Asses: None

Prerequisite Asses

Created Date: 4/8/

Is Visible: □

Is Open: □

AssessTrack
Learning Management Software

Systems and Computer Science » Computer Science I – Spring 2012

Welcome, **Harry Keeling**! [Log Off]

The current server time is 4:31 PM

Do you have suggestions for improving AssessTrack? Please email them to suggestions@assesstrack.com!

Edit Assessment

DRAG QUESTIONS ONTO THE FORM, THEN POPULATE THE QUESTIONS WITH DATA

Question 1 [X]

Text [X]

Menu

Computer Science I – Spring 2012
- Assessments
- Assessment Types
- Course Outcomes and Tags
- Exceptions
- Files
- Members
- Messages

Reminders

Recent Messages

test message from Computer Science I – Spring 2012

Question Importer ×

+

1) [20 pts]

Problem:

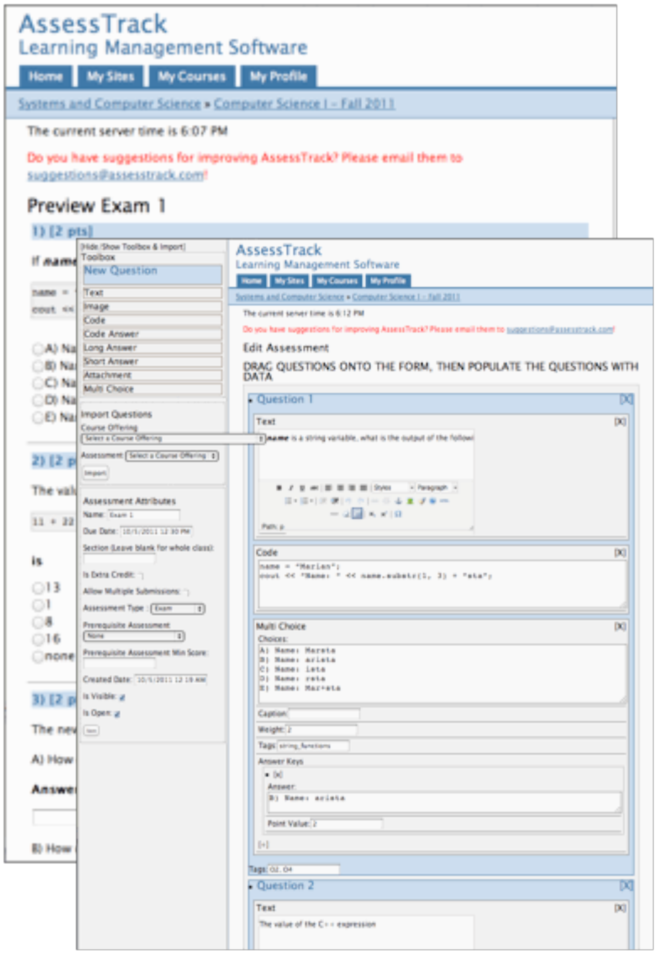
You have become fed up with waiting for Prof. Keeling to calculate your final grade. You have decided that to lower your frustrations and to help you prepare for the final exam. You decide to write a program to calculate your final grade and the grades of 19 of your classmates.

Use the structure chart below to provide the required design for your program.

Figure 2: Quiz Builder Tool

It provides instructors the ability to import components of assessments from previous semesters and to add new assessment components to build a variety of multimedia assessments that can be administered to their students and, in part, automatically graded. This tool provides a drag and drop interface that facilitates the process of creating labs, assignments, quizzes, exams or any other type of assessment instrument that an instructor needs. Survey instruments and multimedia tutorials can also be developed using the Quiz Builder.

The Toolbox, the area inside the gray box in the center of the figure, contains all the items you can use to build an assessment. The first item is the question. An assessment is composed of multiple questions. Each question is composed of one or more of the following: text, images, code, code answer, long answer, short answer, attachment answer, and multiple choice answers. In the center of the figure is a previous assessment that can be imported (in part or in total) into the current assessment and modified as needed. Once saved, assessments can be previewed and modified before deployment. Figure 3, presents a portion of an exam preview screen and the related Quiz Builder screen.



AssessTrack
Learning Management Software

Systems and Computer Science » Computer Science I – Fall 2011

The current server time is 6:07 PM

Do you have suggestions for improving AssessTrack? Please email them to suggestions@assesstrack.com!

Preview Exam 1

1) [2 pts]

[Hide/Show Toolbox & Import]

Toolbox

Import Questions

Course Offering: Select a Course Offering

Assessment: Select a Course Offering

Assessment Attributes

Name: Exam 1

Due Date: 10/1/2011 12:30 PM

Section (Leave blank for whole class):

Is Extra Credit:

Allow Multiple Submissions:

Assessment Type: Exam

Prerequisite Assessment: None

Prerequisite Assessment Min Score:

Created Date: 10/1/2011 12:10 AM

Is Visible: ✓

Is Open: ✓

AssessTrack
Learning Management Software

Systems and Computer Science » Computer Science I – Fall 2011

The current server time is 6:12 PM

Do you have suggestions for improving AssessTrack? Please email them to suggestions@assesstrack.com!

Edit Assessment

DRAG QUESTIONS ONTO THE FORM, THEN POPULATE THE QUESTIONS WITH DATA

Question 1

Text

name is a string variable, what is the output of the follow

Code

```
name = "Marlen";
cout << "Name: " << name.substr(1, 3) + "eta";
```

Multi Choice

Choices:

```
A) Name: Mareta
B) Name: arista
C) Name: leta
D) Name: reta
E) Name: Mar+eta
```

Caption:

Weight: 2

Tags: string_functions

Answer Keys

Answer:

```
B) Name: arista
```

Point Value: 2

Tags: O2, O4

Question 2

Text

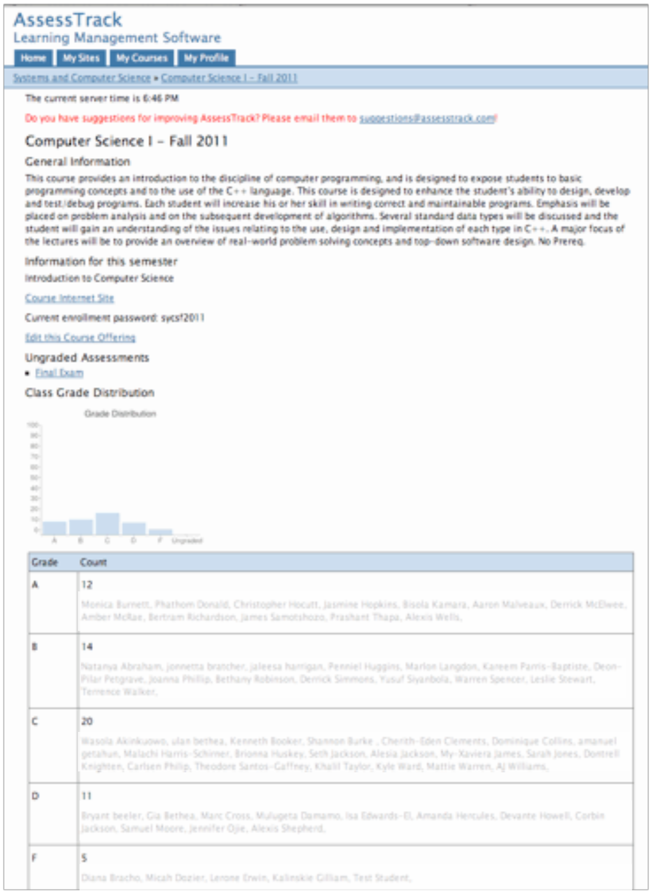
The value of the C++ expression

Figure 3: Assessment Preview Screen and Corresponding Quiz Builder Screen

Notice that in the Quiz Builder screen above, Question 1 components are shown with the correct answer "B) Name: arista" indicated in the Answer Keys area. There can be multiple correct answers to a question, each with a separate point value specified. This allows for partial or full credit to be awarded by the automated grading functions of AssessTrack. Also, notice the topic and educational outcome tags that have been entered to associate this question with the topic "string functions" and course outcomes O2 and O4.

3.3 Online Reporting Functions

Another key function of AssessTrack is the set of online reports that can be generated. The instructor's course home page, the Struggling Student screen and the Final Grades screen are shown in Figure 4.



AssessTrack
Learning Management Software

Systems and Computer Science » Computer Science I – Fall 2011

The current server time is 6:46 PM

Do you have suggestions for improving AssessTrack? Please email them to suggestions@assesstrack.com!

Computer Science I – Fall 2011

General Information

This course provides an introduction to the discipline of computer programming, and is designed to expose students to basic programming concepts and to the use of the C++ language. This course is designed to enhance the student's ability to design, develop and test/debug programs. Each student will increase his or her skill in writing correct and maintainable programs. Emphasis will be placed on problem analysis and on the subsequent development of algorithms. Several standard data types will be discussed and the student will gain an understanding of the issues relating to the use, design and implementation of each type in C++. A major focus of the lectures will be to provide an overview of real-world problem solving concepts and top-down software design. No Prereq.

Information for this semester

Introduction to Computer Science

Course Internet Site

Current enrollment password: sycsf2011

Edit this Course Offering

Ungraded Assessments

- Final Exam

Class Grade Distribution

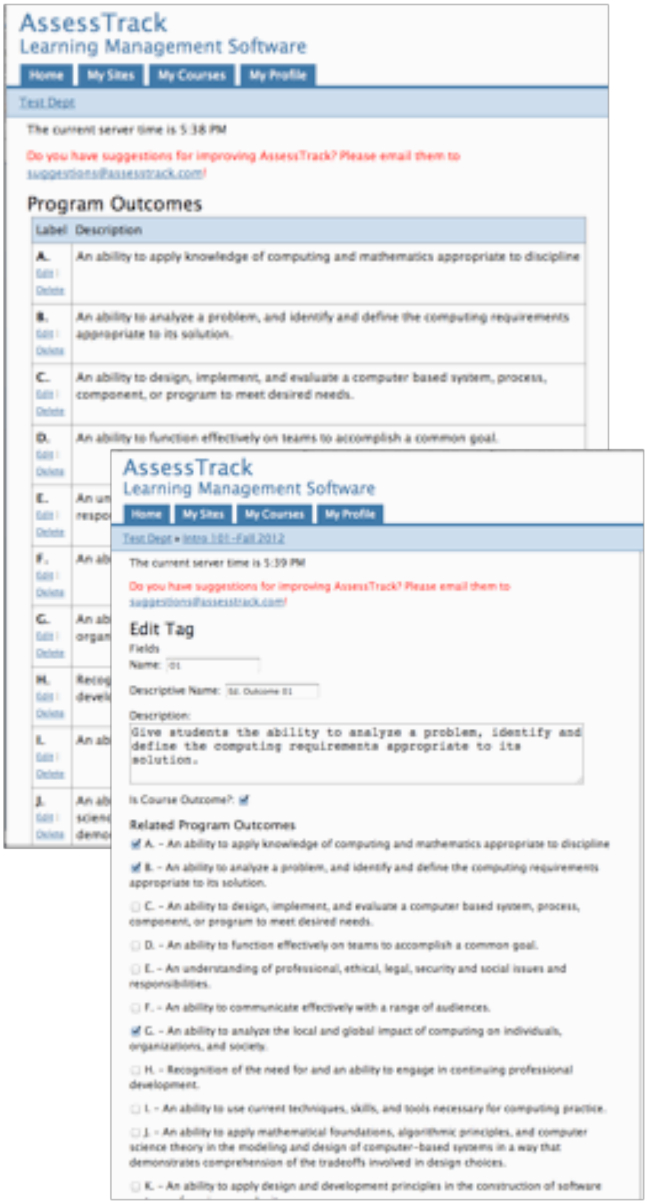
Grade	Count
A	12 Monica Burnett, Phathom Donald, Christopher Hocutt, Jasmine Hopkins, Bisola Kamara, Aaron Malveaux, Derrick McElwee, Amber McRae, Bertram Richardson, James Samotshozo, Prashant Thapa, Alexis Wells,
B	14 Natanya Abraham, jonnetta brancher, jaleesa harrigan, Penniel Huggins, Marlon Langdon, Kareem Parris-Baptiste, Deon-Pilar Petgrave, Joanna Phillip, Bethany Robinson, Derrick Simmons, Yusuf Siyanbola, Warren Spencer, Leslie Stewart, Terrence Walker,
C	20 Wasola Akinkuowo, ulan bethea, Kenneth Booker, Shannon Burke , Cherith-Eden Clements, Dominique Collins, amanuel getahun, Malachi Harris-Schirner, Brionna Huskey, Seth Jackson, Alesia Jackson, My-Xaviera James, Sarah Jones, Dontrell Knighten, Carlsen Philip, Theodore Santos-Gaffney, Khalil Taylor, Kyle Ward, Mattie Warren, AJ Williams,
D	11 Bryant beeler, Gia Bethea, Marc Cross, Mulugeta Damamo, Isa Edwards-El, Amanda Hercules, Devante Howell, Corbin Jackson, Samuel Moore, Jennifer Ojie, Alexis Shepherd,
F	5 Diana Bracho, Micah Dozier, Lerone Erwin, Kalinskie Gilliam, Test Student,

Figure 4: Course Home Page Screen

3.4 Topic and Course Outcome Tags

Once a program administrator (department Chair) has entered their program outcomes, instructors can associate these outcomes with their individual course educational outcomes. Figure 5 illustrates the Program Outcomes screen and the Edit Tags Screen. The Edit Tags screen shows how education course outcomes are associated with one or more program outcomes.

By "tagging" each assessment item to indicate the course topic(s) and/or the educational outcomes that it is intended to assess (see Figure 3), the instructor designates a purpose(s) for each assessment item. This important function makes the assessment and tracking of course topics and educational outcomes possible. Further, this tagging process allows AssessTrack to generate key program self-study reports. Also, the tagging of assessment items with course topics using the Quiz Builder tool provides the basis for AssessTrack's intelligent tutor. Each student's scores on assessment items form a "student model" of each student's competencies and their understanding of the course content. The system's tutoring component uses this student model and the topic tags on each assessment component to generate an individualized study guide for each student.



Test Dept

The current server time is 5:38 PM

Do you have suggestions for improving AssessTrack? Please email them to suggestions@assesstrack.com!

Program Outcomes

Label	Description
A. Edit | Delete	An ability to apply knowledge of computing and mathematics appropriate to discipline
B. Edit | Delete	An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution.
C. Edit | Delete	An ability to design, implement, and evaluate a computer based system, process, component, or program to meet desired needs.
D. Edit | Delete	An ability to function effectively on teams to accomplish a common goal.
E. Edit | Delete	An un respo
F. Edit | Delete	An abi
G. Edit | Delete	An abi organ
H. Edit | Delete	Recog devel
I. Edit | Delete	An abi
J. Edit | Delete	An abi scienc demo

Test Dept » Intro 101 - Fall 2012

The current server time is 5:39 PM

Do you have suggestions for improving AssessTrack? Please email them to suggestions@assesstrack.com!

Edit Tag

Fields

Name: O1

Descriptive Name: Ed. Outcome 01

Description:

```
Give students the ability to analyze a problem, identify and
define the computing requirements appropriate to its
solution.
```

Is Course Outcome?: ☑

Related Program Outcomes

☑ A. - An ability to apply knowledge of computing and mathematics appropriate to discipline

☑ B. - An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution.

☐ C. - An ability to design, implement, and evaluate a computer based system, process, component, or program to meet desired needs.

☐ D. - An ability to function effectively on teams to accomplish a common goal.

☐ E. - An understanding of professional, ethical, legal, security and social issues and responsibilities.

☐ F. - An ability to communicate effectively with a range of audiences.

☑ G. - An ability to analyze the local and global impact of computing on individuals, organizations, and society.

☐ H. - Recognition of the need for and an ability to engage in continuing professional development.

☐ I. - An ability to use current techniques, skills, and tools necessary for computing practice.

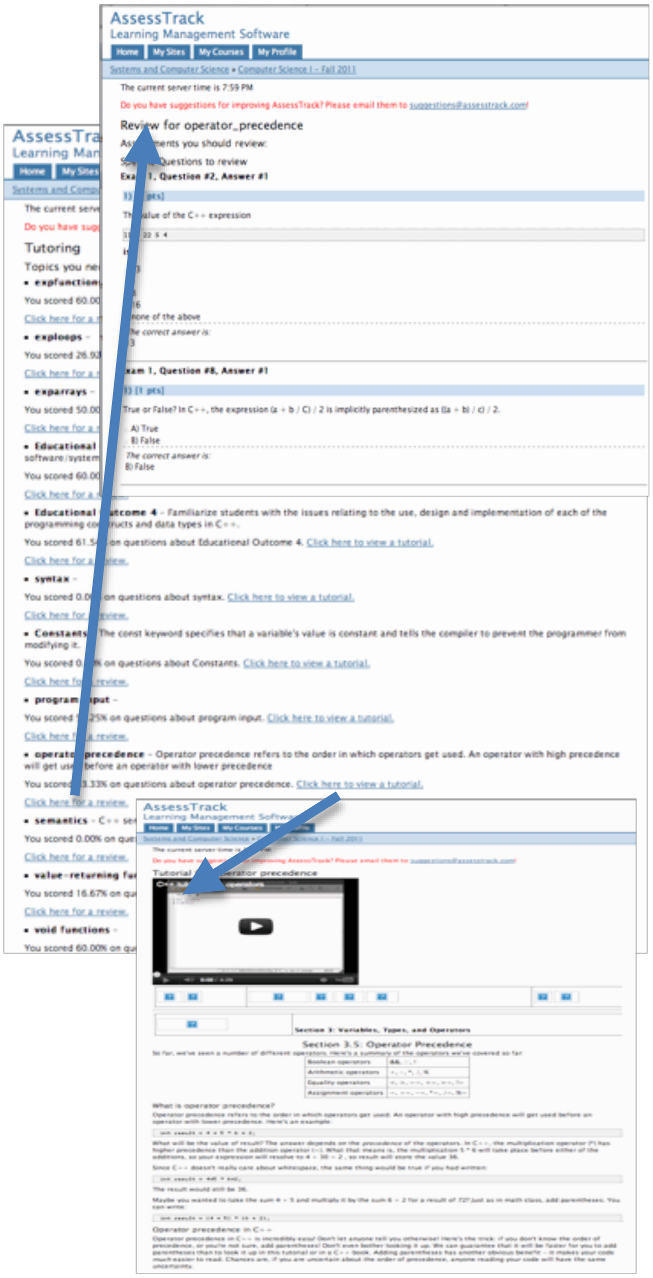
☐ J. - An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices.

☐ K. - An ability to apply design and development principles in the construction of software

Figure 5: Program Outcomes screen and Edit Tag screen

3.5 Intelligent Tutoring

Building on research conducted by the author earlier [8, 9], an intelligent tutoring feature was added to the system last year. Figure 6 presents an example of a study guide for a student who performed poorly on several topics on exams and quizzes. This figure illustrates the Tutoring screen (middle right). This screen is generated by the intelligent tutoring feature and presents the list of topics in which the student has shown a lack of understanding. The Topic Review screen (upper right) shows the topic-related questions that the student answered incorrectly along with the correct answers. The Tutorial screen shows the system-selected remediation (video clips, course material, and book references) collected to address this student's illustrated lack of understanding of "operator precedence".



Systems and Computer Science » Computer Science I - Fall 2011

The current server time is 7:59 PM

Do you have suggestions for improving AssessTrack? Please email them to suggestions@assesstrack.com!

Review for operator_precedence

As[illegible]ents you should review:

S[illegible] Questions to review

Exa[illegible] 1, Question #2, Answer #1

1) [illegible] pts]

Th[illegible] value of the C++ expression

11 [illegible] 22 5 4

is

[illegible]3

[illegible]6

none of the above

The correct answer is:

3

[illegible]xam 1, Question #8, Answer #1

1) [1 pts]

True or False? In C++, the expression (a + b / C) / 2 is implicitly parenthesized as ((a + b) / c) / 2.

A) True
B) False

The correct answer is:
B) False

Systems and Comp[illegible]

The current serve[illegible]

Do you have sugg[illegible]

Tutoring

Topics you nee[illegible]

- **expfunction[illegible]**

You scored 60.00[illegible]

Click here for a r[illegible]

- **exploops** -

You scored 26.92[illegible]

Click here for a r[illegible]

- **exparrays** -

You scored 50.00[illegible]

Click here for a r[illegible]

- **Educational** [illegible] software/system[illegible]

You scored 60.00[illegible]

Click here for a r[illegible]

- **Educational [illegible]utcome 4** - Familiarize students with the issues relating to the use, design and implementation of each of the programming co[illegible]ructs and data types in C++.

You scored 61.5[illegible] on questions about Educational Outcome 4. Click here to view a tutorial.

Click here for a [illegible]view.

- **syntax** -

You scored 0.0[illegible] on questions about syntax. Click here to view a tutorial.

Click here for [illegible]view.

- **Constants** [illegible] The const keyword specifies that a variable's value is constant and tells the compiler to prevent the programmer from modifying it.

You scored 0.[illegible]% on questions about Constants. Click here to view a tutorial.

Click here fo[illegible] review.

- **program [illegible]put** -

You scored [illegible]25% on questions about program input. Click here to view a tutorial.

Click here f[illegible] a review.

- **operato[illegible] precedence** - Operator precedence refers to the order in which operators get used. An operator with high precedence will get use[illegible] before an operator with lower precedence

You score[illegible]3.33% on questions about operator precedence. Click here to view a tutorial.

Click here for a review.

- **semantics** - C++ se[illegible]

You scored 0.00% on que[illegible]

Click here for a review.

- **value-returning fu[illegible]**

You scored 16.67% on qu[illegible]

Click here for a review.

- **void functions** -

You scored 60.00% on qu[illegible]

Systems and Computer Science » [illegible] - Fall 2011

The current server time is [illegible]

Do you have [illegible] improving AssessTrack? Please email them to suggestions@assesstrack.com!

Tutorial [illegible]erator precedence

Section 3: Variables, Types, and Operators

Section 3.5: Operator Precedence

So far, we've seen a number of different operators. Here's a summary of the operators we've covered so far:

Boolean operators	&&, ||, !
Arithmetic operators	+, -, *, /, %
Equality operators	[illegible]
Assignment operators	=, +=, -=, *=, /=, %=

What is operator precedence?

Operator precedence refers to the order in which operators get used. An operator with high precedence will get used before an operator with lower precedence. Here's an example:

```
int result = 4 + 5 * 6 + 2;
```

What will be the value of result? The answer depends on the precedence of the operators. In C++, the multiplication operator (*) has higher precedence than the addition operator (+). What that means is, the multiplication 5 * 6 will take place before either of the additions, so your expression will resolve to 4 + 30 + 2 , so result will store the value 36.

Since C++ doesn't really care about whitespace, the same thing would be true if you had written:

```
int result = 4+5 * 6+2;
```

The result would still be 36.

Maybe you wanted to take the sum 4 + 5 and multiply it by the sum 6 + 2 for a result of 72? Just as in math class, add parentheses. You can write:

```
int result = (4 + 5) * (6 + 2);
```

Operator precedence in C++

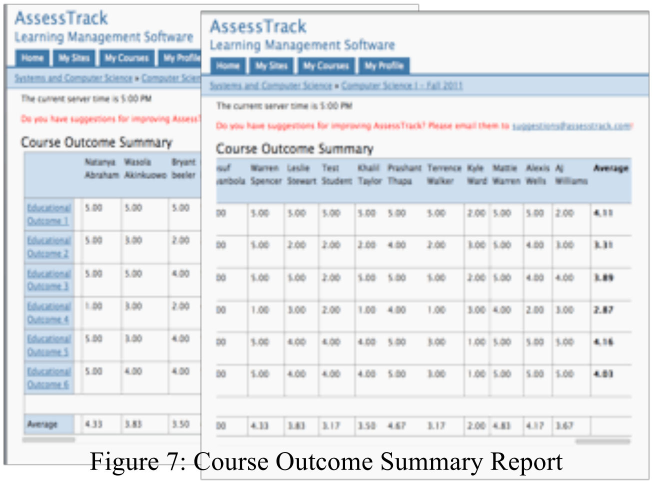
Operator precedence in C++ is incredibly easy! Don't let anyone tell you otherwise! Here's the trick: if you don't know the order of precedence, or you're not sure, add parentheses! Don't even bother looking it up. We can guarantee that it will be faster for you to add parentheses than to look it up in this tutorial or in a C++ book. Adding parentheses has another obvious benefit -- it makes your code much easier to read. Chances are, if you are uncertain about the order of precedence, anyone reading your code will have the same uncertainty.

Figure 6: Tutoring, Topic Review and Tutorial screens

This feature gives each student a system-generated, online study guide that focuses on his or her topic-related deficiencies.

3.6 Self-Study Reports

Several reports are printed by AssessTrack for entry into the Course Books that are required by ABET as a result of institutional self-study. Figure 7 presents one of the most important reports required by ABET, the Course Outcome Summary Report. This report analyzes attainment of each course outcome. The metric used (1 to 5) is computed from the student's percent of accuracy on assessment items associated with each course outcome. The average measure of all students for each course outcome is shown on the far right.



AssessTrack
Learning Management Software

Systems and Computer Science » Computer Scien

The current server time is 5:00 PM

Do you have suggestions for improving Assess

Course Outcome Summary

	Natanya Abraham	Wasola Akinkuowo	Bryant beeler
Educational Outcome 1	5.00	5.00	5.00
Educational Outcome 2	5.00	3.00	2.00
Educational Outcome 3	5.00	5.00	4.00
Educational Outcome 4	1.00	3.00	2.00
Educational Outcome 5	5.00	3.00	4.00
Educational Outcome 6	5.00	4.00	4.00
Average	4.33	3.83	3.50

AssessTrack
Learning Management Software

Systems and Computer Science » Computer Science I - Fall 2011

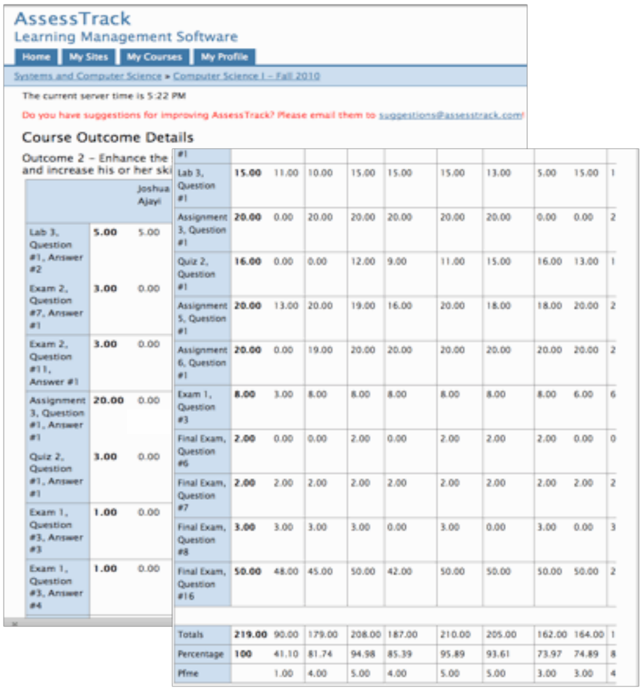
The current server time is 5:00 PM

Do you have suggestions for improving AssessTrack? Please email them to suggestions@assesstrack.com!

Course Outcome Summary

suf anbola	Warren Spencer	Leslie Stewart	Test Student	Khalil Taylor	Prashant Thapa	Terrence Walker	Kyle Ward	Mattie Warren	Alexis Wells	Aj Williams	Average
00	5.00	5.00	5.00	5.00	5.00	5.00	2.00	5.00	5.00	2.00	4.11
00	5.00	2.00	2.00	2.00	4.00	2.00	3.00	5.00	4.00	3.00	3.31
00	5.00	5.00	2.00	5.00	5.00	5.00	2.00	5.00	4.00	4.00	3.89
00	1.00	3.00	2.00	1.00	4.00	1.00	3.00	4.00	2.00	3.00	2.87
00	5.00	4.00	4.00	4.00	5.00	3.00	1.00	5.00	5.00	5.00	4.16
00	5.00	4.00	4.00	4.00	5.00	3.00	1.00	5.00	5.00	5.00	4.03
00	4.33	3.83	3.17	3.50	4.67	3.17	2.00	4.83	4.17	3.67	

Figure 7: Course Outcome Summary Report



AssessTrack
Learning Management Software

Systems and Computer Science » Computer Science I - Fall 2010

The current server time is 5:22 PM

Do you have suggestions for improving AssessTrack? Please email them to suggestions@assesstrack.com!

Course Outcome Details

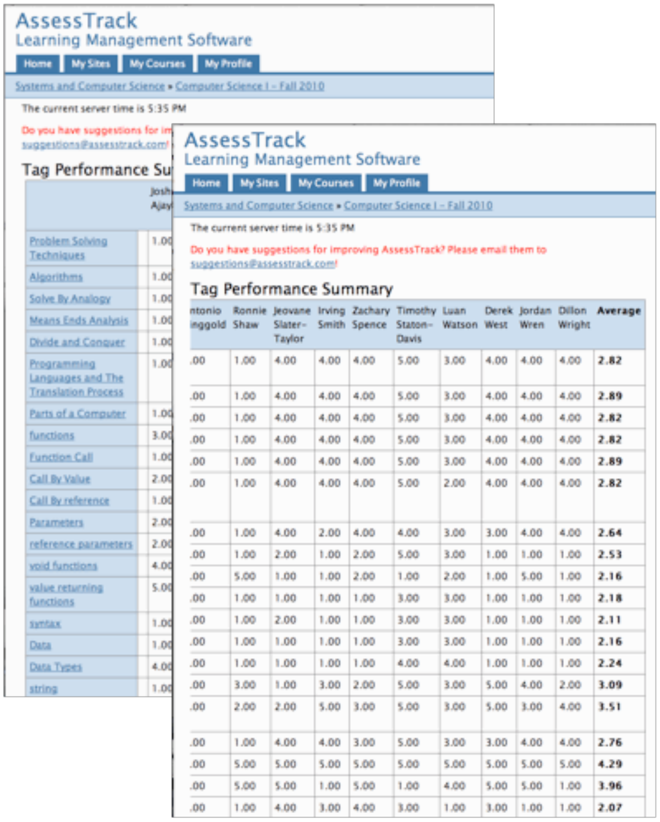
Outcome 2 - Enhance the
and increase his or her ski

		Joshua Ajayi
Lab 3, Question #1, Answer #2	5.00	5.00
Exam 2, Question #7, Answer #1	3.00	0.00
Exam 2, Question #11, Answer #1	3.00	0.00
Assignment 3, Question #1, Answer #1	20.00	0.00
Quiz 2, Question #1, Answer #1	3.00	0.00
Exam 1, Question #3, Answer #3	1.00	0.00
Exam 1, Question #3, Answer #4	1.00	0.00

#1										
Lab 3, Question #1	15.00	11.00	10.00	15.00	15.00	15.00	13.00	5.00	15.00	1
Assignment 3, Question #1	20.00	0.00	20.00	20.00	20.00	20.00	20.00	0.00	0.00	2
Quiz 2, Question #1	16.00	0.00	0.00	12.00	9.00	11.00	15.00	16.00	13.00	1
Assignment 5, Question #1	20.00	13.00	20.00	19.00	16.00	20.00	18.00	18.00	20.00	2
Assignment 6, Question #1	20.00	0.00	19.00	20.00	20.00	20.00	20.00	20.00	20.00	2
Exam 1, Question #3	8.00	3.00	8.00	8.00	8.00	8.00	8.00	8.00	6.00	6
Final Exam, Question #6	2.00	0.00	0.00	2.00	0.00	2.00	2.00	2.00	0.00	0
Final Exam, Question #7	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2
Final Exam, Question #8	3.00	3.00	3.00	3.00	0.00	3.00	0.00	3.00	0.00	3
Final Exam, Question #16	50.00	48.00	45.00	50.00	42.00	50.00	50.00	50.00	50.00	2
Totals	219.00	90.00	179.00	208.00	187.00	210.00	205.00	162.00	164.00	1
Percentage	100	41.10	81.74	94.98	85.39	95.89	93.61	73.97	74.89	8
Pfme		1.00	4.00	5.00	4.00	5.00	5.00	3.00	3.00	4

Figure 8: Details for Course Educational Outcome 2

Figure 8 shows the details for Outcome 2. The averages at the end of this report are computed per ABET instructions. Figure 9 presents another report used by instructors to determine how well students have understood the concepts and topics taught in a course.



AssessTrack
Learning Management Software

Systems and Computer Science » Computer Science I - Fall 2010

The current server time is 5:35 PM

Do you have suggestions for im
suggestions@assesstrack.com!

Tag Performance Su

	Josh Ajay
Problem Solving Techniques	1.00
Algorithms	1.00
Solve By Analogy	1.00
Means Ends Analysis	1.00
Divide and Conquer	1.00
Programming Languages and The Translation Process	1.00
Parts of a Computer	1.00
functions	3.00
Function Call	1.00
Call By Value	2.00
Call By reference	1.00
Parameters	2.00
reference parameters	2.00
void functions	4.00
value returning functions	5.00
syntax	1.00
Data	1.00
Data Types	4.00
string	1.00

AssessTrack
Learning Management Software

Systems and Computer Science » Computer Science I - Fall 2010

The current server time is 5:35 PM

Do you have suggestions for improving AssessTrack? Please email them to suggestions@assesstrack.com!

Tag Performance Summary

ntonio inggold	Ronnie Shaw	Jeovane Slater-Taylor	Irving Smith	Zachary Spence	Timothy Staton-Davis	Luan Watson	Derek West	Jordan Wren	Dillon Wright	Average
.00	1.00	4.00	4.00	4.00	5.00	3.00	4.00	4.00	4.00	2.82
.00	1.00	4.00	4.00	4.00	5.00	3.00	4.00	4.00	4.00	2.89
.00	1.00	4.00	4.00	4.00	5.00	3.00	4.00	4.00	4.00	2.82
.00	1.00	4.00	4.00	4.00	5.00	3.00	4.00	4.00	4.00	2.82
.00	1.00	4.00	4.00	4.00	5.00	3.00	4.00	4.00	4.00	2.89
.00	1.00	4.00	4.00	4.00	5.00	2.00	4.00	4.00	4.00	2.82
.00	1.00	4.00	2.00	4.00	4.00	3.00	3.00	4.00	4.00	2.64
.00	1.00	2.00	1.00	2.00	5.00	3.00	1.00	1.00	1.00	2.53
.00	5.00	1.00	1.00	2.00	1.00	2.00	1.00	5.00	1.00	2.16
.00	1.00	1.00	1.00	1.00	3.00	3.00	1.00	1.00	1.00	2.18
.00	1.00	2.00	1.00	1.00	3.00	3.00	1.00	1.00	1.00	2.11
.00	1.00	1.00	1.00	1.00	3.00	3.00	1.00	1.00	1.00	2.16
.00	1.00	1.00	1.00	1.00	4.00	4.00	1.00	1.00	1.00	2.24
.00	3.00	1.00	3.00	2.00	5.00	3.00	5.00	4.00	2.00	3.09
.00	2.00	2.00	5.00	3.00	5.00	3.00	5.00	3.00	4.00	3.51
.00	1.00	4.00	4.00	3.00	5.00	3.00	3.00	4.00	4.00	2.76
.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	4.29
.00	5.00	5.00	1.00	5.00	1.00	4.00	5.00	5.00	1.00	3.96
.00	1.00	4.00	3.00	4.00	3.00	1.00	3.00	1.00	1.00	2.07

Figure 9: Tag Performance Summary (beginning and end)

These reports (Figures 7, 8, and 9) show analyses of student performance and educational outcomes. These reports can be viewed at any point in the semester. Subsequently, the instructor can modify the instructional process to address any indicated deficiencies. This type of formative evaluation has been encouraged by ABET.

4.0 EVALUATION

Last year, students responded to a survey directed, in part, at measuring their impressions of AssessTrack. The survey itself was developed using AssessTrack's QuizBuilder feature and was analyzed using question tags and the software's tag-based reporting feature. Using a 5-point Likert scale (5-strongly agree, 4-agree, 3-neither agree or disagree, 2-disagree, 1-strongly disagree) the survey results on the next page were tabulated.

Average Response	Question
3.7	Taking exams, quizzes and assignments online made the work more enjoyable and fun
3.96	The online course management system improved communication between the teacher and student
4.37	The website was easy to manage and understand
4.63	Seeing your grades displayed immediately on the website helped give you a true understanding of where you stood in the class

The results above indicate clearly that students enjoy the fact that AssessTrack provides easy, online access to educational material and assessment results. There was also a noticeable improvement with timely submission of student assignments. This improvement, we believe, was due to the submission deadlines enforced by the AssessTrack software. Also, the survey asked two questions directed at measuring the software's impact on studying.

Average Response	Question
4.63	Having everything located online made things easier to access and catch up when absent from school
4.11	The class slides, lectures, notes, etc. available online made things more maintainable for studying

The students indicated that the tutoring materials (short videos, text references, and other material) help them study by providing access from anywhere there is web access.

Learners utilized the tutoring functions of AssessTrack for two semesters last year and there was clear evidence that it enhances student learning. In co-developer Robert Person's master thesis [10] he studied the impact of the tutorials from AssessTrack's intelligent tutor. He found that the introduction of supplemental teaching materials have had a positive impact on student learning. The data revealed that there was a 12% increase in performance among those students who made use of these tutorials.

5. CONCLUSIONS

Clearly, the accreditation self-study reports that are generated by this LMS will be an enormous benefit to both the instructors and the ABET accreditation team on their next visit. Moreover, overall teaching effectiveness has improved with the advent of AssessTrack. The improved ability to prepare, disseminate and automatically grade some assessment items provides immediate feedback to both student and teacher. The software improved the way that we conceptualize our students by providing a "student model" showing their individual weaknesses and, by default, their strengths. Being able to analyze the class's performance at the question level helps the teacher prepare better assessment items. Further, the ability to see the class's performance by course educational outcome gives the instructor a unique view of their teaching effectiveness in a formative manner that can directly affect the time spent on course topics and the manner in which course content is delivered.

For students, AssessTrack gives online access to course materials including video taped lectures. Via any web-accessible device, students currently receive assignments, submit their work, and take traditional assessments (labs, quizzes, and exams). Further, AssessTrack gives learners 24/7 access to detailed analysis about how well they are doing in a course and perhaps more importantly, it helps them clearly identify their areas of weakness. There is a mobile version near completion designed specifically for use on tablets and smart phones. We look forward to continuing this research advancing our tutoring methods and adding elements of social networking.

6. REFERENCES

[1] Hart, J. Directory of Learning & Performance Tools, 2012, Retrieved Aug 2, 2012 from Centre for learning & Performance Technologies: http://c4lpt.co.uk/directory-of-learning-performance-tools/.

[2] Harrington, C., Gordon, S. and Schibik, T. Course Management System Utilization and Implications for Practice: A National Survey of Department Chairpersons, *Online Journal of Distance Learning Administration, 7*(4) (2004). Retrieved July 15, 2012 from State University of West Georgia, Distance Education Center: http://www.westga.edu/~distance/ojdla/winter74/harrington74.htm.

[3] Nijhuis, G. and Collis, B. Using a web-based course-management system: an evaluation of management tasks and time implications for the instructor. *Evaluation and Program Planning, 26* (2). 193-201.

[4] Unal, Z. and Unal, A. Evaluating and Comparing the Usability of Web-based Course Management Systems. *Journal of Information Technology Education, 10*, (2011). Retrieved Jan. 5, 2012: http://www.jite.org/documents/Vol10/JITEv10p019-038Unal904.pdf.

[5] Johnson, D., Means, T., and Graff, R. Student Perceptions of Course Management Systems Tools – Implications for evaluation and adoption of online Tools in Higher Education, Retrieved June 14, 2012 from EDUCAUSE, Center for Applied Research: http://net.educause.edu/ir/library/pdf/ERB1202.pdf.

[6] Swan, K., Matthews, D., Boles, E. and Day, S. Linking online course design and implementation to learning outcomes: A design experiment. *The Internet and Higher Education*, *15*(2). 81–88.

[7] Information Technology ABET Accreditation Summary, ACM Special Interest Group for Information Technology Education, Retrieved Jan. 5, 2012, from http://www.acm.org/search?SearchableText=program+ Accreditation.

[8] Keeling, H. A Methodology for Building Intelligent Educational Agents. in *9th International Conference on Artificial Intelligence in Education,* (Lemans, France, 1999), Springer Publishers, 183-204.

[9] Keeling, H. Developing an Intelligent Educational Agent with Disciple. *In The International Journal of Artificial Intelligence in Education, 10*(2) 5-16.

[10] Person, R. A Framework for Measuring the Impact of Supplementary Learning Content in Online Learning Management Systems. *Masters Thesis*, Dept. Systems and Computer Science, Howard University.

A Bottom-Up Outcome-Based Learning Assessment Process for Accrediting Computing Programs

Haidar M. Harmanani
Department of Computer Science and Mathematics
Lebanese American University
Byblos, 1401 2010, Lebanon

Abstract—**The push for a *culture of evidence* that guides improvement in higher-education has made outcome-based assessment a necessity. Furthermore, the recent move by ABET's CAC into more rigorous assessment has caused anxiety among faculty and administrators. Assessment leaders face various challenges including *process design and implementation, faculty buy-in*, and *resources availability*. This paper presents a bottom-up outcome-based assessment approach that facilitates faculty participation while simplifying the assessment and reporting processes. The proposed approach has been implemented and used for the successful accreditation of a computer science program, and can be easily adapted to any higher education program.**

Keywords: Program Outcome Assessment, ABET Accreditation

I. Introduction

Assessing learning has been at the heart of higher education ever since its inception; *grades* as well as *retention* and *graduation rates* have always been used as measures of students performance. However, over the past 40 years, there has been an increasing pressure on higher education in order to create a *culture of evidence* that guides improvements in teaching and learning. While the demand to establish such a culture appears to be new, it has a long lineage and can be traced back as far as 1936 when the *Carnegie Foundation for the Advancement of Teaching* made a strong argument for improving higher education through a process that includes setting up acceptable standards and devising appropriate methods of measuring achievement while introducing flexibility in educational offerings [1]. Assessment of students learning has consequently progressed from the rise and development of standardized tests, to the assessment of learning for general and graduate education, and finally to the current era of external accountability [2]. A recent study by the *National Institute for Learning Outcomes Assessment* has concluded that the most common uses of assessment data are to primarily prepare for program and institutional accreditation [3]; among all schools, the least common uses for assessment data are for making daily resource decisions, admissions and transfer policies, and faculty/staff performance. The study also noted that most institutions conduct learning outcomes assessment on a *shoestring* while gaining faculty involvement and support in assessment remains a major challenge. The push for the "culture of evidence" has been capped by a new wave of outcomes-based assessment wars among professional and regional accreditation agencies. For example, the senior college commission of the Western Association of Schools and Colleges (WASC) adopted changes that shift the focus from institutions being merely engaged in assessment to being about what the results are and what they mean [4]. ABET accredits computing and engineering programs which need to demonstrate, among others, that they meet a set of program outcomes under criterion 3 (known as the ABET's *a-k* in computing and engineering, ABET's *a-j* in IS, and ABET's *a-n* in IT).

Outcomes-based assessment is a learning assessment approach where students are expected to acquire specific skills, knowledge and behavior as they progress through a course or a program [5]. Assessment can be *formative* or *summative* and is typically tackled at the *classroom level*, the *program level*, or the *institutional level*.

This paper describes an outcome-based assessment approach for accrediting computer science programs based on an effective course level assessment that facilitates and enhances faculty participation. The proposed approach is based on a bottom-up model that can be easily adapted to any program. In fact, the approach is currently being used in order to assess learning in various arts and sciences programs in the School of Arts and Sciences at the Lebanese American University. The remainder of this paper is organized as follows. Section II describes ABET computing accreditation and assessment while section III presents the proposed learning assessment and planning model. Section IV describes the assessment process and reporting while section V presents the evaluation process and closing the loop. We conclude with remarks and observations in section VI.

II. Background

ABET's Computing Accreditation Commission (CAC) accredits computing programs that demonstrate meeting nine well defined criteria. Currently, there are a total of 344 ABET-accredited computing programs with 260, 37, 22, and 21 programs accredited under the curricula of *computer science*, *information systems*, *software engineering*, and *information technology*, respectively. ABET also currently ac-

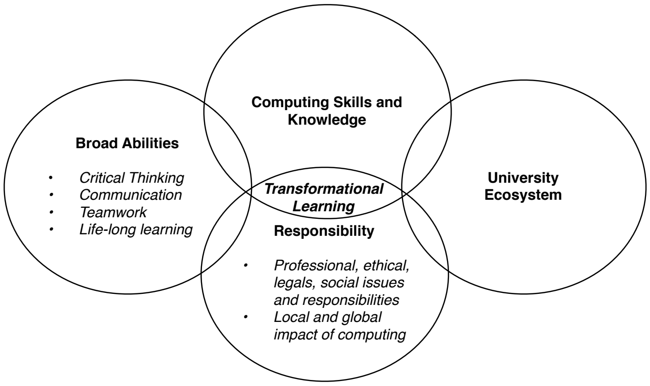

Computing Skills and Knowledge
Broad Abilities
Critical Thinking
Communication
Teamwork
Life-long learning
Transformational Learning
University Ecosystem
Responsibility
Professional, ethical, legals, social issues and responsibilities
Local and global impact of computing

Fig. 1. ABET Computing Learning Outcomes

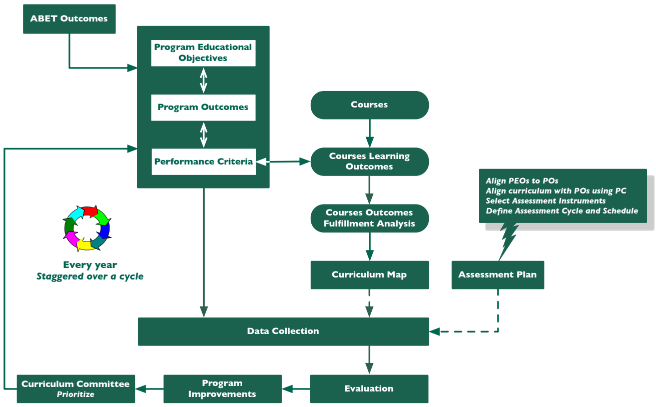


Fig. 2. Assessment Process

credits 9 programs under the general CAC criteria. Criterion 4, *continuous improvement* [6], states that an accredited program "must regularly use a appropriate, documented processes for assessing and evaluating the extent to which the student outcomes are being attained. The results of these evaluations must be systematically utilized as input for the continuous improvement of the program. Other available information may also be used to assist in the continuous improvement of the program." Because of its relative novelty in computing accreditation, the implementation of effective outcome-based assessment processes has caused anxiety among faculty and administrators in their quest for ABET accreditation [7].

ABET's push for a culture of evidence has demonstrated that a collegial model for curricular and pedagogical reform represents a more effective approach for improving curricula in higher education, in contrast to the recommendations of the Spellings Commission's that calls for federal reliance upon regulatory, monitoring and inducement strategies. Outcome-based assessment represents a major shift of ABET accreditation focus from curricular audit to include learning assessment by requiring that ABET accredited programs demonstrate that they meet three sets of outcomes (Figure 1). The first set includes knowledge that students should acquire in the major. The measurement of these outcomes has proven to be rather difficult since faculty within a discipline do not necessarily agree neither on the meaning of such specific outcomes nor on how to measure them. The second set of learning outcomes include broad abilities such as how to think critically, acquire life-long learning, and communicate clearly and concisely. The third set of learning outcomes relate to individual ethical and social responsibilities. When considered together and supported by a college's environmental ecology, the ABET learning outcomes should contribute to transformational learning outcomes [2].

The assessment process should build a body of evidence to improve the program. Such a process should evolve around the *program's mission*, *objectives*, and *outcomes*. The program educational objectives should be aligned with the needs of the program constituencies and the institution's mission. The student outcomes should be measurable, based on the needs of the program's constituencies. Furthermore, the program should develop a systematic and on-going process where clear instruments are adopted in order to acquire assessment information over time. The other caveat is that CAC requires that the program *enable* the CAC student outcomes, which may not necessarily be the same as the program's student outcomes.

Assessment instruments should be multi-faceted and include direct and indirect methods. Typically indirect methods include *students or faculty surveys*, *alumni surveys*, *external advisory boards*, *focus groups*, *exit surveys*, and *exit interviews*. Direct assessment methods include programming *assignments*, *projects*, *in-class tests*, *portfolios*, *oral presentations* and the *ETS's Major Field Test* [8], [9], [10], [11]. Blanford *et al.* [8] propose personal class assessment which is a course-based assessment tool in which the instructor writes an assessment of the course being taught. The authors also propose an assessment day as an effective way for faculty to meet, evaluate assessment results, and provide improvement recommendations. Typically each instrument has its own advantages and limitations and thus it is typically advised to mix different assessment tools for triangulation purposes [12]. Assessment should be followed by evaluation where thresholds are established in order to analyze and report the data to the stakeholders. Data are analyzed to identify problems and solutions must be proposed and implemented [13]. Crouch [5] recommends the formation of a departmental steering committee of senior faculty members to consolidate all course outcomes into a final set of outcomes.

III. Computer Science Outcome-Based Assessment

Established in 1924, the Lebanese American University (LAU) is chartered by the *Board of Regents of the State University of New York* and is accredited by the *New England*

Association of Schools and Colleges (NEASC). The Computer Science program was established in 1977, and currently offers degrees leading to a *Bachelor of Science in Computer Science*, and a *Master of Science in Computer Science*. Currently, the program has 217 students. In 2008, LAU embarked on accrediting all professional programs, including the computer science program. The accreditation of the computer science program was successfully completed in two years, and resulted with a wide awareness of outcome-based assessment and systematic program review and improvement. The Program implemented an efficient bottom-up outcome-based assessment process that integrates learning assessment at the course and at the program levels. The process, shown in Figure 2, includes: *assessment planning*, *the selection of performance criteria*, *curriculum mapping*, *program assessment*, *program evaluation*, and *closing the loop*.

A. Assessment Planning

Faculty participation in program assessment and curricular development remains a challenge as these time-consuming activities take faculty away from *research* and *publication* which remain at most universities highly valued in promotion, tenure, and merit-pay decisions. On the other hand, faculty feel strongly about “their” courses, which are the main strategy that are used to assess the program outcomes. Strategies are implemented for course learning outcomes in order to support the program outcomes, and ultimately the program educational objectives. Program objectives and outcomes do not need to be assessed every semester; however, it is essential that assessment activities integrate the assessment of all outcomes over the regular assessment cycle. Typically, the following are noted:

(a) Assessment should not only be used to satisfy accreditation, but also as a major internal driver for systematic program review and improvement;
(b) Learning assessment at the course level can be formative as well as summative, and is implemented over the span of one semester. It has the advantages of broad faculty involvement and easy implementation;
(c) Learning assessment at the program level, as required by ABET Accreditation, is summative. Assessment and improvements are typically implemented over a 3-4 years cycle.

In what follows, we describe the various steps involved in planning assessment in a bottom-up fashion, including the mapping of courses to program outcomes at the course outcome level, the selection of assessment instruments, performance criteria, and curricular mapping.

1) Aligning Courses and Student Outcomes: Assessment planning should start *concurrently* at the course as well as at the program level. Thus, at the course level, each course is assigned two “owners” in the form of a *coordinator* and a *co-coordinator*. The rational for having two coordinators is two-fold. On one hand, this engages experts in the area and increases faculty buy-in in assessment. On the other hand, the coordinator and co-coordinator constitute a committee of area experts that provides oversight and broad perspective in the collection and review of the assessment evidence in the course. For each course, the committee of area experts shall:

(a) Establish and review the course learning outcomes;
(b) Select the assessment instruments that will be used to measure each course learning outcome;
(c) Establish an acceptable performance level;
(d) Oversee courses including the selection of textbooks, the monitoring of course delivery and content, the coordination of the assessment process, and the follow up with the chairperson on any issues that may arise;
(e) Collect data at the course-outcome level under the related program outcomes.

Assessment planning at the program level starts with a departmental steering committee that reviews all program objectives and outcomes in order to ensure that they are *measurable*, *realistic*, *contemporary*, and *aligned* with the School’s and University missions as well as with the needs of the program’s constituencies. It is recommended at this stage not to get involved in lengthy debates or controversial changes as such changes should be the output of the assessment process.

2) Assessment Instruments: The program’s assessment plan should measure the program outcomes using multiple instruments in order to allow for triangulation in the assessment process. The assessment instruments vary depending on what the committee of experts is trying to learn, and on the nature of the course. The instruments that we have implemented and adopted included *embedded assessment*, *scoring rubrics*, *exit surveys*, *students meetings*, and *locally developed exams*.

3) Performance Criteria Selection: The measurement of program outcomes, especially the technical ones, has proven to be rather elusive and difficult since faculty within a discipline do not necessarily agree neither on the meaning of such outcomes nor on how to measure them. In order to resolve this issue, ABET *recommends* the use of performance criteria which are “specific, measurable statements identifying the performance(s) required to meet the outcome, confirmable through evidence."

In order to facilitate the assessment process, we use a bottom-up process where course outcomes are selected as performance criteria. Thus, courses coordinators can easily establish the relationship between student outcomes and courses outcomes using three learning levels, *introductory*, *reinforcement*, and *emphasis* levels [14]. This process is facilitated using an outcome-fulfillment form, that establishes a course map between the program outcomes and the course outcomes. This process can be facilitated using a simple “course map” that establishes a learning relationship between program outcomes and courses learning outcomes. The course map uses the following three learning levels:

- *Introduction level* where students are not expected to be familiar with the content or skill at the collegiate or graduate level. Instruction and learning activities focus on basic knowledge, skills, and/or competencies at an entry-level of complexity.

- ***R**einforcement level* where students are expected to possess a basic level of knowledge and familiarity with the content or skills at the collegiate or graduate level. Instruction and learning activities concentrate on enhancing and strengthening knowledge, skills, and expanding complexity.
- ***E**mphasis level* where students are expected to possess a strong and advanced foundation in the knowledge, skill, or competency at the collegiate or graduate level. Instructional and learning activities focus on the use of the content or skills in multiple contexts and at multiple levels of complexity.

4) Curriculum Mapping: Curriculum design and development is a process that requires continuous improvement. An efficient tool that is typically used in order to reveal gaps and repetitions in content is the curriculum map. The map is a matrix that identifies on one dimension the courses and on the other dimension the student outcomes. A given cell in the matrix indicates the level of correspondence between the courses and the outcomes.

Curriculum maps can be used in order to check for consistency and coherency in the program by aggregating courses in order to identify where student outcomes are enabled and at what learning level. This fine-grained mapping provides more information on the programÕs weaknesses, strengths, and areas of improvement. It does also provides the process with he optimal courses that can be used for data collection and measurement.

5) Assessment Schedule: Once the curriculum map has been finalized, the assessment coordinator collates all information and sets the schedule of the assessment process based on courses that emphasize the student outcomes. The assessment schedule includes:

(a) The student outcome to be assessed;
(b) The assessment plan for the outcome which states when the course is assessed and through which courses;
(c) The evaluation party.

IV. Assessment

A. Program Educational Objectives Assessment

As of the 2013-2014 accreditation cycle, programs are no longer required to assess the extent to which graduates achieve the *Program Educational Objectives* (PEOs). However, the program is required to instill and systematically utilize an “effective process," that involves the program constituencies for the periodic review of these program educational objectives. The program should ensure that the objectives remain consistent with the institutional mission, the program’s constituents’ needs, and these criteria [6].

The computer science program’s educational objectives at LAU are reviewed based on a three-year cycle that focuses on the alumni who have been in the workforce or graduate school for two to five years in addition to the Advisory Council.

B. Student Outcomes’ Assessment

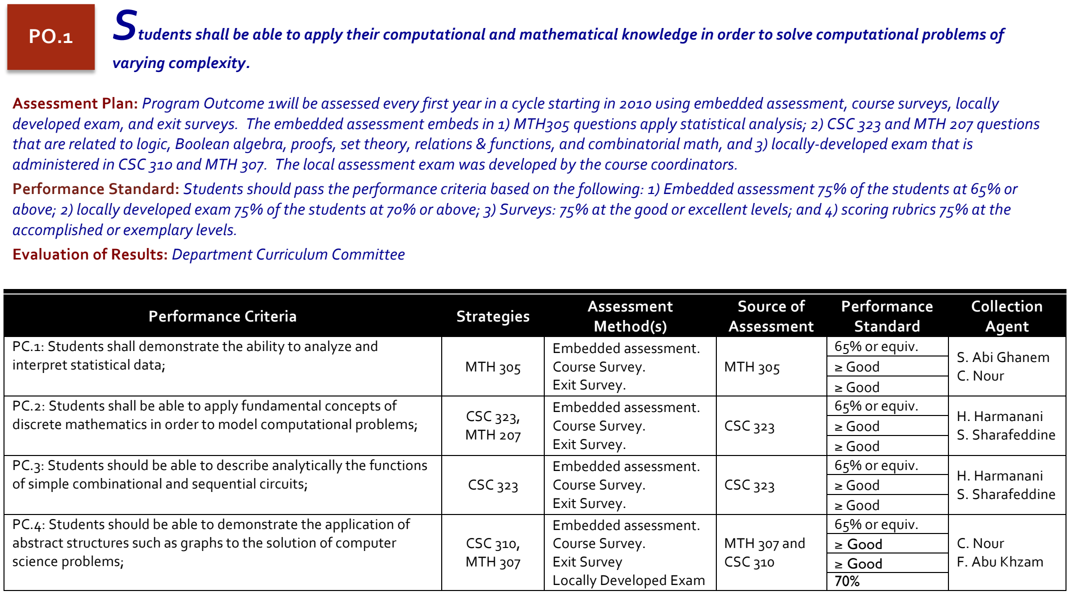
Once the assessment plan has been finalized and the schedule set, the *assessment officer* communicates to all course coordinators the schedule of assessment of their courses. The course coordinators are also communicated the *faculty course assessment form* that is used in order to streamline the assessment process, and which is a modified version of the FCAR that was first introduced by Estell [15] The assessment process measures the performance criteria using the appropriate instruments that were selected by the coordinators. The FCAR includes a *header*, *grades distribution*, *program outcome assessment*, *student feedback*, *reflection*, and *proposed actions for course improvement*. The FCAR makes use of the collected assessment data in order to draw conclusions regarding the students meeting the student outcomes. The effective use of FCAR should lead to a more consistent standard of teaching, learning, and assessment. The FCAR is presented to the assessment officer who shall send an annual report to the departmental curriculum committee. The departmental curriculum committee prioritizes and then sends a recommendation to the faculty.

An outcome is deemed to be met based on the dominant classification of its performance criteria. A performance criterion is classified to be *below*, *minimally meet*, *meet*, or *exceed expectations* based on the following:

1) Whenever scoring rubrics are used, a criterion is said to be met if at least 75% of the assessed students receive a score of 3 or 4 based on the following scoring method:
   (a) Beginning
   (b) Developing
   (c) Accomplished
   (d) Exemplary
2) Whenever embedded assessment is being used, a criterion is said to be met if 75% of the assessed students receive a score of at least 65% or above. Since the level questions tend to be either fairly easy or somewhat difficult.), the final threshold is determined by the course coordinators but should not be in any case less 60%.
3) Whenever a locally developed exam is administered,

Course Outcome:			
Related Student Outcome:			
Assessment Method	*Average Score*	*Performance standard*	*Percent Meeting Performance Standard*

TABLE I
Assessment Part of the FCAR



PO.1 ***Students shall be able to apply their computational and mathematical knowledge in order to solve computational problems of varying complexity.***

Assessment Plan: *Program Outcome 1will be assessed every first year in a cycle starting in 2010 using embedded assessment, course surveys, locally developed exam, and exit surveys. The embedded assessment embeds in 1) MTH305 questions apply statistical analysis; 2) CSC 323 and MTH 207 questions that are related to logic, Boolean algebra, proofs, set theory, relations & functions, and combinatorial math, and 3) locally-developed exam that is administered in CSC 310 and MTH 307. The local assessment exam was developed by the course coordinators.*

Performance Standard: *Students should pass the performance criteria based on the following: 1) Embedded assessment 75% of the students at 65% or above; 2) locally developed exam 75% of the students at 70% or above; 3) Surveys: 75% at the good or excellent levels; and 4) scoring rubrics 75% at the accomplished or exemplary levels.*

Evaluation of Results: *Department Curriculum Committee*

Performance Criteria	Strategies	Assessment Method(s)	Source of Assessment	Performance Standard	Collection Agent
PC.1: Students shall demonstrate the ability to analyze and interpret statistical data;	MTH 305	Embedded assessment. Course Survey. Exit Survey.	MTH 305	65% or equiv. ≥ Good ≥ Good	S. Abi Ghanem C. Nour
PC.2: Students shall be able to apply fundamental concepts of discrete mathematics in order to model computational problems;	CSC 323, MTH 207	Embedded assessment. Course Survey. Exit Survey.	CSC 323	65% or equiv. ≥ Good ≥ Good	H. Harmanani S. Sharafeddine
PC.3: Students should be able to describe analytically the functions of simple combinational and sequential circuits;	CSC 323	Embedded assessment. Course Survey. Exit Survey.	CSC 323	65% or equiv. ≥ Good ≥ Good	H. Harmanani S. Sharafeddine
PC.4: Students should be able to demonstrate the application of abstract structures such as graphs to the solution of computer science problems;	CSC 310, MTH 307	Embedded assessment. Course Survey. Exit Survey Locally Developed Exam	MTH 307 and CSC 310	65% or equiv. ≥ Good ≥ Good 70%	C. Nour F. Abu Khzam

Fig. 3. Sample Assessment Plan for Student Outcome 1

a criterion is said to be met if 75% of the assessed students receive a score of at least 70% or above.

4) Whenever surveys are administered, a criterion is said to be met if 75% of the assessed students are at the good or above levels.

We next map the percentages that were received per performance criteria to *numeric score*, from 1 to 4, as shown in Table II in order to be able to conclude whether an outcome is met or not. The scores are next aggregated based on a simple weighted formula where each performance criteria is assigned a weight, $w_i, i = 1, 2, ..., n$, and an aggregation rule $R(V) = sign \sum_{i=1}^{n} w_i v_i$. Thus, the outcome is deemed to be met or not based on the output of the weighted majority function.

V. Evaluation and Closing the Loop

The last step in the outcome-based assessment is to analyze the collected data and to feed into the program possible identified improvements. The process should be documented and reported for the purpose of ABET accreditation, and is typically driven by a departmental curricular committee. Changes may include the adjustment of course contents, the addition or deletion of courses, and the acquisition of equipments or facilities, to cite few. One of the challenges during this phase is the ability to prioritize the proposed improvements and to find the proper balance between program needs and financial and academic constraints.

Percent Student Meeting Criterion	Classification	Weight
$\leqslant$ 69%	Below Expectations	1
70%—79%	Minimally Meet Expectations	2
80%—89%	Meet Expectations	3
90%—100%	Exceed Expectations	4

TABLE II
Scoring Mapping

VI. Conclusion

We have presented a systematic process for outcome-based assessment that has led to the accreditation of the computer science program at the Lebanese American University. The process is based on a bottom-up approach that minimizes faculty's effort and increases faculty buy-in. The process can be easily automated and efficiently implemented.

References

[1] I. Kendle, “Examinations and their Substitutes in the United States,” The Carnegie Foundation for the Advancement of Teaching, Tech. Rep., 1936.

[2] R. J. Shavelson, *Measuring College Learning Responsibly: Accountability in a New Era*. Stanford University Press, 2010.

[3] G. Kuh and S. Ikenberry, “More Than You Think, Less Than We Need: Learning Outcomes Assessment in American Higher Education,” National Institute for Learning Outcomes Assessment, Tech. Rep., October 2009. [Online]. Available: http://www.learningoutcomeassessment.org/

[4] D. Lederman. (2011, November) Raising the Bar on Quality Assurance. [Online]. Available: http://www.insidehighered.com/

[5] B. Crouch and L. Schwartzman, “Computer Science Accreditation: the Advantages of Being Different,” in *34th SIGCSE technical symposium on Computer Science education*, 2003, pp. 36–40.

[6] ABET Inc., “Criteria For Accrediting Computing Programs.” [Online]. Available: http: //www.abet.org/

[7] C. Cook and P. Mathur and M. Visconti, “Assessment of CAC Self-Study Report,” in *34th ASEE/IEEE Frontiers in Education Conference*, October 2004, pp. 12–17.

[8] D. Blanford and D. Hwang, “Five Easy but Effective Assessment Methods,” in *34th SIGCSE Technical Symposium on Computer Science Education*, 2003, pp. 41–44.

[9] K. Sanders and R. McCartney, "Program Assessment Tools in Computer Science: a Report from the Trenches," in *34th SIGCSE Technical Symposium on Computer Science Education*, 2003, pp. 31–35.

[10] M. Dick, "Student Interviews as a Tool for Assessment and Learning in a Systems Analysis and Design Course," in *ACM Annual Conference on Innovation and Technology in Computer Science Education*, 2005, pp. 24–28.

[11] K. E. Sanders and R. McCartney, "Collected Wisdom: Assessment Tools for Computer Science Programs," *Computer Science Education*, vol. 14, no. 3, pp. 183–203, 2004.

[12] G. Rogers, "IDEAL: Unpublished Notes," 2009, aBET Inc.

[13] B. Maxim, "Closing the Loop: Assessment and Accreditation," *Journal of Computing Sciences in Colleges*, vol. 20, no. 1, pp. 7–18, 2004.

[14] P. Maki, *Assessing for Learning: Building a Sustainable Commitment Across the Institution*. Stylus, 2012.

[15] J. Estell, "The Faculty Course Assessment Report," in *6th Best Assessment Processes Symposium*, 2004.

THREE FIRST-TIME COMPUTER ENGINEERING PROGRAM ACCREDITATION EXPERIENCES

William A. Stapleton
Ingram School of Engineering, Texas State University – San Marcos
601 University Drive, 2240 RFM, San Marcos, TX 78666 USA
wstapleton@txstate.edu, Voice: (512) 245-8746, FAX: (512) 245-7771

The 2013 International Conference on Frontiers in Education: Computer Science and Computer Engineering

ABSTRACT

This paper describes the successful first-time accreditation experiences under the ABET EC2000 criteria with three Computer Engineering programs at The University of South Alabama in 1998, The University of Alabama in 2002, and Texas State University-San Marcos in 2012. The author was directly involved with each of these successful accreditation efforts and offers observations on the common elements underpinning their success.

Keywords: ABET accreditation, program development

1. INTRODUCTION

The recognized accreditation agency for engineering and related baccalaureate programs in the United States of America is the Accreditation Board for Engineering and Technology (ABET) [1]. During the waning years of the 20th century, ABET revised its' accreditation criteria in preparation for the 21st century in order to move toward a culture of continuous quality improvement. This revised set of criteria became known as Engineering Criteria 2000 (EC2000). Even now, a decade and a half later, many programs are still struggling with the reality of the necessity of the assessment and evaluation cycle which makes successful implementation of the ABET criteria possible. Presented herein are observations on the ABET accreditation process for new engineering programs garnered from one faculty member's (reasonably unique) experience with bringing three Computer Engineering programs through the process of initial accreditation.

All three of the Computer Engineering programs described herein were structured from their inception with the consideration of the ABET criteria specific to Computer Engineering [2]. Further, all three programs also used the model Computer Engineering curriculum developed jointly by the Institute of Electrical and Electronics Engineers (IEEE) and Association for Computing Machines (ACM). The IEEE/ACM model curriculum offers several templates for building a Computer Engineering curriculum under a variety of administrative structures [3].

The first of the three programs serving as the basis for this paper's observations on the accreditation process comes from the Department of Electrical and Computer Engineering (ECE) at The University of South Alabama (USA) [4]. Their Computer Engineering program was developed as a jointly-administered program overseen by the Electrical Engineering faculty in the College of Engineering and the Computer Science faculty in the College of Computer and Information Sciences. The program's origins came in the mid-1990s and its' first accreditation visit came in 1998. The engineering programs at The University of South Alabama were the first in the state of Alabama to be accredited under the EC2000 criteria.

The second of the three programs serving as the basis for this paper's observations on the accreditation process comes from the Department of Electrical and Computer Engineering (ECE) at The University of Alabama (UA) [5]. Their Computer Engineering program was developed as a program option within Electrical Engineering with the intention of using that process to eventually spawn a separate Computer Engineering degree. The program's origins came in the late 1990s and its' first accreditation visit came in 2002.

The third of the three programs serving as the basis for this paper's observations on the accreditation process comes from the Ingram School of Engineering at Texas State University-San Marcos (TSU-SM) [4]. Their Computer Engineering program was developed as a program option within Electrical Engineering with independent accreditation. The program's origins came with the creation of the Ingram School of Engineering in 2008 and the first accreditation visit came in 2012. The final ABET accreditation decision for the Computer Engineering program will come during the ABET general session in 2013. Preliminary comments from the accreditation visiting team suggest a high probability of successful accreditation.

Section 2 of this paper briefly discusses the ABET criteria and challenges in assessing them. Sections 3-5 discuss the lessons learned in the process of each of the initial accreditation efforts. Section 6 concludes with recommendations to programs seeking initial accreditation.

2. ABET CRITERIA

The ABET EC2000 Criteria revolve around a set of eight general criteria categories. These criteria are:

1. Students
2. Program Educational Objectives
3. Student Outcomes
4. Continuous Improvement
5. Curriculum
6. Faculty
7. Facilities
8. Institutional Support

Criteria 1, 4, 7, and 8 are generally straightforward for most programs to address. A full description of the student body, especially a demographic breakdown of the students' backgrounds, is generally already compiled by a university's administration. Similarly, information about the faculty and facilities are readily documented. Institutional support should be readily apparent in any university willing to expend the resources to create an engineering program.

Demonstrating the curriculum requirements of Criterion 5 involves documentation of the coursework within the program. All programs require a minimum of 32 semester credit hours of mathematics and basic sciences and a minimum of 48 semester hours of engineering topics. In addition, the program must include a core curriculum appropriate for the institution. Each of the three programs represented here met these requirements. However, each of the programs had to seek permission to exceed a target limit of 120 semester hours. In the cases of USA and UA, the state-required core curriculum totals 36 semester credit hours. The programs totaled 131 and 130 semester hours respectively, which is 95 and 94 hours beyond core curriculum respectively. In the case of TSU-SM, the state-required core curriculum totals 48 semester credit hours. The program totals 136 semester hours, which is 88 hours beyond core. An informal survey of 25 Computer Engineering programs nationwide found an average between 95-96 semester hours beyond core with a minimum of 88 hours beyond core and a maximum of 108 hours beyond core. In the same survey, core curricula ranged from a minimum of 27 semester hours to a maximum of 48 semester hours with an average in the 34-35 hour range. Of the 25 programs considered, only two were able to meet the 120 hour goal and both were arranged with 30 hours of core curriculum and 90 hours beyond core curriculum. Arranging a curriculum to meet topical requirements while fitting within any credit hour limitations can be an ongoing battle in creating a new program.

Dealing with Criteria 2, 3, and 4 almost always forms the bulk of the accreditation efforts of a program on an ongoing basis. Program Educational Objectives

(PEO) are longer-term goals for the educational experience. PEOs are related to the skill sets that graduates should develop in their careers as a result of their educational experiences. Evidence of achievement of the PEOs should be available as soon as 3-5 years into the graduate's engineering career. Direct measurement of achievement of PEOs can be very difficult for new programs which do not yet have graduates who have been in the workforce for 3-5 years. Some direct data can be gathered from the employers of more recent graduates and some indirect measurement can be made early in the post-graduate career. Established programs can more easily gather direct measurements from the employers of graduates who have spent at least 3-5 years in the workplace.

Student Outcomes (SO) are shorter-term goals for students to achieve before they are allowed to graduate. ABET provides the most detailed requirements for the criterion and most programs spend the majority of their effort addressing SOs. ABET specifies the following outcomes as the minimum requirement for SOs for any program [1]:

a) an ability to apply knowledge of mathematics, science, and engineering

b) an ability to design and conduct experiments, as well as to analyze and interpret data

c) an ability to design a system, component, or process to meet desired needs within realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability

d) an ability to function on multidisciplinary teams

e) an ability to identify, formulate, and solve engineering problems

f) an understanding of professional and ethical responsibility

g) an ability to communicate effectively

h) the broad education necessary to understand the impact of engineering solutions in a global, economic, environmental, and societal context

i) a recognition of the need for, and an ability to engage in life-long learning

j) a knowledge of contemporary issues

k) an ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.

The key for any program, especially new programs, is to demonstrate that all of the SOs are being achieved. In order to do so, the program must show that achievement is being actively and systematically assessed and that the results of the assessment are being analyzed and utilized to further improve future achievement of the SOs. In an ideal situation, achievement of the SOs by students prior to graduation should support future achievement of the PEOs by graduates.

Continuous Improvement, as its own topic, and in relation to both PEOs and SOs, is one of the largest differences between EC2000 and its' predecessor curricula. There is a much greater emphasis on *process*. It is very important for a new engineering program to develop a strong process which will continuously *assess* its progress, *analyze* the data from the assessment process, and *apply* the results of the analysis to improve and strengthen the program. This can be illustrated with the three exemplar programs in the next sections.

3. THE UNIVERSITY OF SOUTH ALABAMA

The Computer Engineering program at The University of South Alabama was first accredited in the 1998-1999 cycle. The program was developed during the mid-1990s and was the first Computer Engineering program accredited under EC2000 in Alabama. Like most programs, this program chose classical direct assessment of SOs a, b, c, e, and k based on performance in coursework. These SOs are the traditional "hard skill" technical subjects. SOs d, f, g, h, i, j, and k are the "soft skill" subjects. Most new programs have to make a special effort to address these soft skills. The program at USA addressed these by adding a course entitled "Professionalism and Ethics in Electrical and Computer Engineering" into the curriculum which contained lessons and assignments specifically tailored to address each of the soft skills. For example, to address SO f, “an understanding of professional and ethical responsibility,” the class had lessons on the IEEE Code of Ethics as well as other professional codes of ethics and had an assignment to create their own code of ethics with requirements to justify why each element of their code was the same as or different from the codes studied.

The assessment efforts at USA took a very systematic approach to utilizing both direct and indirect assessment with an eye to cross-correlating the two [7,8,9]. A subset of the SOs was assessed in each course in the curriculum taught in either the College of Engineering or the College of Computer and Information Sciences. Care was taken so that each SO was assessed in at least three different courses for any program track. Further, each topic to be taught in each course was assigned to one of the SOs. For each topic three direct and three indirect assessments were made. The typical three direct assessments would be a quiz given on a topic after a reading assignment but before lectures on the material, a quiz or homework assignment after an initial lecture on the material, and a test question after the students have had time to practice the material. The three indirect assessments typically took the form of three surveys of the students taken at the same time as the direct assessments. A committee of the faculty met twice per semester to analyze the results of the assessments and propose responses to the fruits of the analysis.

During the ABET accreditation visit, the process at USA was scrutinized by the visiting team. Praise was offered for providing *evidence* of a process which systematically assesses all of the SOs and utilizes the results of thorough analysis. The praise was not untempered, however. The visiting team suggested that the process was even more complex than was necessarily required and expressed an opinion that it would be easy for faculty to find six assessments of each topic in each class to become burdensome over time. After accreditation was awarded, the faculty chose to reduce the number of topics assessed in each course. The basic process otherwise remained the same and has continued to be used. Accreditation has been maintained continuously since the initial accreditation.

4. THE UNIVERSITY OF ALABAMA

The Computer Engineering program at The University of Alabama was first accredited in the 2001-2002 cycle. The program was developed during the late 1990s as a separately accredited option track within Electrical Engineering. Unlike the USA program, the UA program did not create a separate “soft skills” course. Rather, the choice was made to incorporate modules on these skills spread throughout all technical courses.

The assessment process at UA revolved around the technique of “embedded assessment” [10,11,12,13]. For this process, the SOs which would be assessed in each course were chosen such that there would be multiple assessments of each SO across the program but that no single course would be required to assess less than two nor more than four SOs during any given semester. The faculty as a whole meets at least once per year in a retreat to review the entire curriculum, especially the analysis of the assessment data, in order to update and optimize the program.

During the ABET accreditation visit, the process at UA was scrutinized by the visiting team. Praise was offered for providing *evidence* of a process which systematically assesses all of the SOs and utilizes the results of thorough analysis. The only concern raised involved the feedback from the program constituencies. The ABET visiting team noted that constituencies were well defined and evidence had been provided that they were involved with the creation of the programmatic plan. However, the annual faculty review process did not incorporate ongoing feedback from the external constituencies. After accreditation was awarded, the faculty began to build an advisory committee composed of representatives of the external constituencies. Over the course of several years, this advisory committee grew to approximately two dozen members. The advisory committee is convened and consulted annually to review the analysis of the assessment data and recommend updates to the program. The recommendation of this advisory committee is taken into account by the faculty at their annual conclave. Accreditation has been maintained continuously since the initial accreditation.

5. TEXAS STATE UNIVERSITY-SAN MARCOS

The Ingram School of Engineering at Texas State University-San Marcos was founded in 2008 with three programs, one of which was Electrical Engineering [14]. Three of the faculty members organizing the program had former ties to the UA program and tried to incorporate the best lessons from that program. In 2009, the faculty decided to organize Computer Engineering as a program option within Electrical Engineering. When initial accreditation was sought during the 2012-2013 cycle, the decision was made to seek separate accreditation for Computer Engineering as if it were a stand-alone program.

The TSU-SM program also chose to use the embedded assessment methodology. The assessment of SOs was distributed among the courses in order to ensure multiple assessments of each SO without overburdening any particular class. The soft skill modules were distributed throughout the curriculum. The assessment process ensures that a minimum of two assessments of each SO are taken from each student in each classes in which they are assessed.

The faculty as a whole meets at least once per semester to review the entire curriculum, especially the analysis of the assessment data, in order to update and optimize the program. The faculty also organized an Industrial Advisory Board (IAB) to provide external constituency feedback. The IAB meets annually.

During the ABET accreditation visit, the process at TSU-SM was scrutinized by the visiting team. While the accreditation process is not complete as of this writing, the initial feedback from the ABET visitors makes accreditation seem likely. The final decision will be made during the ABET annual meeting during the summer of 2013. The accreditor’s comments indicate two areas of potential concern. The first potential concern involves sufficient evidence that all of the realistic design constraints are considered in every student’s Capstone Design. While the accreditors indicate that there is evidence that the class as a whole is considering these, it is not absolutely clear that every student considers every constraint. The faculty have already met and considered methods for assuring assessment of every constraint for every project. These assessment assurances were implemented in the Fall 2012 semester and will be implemented in each subsequent semester.

6. CONCLUSIONS

The three Computer Engineering programs at USA, UA, and TSU-SM described herein which went for recent first-time ABET accreditation all shared certain successful traits and each offered a different opportunity to improve future efforts.

When first developing a program, it is critical to choose the coursework carefully to accommodate the 32 minimum hours of mathematics and basic sciences, the 48 minimum hours of engineering topics, and the locally required core curriculum. Balancing these minimum requirements with any credit hour limits, such as the increasingly common 120 hour limits in many locations, can be a daunting task. This author has not found an accredited program requiring less than 88 semester hours beyond any required core curriculum.

A key component of the EC2000 accreditation standards is ensuring a process of continuing assessment and improvement. It is important for a new program to remember not only that they must develop a set of desired student outcomes, but that they must a process to continuously assess their success in meeting these outcomes, to analyze the results of the assessments, and to adapt based on the results of the analysis. Carefully selecting where within the curriculum to assess each of the SOs will allow the assessment burden to be limited for any given class while still providing comprehensive assessment of all of the SOs.

It is imperative as programs prepare for accreditation, that they provide documentation showing concrete examples of how the process has been used and how it will be sustainable to continue the process into the future. Remember also that it is important to include feedback from all of the program's constituencies in an ongoing cycle of assessment and adjustment.

7. REFERENCES

[1] WWW Link: http://www.abet.org

[2] WWW Link: http://abet.org/DisplayTemplates/ DocsHandbook.aspx?id=3149

[3] WWW Link: http://www.acm.org/education/ curricula-recommendations

[4] WWW Link: http://www.southalabama.edu/ engineering/ece/index.html

[5] WWW Link: http://www.ua.edu/majors/ electricalandcomputerengineering.html

[6] WWW Link: http://www.engineering.txstate.edu

[7] Parker, M. R., W. A. Stapleton, and T. Thomas, "Assessment Processes for the Evaluation of Engineering Instruction Quality," Proceedings of IEEE Southeastcon 2003, Ocho Rios, Jamaica, April 4-6, 2003.

[8] Parker, M. R., W. A. Stapleton, and T. Thomas. "Post-Assessment Evaluation of Engineering Programs," Proceedings of Engineering Foundation Conference on Engineering Education for Global Practice in the 21st Century, Barga, Italy, April 2000.

[9] Parker, Martin R., and William A. Stapleton. "ABET EC 2000: Quantifying the Assessment of Outcomes," Proceedings of the Annual Southeastern Section of the ASEE, Clemson University. 139-146, April 1998.

[10] Stapleton, William A., "Embedded Assessment of Microcomputer Fundamentals for Embedded Systems Education", The 2008 International Conference on Frontiers in Education: Computer Science and Computer Engineering, Las Vegas, NV, July 13-17, 2008.

[11] Stapleton, William A., "Microcomputer Fundamentals for Embedded Systems Education," 36th Annual Frontiers in Education Conference (FIE 2006), San Diego, CA, October 28-31, 2006.

[12] Ricks, Kenneth G., David J Jackson, William A. Stapleton, "Addressing Embedded Programming Needs within an ECE Curriculum", 2006 Workshop on Embedded Systems Education (WESE 2006) within the ACM Conference on Embedded Systems Software, Seoul, South Korea, October 22-25, 2006.

[13] Ricks, K. G., Stapleton, W. A., Jackson, D. J., "A Focused Curriculum for Embedded Systems", Proceedings of the 32nd Annual International Symposium on Computer Architecture (ISCA), Madison, Wisconsin, June 2005.

[14] Stapleton, William, "Zero To Two Hundred In Two Years: Launching A New Program," The 2010 International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS'10), pp. 104-108, July 2010. (Acceptance rate: 26%)

Improving Academic Quality in a Computer Science Graduate Program

Meline Kevorkian#1, Gregory Simco#2

# Office of Academic Affairs, Nova Southeastern University
3301 College Avenue, Fort Lauderdale, FL 33314, USA
melinek@nova.edu

# Graduate School of Computer and Information Sciences, Nova Southeastern University
3301 College Avenue, Fort Lauderdale, FL 33314, USA
greg@nova.edu

The 2013 World Congress in Computer Science,
Computer Engineering, and Applied Computing

Abstract*—Building quality programs based on academic metrics are considered an effective means for recruitment and retention in higher education, especially in the areas of science, technology, and math. The aim of this paper is to share the academic review process of Nova Southeastern University and the results of the collegial, peer, and external review to continually build the quality of the Master of Science in Computer Science.

Keywords: Computer Science, Academic Review, and Assessment

I. Introduction

Although most institutions of higher education are collecting evidence of student learning, it is not clear how results are being used for improvement (Baker, Jankowski, Provezis, 2012). Some institutions measure education program quality strictly on the publications of faculty (Laender, Lucena, Maldonado, Siva & Ziviani, 2008). According to the New Leadership Alliance's Committing to Quality Guidelines for Assessment and Accountability in Higher Education (2012), the following are four principles for effective assessment practice: set ambitious goals, gather evidence of student learning, use evidence to improve student learning ,and report evidence and results. Borrego and Cutler (2010) recommend enlisting assessment/evaluation assistance when evaluating and improving the curriculum. At Nova Southeastern University, Academic Program Review consists of on-going, high quality peer reviews of all the University's academic units and programs on a five-year cycle. This paper reports on this academic review process; this is followed by the results occurring through the review the M.S. in Computer Science. Wendler, Bridgeman, Cline, Millet, Rock, Bell & McAllister (2010) suggest that a strong system of graduate education is the basis for global competiveness. Additionally, they discuss how graduate students and academic expectations are changing.

II. Academic Program Review

The continuous assessment of academic learning in order to stimulate action for improvement stands as one of the most challenging and necessary undertakings for higher education academic leaders. In response to this need, Nova Southeastern University created an Academic Program Review system, which facilitates the university's processes for internal reflection and program growth. The Academic Program Review allows for an outcome-based review of the program's progress on the road to academic excellence and preeminence by defining and enumerating academic improvement metrics in the areas of curriculum, faculty, students, student services, and living their mission. Academic review involves a six-step process that is evaluative and prescriptive leading to plans for building the quality of the program, students, curriculum, and resources.

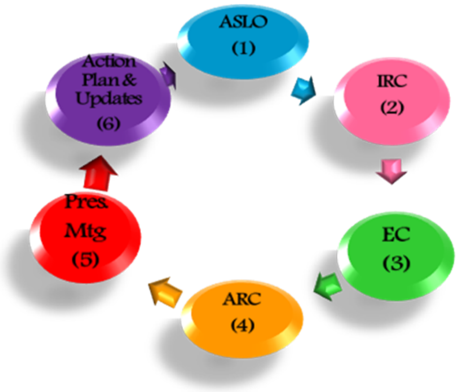


Fig. 1. Academic Program Review Cycle

The six-step evaluation process includes:

1. Academic Information Collection & The Assessment of Student Learning Outcomes (ASLO);
2. An Internal Review Committee (IRC) Evaluation and Report;
3. An External Consultant's (EC) Evaluation and Report;

4. A Summary of Evaluations and Recommendations from the Academic Review Committee (ARC);
5. The program's response to the Academic Review Committee's summary, evaluation, and recommendations; and
6. An action plan developed by the dean and faculty and accepted by NSU's president and provost. The action plans, once accepted, are incorporated into each academic unit's long and short-term planning process.

A. Assessing learning

Working with data collected for the M.S. in Computer Science, the faculty measured their programs against the standards of the most prestigious programs in the areas of curriculum, faculty, students, educational support services, and mission and vision. Computer Science faculty, faculty from the university, and an external consultant worked together to evaluate the program and outline necessary elements of preeminence. The process began with a formal reporting of the assessment of student learning outcomes. This paper reports on the program outcomes and includes a discussion of the rationale for which outcomes are being analyzed. The direct assessment method is described in the report, including how data is collected and analyzed. Faculty are asked to clearly show the linkages between the expected content of student products and performances, assessment instruments (such as rubric items), and the program outcome under study. An outside reader must be able to see the relationship between the direct assessment, the assessment instrument, and the program student-learning outcome. An analysis, interpretation, and discussion of result is compared to the expected level of achievement of the program-learning outcome. If applicable, a comparative analysis of student achievement for each type of location and/or modality is provided with a summary of achievements, strengths, and weaknesses in student learning. Faculty are asked to reflect upon the assessment results, outline strategies for improving student learning, and discusses implementation methods.

B. Internal Review Committee

Reviews are forward-looking and the Internal Review Committee Report is expansive. First, the Report identifies preeminent programs and highlights what qualities define "preeminence". Next, the Report summarizes the information that will be presented and positions the program in relation to the University's mission statement and overall goals. The most substantive part of the Report focuses on specific and in-depth analyses of the program.

The Assessment of Quality section of the IRC covers five general areas: Faculty Development, Student Enhancement, Curriculum Development, Student Services, and the Program's Fit with the University. Under each of these general headings, several areas are analyzed. For example, the Faculty Development section analyzes whether the faculty are sufficient in number and quality. Student data includes enrollment, aptitude to perform in the program, and participation in faculty-student research.

C. External Consultant's Evaluation

External Consultants (EC) are experts in the field of the program to be reviewed, and accepted as leaders in their area of expertise. For each program to be reviewed, a designated EC will provide to the Provost a written evaluation of the program and a critique of the IRC report. Subsequent to the conclusion of site visit, the EC will provide a written assessment that comments on the validity and accuracy of the internal review committee report.

D. Academic Review Committee

The Academic Review Committee (ARC) is a permanent, standing university committee appointed by and reporting to the President. The main role of members is to serve as chairpersons for the Developing Academic Review Summaries that succinctly present the strengths and weaknesses of each program. Subsequent to the conclusion of an External Consultant's visit, the consultant provides a written assessment commenting on the validity and accuracy of the internal review committee report. The success of an academic program reviews depends on timely follow-through by the Dean, ARC, and the President's Office, with primary emphasis on how the academic unit can be improved through the effective use or reallocation of its existing resources.

III. Academic Review Results

The academic review results of the M.S. in Computer Science indicated particular attention was needed in the curriculum area. Although regular faculty review of the curriculum occurred, additional attention was needed to ensure the depth and flexibility in course selection for students. The EC found the curriculum to provide the breadth of the computer science field but at the expense of flexibility. In our experience, the curriculum revisions that enhanced flexibility of the program translated into an increased enrollment. Overall enrollment in our M.S. in Computer Science increased 43% from the fall of 2011 to the fall of 2012. New starts from academic year 2011-2012 to academic year 2012-2013 increased 133%. To this end, faculty should be discussing ways to add flexibility to their computer science curriculum to better address the needs of the workforce and attract more students into critical areas such as science, engineering, math, and technology. By designing a curriculum that is relevant we can engender interest in the field of computer science.

IV. Actions for Improving Academic Quality

As part of the follow through on the M.S. in Computer Science, the resulting action plan charged a faculty committee to review the program's curriculum with an eye to creating a more flexible program with multiple tracks. The committee created six concentrations in the M.S. of Computer Science program: Theory, Software Engineering, Computer

Systems, Database, Security, and Real-World Computing that began in fall of 2012. These six concentrations are designed to provide students with a deeper understanding of computer science theory and application in the areas of algorithms, computer systems, and software development. These concentrations are intended to supplement and enhance student study in computer science.

V. Implications of the Findings

When thinking about assessment and improving quality that sets ambitious goals and a process that uses comprehensive program review process, higher education is called to gather data to improve student learning and improve the quality of its academic programs. Clearly, further work is needed to improve the quality of our academic programs and to optimize the learning in our graduate schools, especially in the areas of science, technology, engineering, and math. Perhaps the most significant aspect of academic review is charging faculty to look at regional and professional accreditation standards to improve quality. This suggests the need for formal documentation of curriculum review and a formalized process that allows opportunity to reflect on the program as a whole.

References

[1] Baker, G. R., Jankowski, N. A., Provezis, S., & Kinzie, J. (2012). Using assessment results: Promising practices of institutions that do it well. *National Institute for Learning Outcomes Assessment (NILOA)*, 2-19.

[2] Borrego, M., & Cutler, S. (2010). Constructive alignment of interdisciplinary graduate curriculum in engineering and science: An analysis of successful IGERT proposals. *Journal of Engineering Education*, 355-369.

[3] Committing to quality: Guidelines for assessment and accountability. (2012). In *New Leadership Alliance for Student Learning and Accountability* (pp. 1-13). Washington, DC: New Leadership Alliance for Student Learning and Accountability.

[4] Wendler, C., Bridgeman, B., Cline, F., Millet, C., Rock, J., Bell, N., & McAllister, P. (2010). *The Path Forward: The Future of Graduate Education in the United States.* Princeton, NJ: Educational Testing Service.

Student Assessment of Student Work

Donald R. Schwartz

Department of Computer Science
Millsaps College
Jackson, MS USA

School of Computer Science and Mathematics
Marist College
Poughkeepsie, NY USA

Abstract - *One of our goals is to graduate students who possess the skill sets necessary for success in both the workforce and in their graduate studies. Assessment plays a significant rolet in this – students who have learned how to assess not only the work of others, but their own work, have an advantage over those who must acquire this skill after graduation. This paper describes some of the ways we have incorporated assessment skills throughout our curriculum. It also includes samples of actual student self-assessment and student peer-assessment exercises.*

Keywords: *Assessment of Student Work, Student Self-Assessment, Collaborative Learning, Factors that lead to success in CS and CE*

1 Introduction and Philosophy

There are a wide variety of studies that describe the benefits of student self-assessment.[1][2] While self-assessment can take many forms, the over-riding goals are quite similar: positive effects on student performance, motivation, engagement and learning.[3][4][5]

One of our goals is to graduate students who possess the skill sets necessary for success in both the workforce and in their graduate studies. Assessment plays a big part in this – students who have learned how to assess not only the work of others, but their own work, have an advantage over those who must pick up this skill after graduation. This paper describes some of the ways we have incorporated assessment skills throughout our curriculum.

2 Off to a Good Start: Assessment in CS1/CS2

Our students begin the process of assessment very early on in their studies. Our CS2 course begins with a rather quick review of CS1, which allows for a refresher in the programming language itself. After this, we'll start with some of the simple sorting algorithms (bubble/ selection/ insertion), at least one of which was already covered in CS1. Students work in small groups throughout the semester. The benefits of Groupwork are well established. [6][7][8][9] Different groups will "tackle" the different sorts, with the goal being that each member of the group must be able to explain the algorithm, discuss the code, and then explain how the implementation differs from the algorithm, if at all. This exercise of comparing the implementation to the original algorithm gives the students a "taste" of one type of assessment – determining whether the implementation actually follows the algorithm. Identifying differences introduces the concept of assessing code – evaluating its structure, efficiency and efficacy.

Beginning in CS2, students begin to learn how to assess other students' code. Since students are almost always working in groups, my usual approach to a typical class session is to introduce the new concept to be explored, go over a few examples, then have the groups work on defining/solving the problem. For example, I'll introduce the concept of a LinkedList and go through some examples of using methods defined for the LinkedList class, such as printEachNode, insert and delete. I'll have each group then come up with the headers/interface for each method. We'll share those ideas and come up with a common set of headers. Each group will then work on, for example, the insert method. Once the groups are finished, each will put their solution on one of the boards. Then, each group rotates to another board and evaluates that group's code. I remind the students that they are NOT trying to determine whether that group's solution is identical to their own; rather, they are trying to determine whether that group's code "works" – whether it conforms to our goal of FREE code. In my classes, FREE stands for **F**unctional (solves the problem), **R**eliable (will not crash, even on special cases), **E**fficient (has a reasonable Big-Oh) and **E**legant (no repeated code, is as concise as possible – a goal which is often more difficult than the others to achieve). The groups evaluate the code, trace through it on all special cases, and make a determination. If errors/issues are identified, the group marks the code in question, but does not correct the code. After each group finishes, and as time allows, the groups rotate to the next board and repeat the process. Eventually, each group returns to its original code and reacts to the feedback they received. If errors exist, the group corrects its code. If they dispute the feedback, they provide evidence to counter the feedback. After this, I'll usually pick one or two groups' code samples and discuss the "elegance" of the code, providing help in making their code as elegant as possible. Then we move on to the delete method and repeat. This same pattern holds for almost all topics (linked lists, stacks, queues, trees and graphs, etc.).

As students assess the code written by other groups, they get very good insight into the reality that there are almost always multiple ways to implement an algorithm. Tracing through code that takes a different approach is beneficial, especially in terms of generating useful test cases. The process of assessing the code's correctness and/or identifying the errors that might exist helps students learn to evaluate their own code much more carefully, especially when they know many other groups will soon be doing the same thing. Updating their own code based on feedback from their peers is always a useful exercise.

Group work is not limited, of course, to generating code. As the semester progresses and their skill levels rise, groups are then tasked with generating algorithms/approaches for new problems. Take, for instance, the Minimum Spanning Tree problem. I'll begin the class by talking about the "internet boom" period of the late 90's/early 00's. I'll show them a graph and explain to them that the nodes are the computers in the network, while the edges represent the connections, but then explain that the weights of the edges represent the price of that connection, since back in the boom times the company bought coaxial cable with wires made of gold instead of copper. (I tell them that, for a while, there seemed to be an infinite influx of money generated by selling stock in the company, even before the company produced any products whatsoever.) Now, our pretend company is experiencing a downturn in revenue, and will need to sell off as much gold as possible, while maintaining a connected network. Each group needs to determine how much money we can generate by selling the most cable.

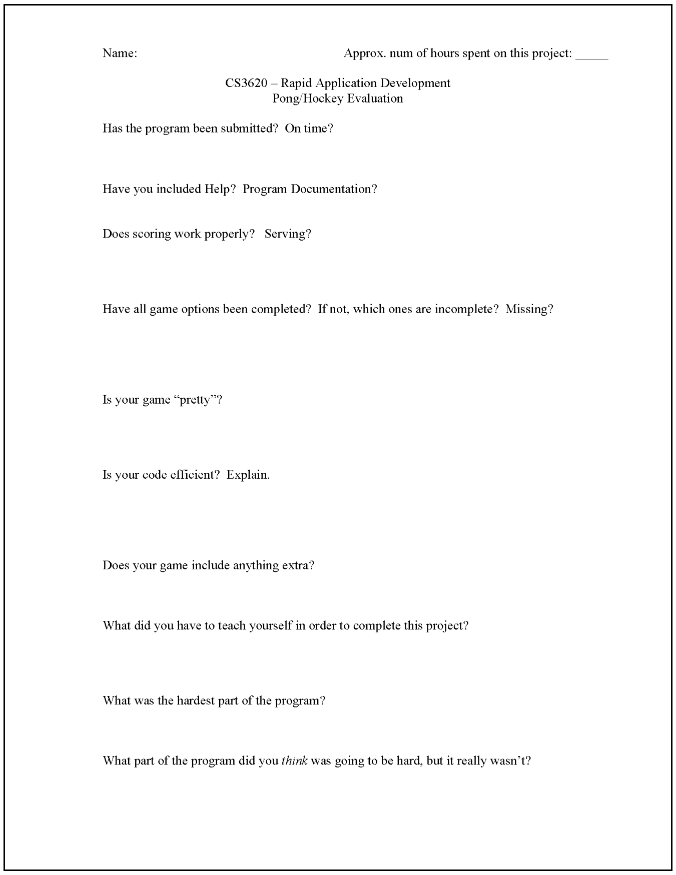
As the groups announce their findings, the challenge becomes trying to find a way to save even more money. Eventually, I'll let them know when we've reached the best answer. (As the groups work, I'll give them a hint about approaching the problem with the focus being "How can we minimize the cost of the cables we keep, rather than maximizing the cost of the cables we sell?".) Once we determine the "best" answer, each group must develop an algorithm that other groups can use to generate a correct answer. After students test their algorithms on the original graph, I'll give them more "carefully selected" sample graphs to use. Assessing the correctness and straightforwardness of their own algorithms, before any code is written, introduces students to a new level of assessment. Going through the exercise of strictly following another group's algorithm not only exposes students to multiple solutions, it also gives students the opportunity to assess the clarity of each algorithm. Each group must provide feedback about the algorithms, including suggestions for making the descriptions as clear and concise as possible. Exercises of this type help to demonstrate that "computer science" is much, much more than "writing and debugging programs". They also stress the importance of developing and testing algorithms well before any code is written. (Happily, every time I've given this exercise, at least one group comes up with a correct algorithm. Then I'll tell the students that sometimes, timing is everything. They'd likely be very famous, but it turns out that Kruskal (or Prim) came up with that same algorithm ages ago – but the fact that they're doing "similar" work while still in CS2 is wonderful!)

3 Adding to the Skill Set: Assessment in Intermediate and Upper-Level Courses

In our Algorithms class (an intermediate-level course), students move beyond assessing their own code and the code of their peers, and start assessing code from the textbooks we use. In this class, I still use an old version of a rather well-known book about advanced algorithms and data structures in C++, mainly because the author does a very good job of explaining the algorithms themselves, but also uses programming "tricks" and non-structured code (infinite for-loops containing a break statement within an embedded if-else, for instance) in some of his implementations. These examples serve multiple purposes: they expose students to non-structured code, which they'll likely encounter in the "real world"; they expose students to some of the "features" available in the implementation language, which wouldn't necessarily be covered in their other classes; and they allow the professor to utter the phrase "Just because you *can* doesn't mean you *should*!!" over and over again. (They also provide excellent examples of code that violates many of the practices we stress in our Software Engineering course.) As we study each example in class, I require that students identify the areas of code that are so peculiar that they "make me vomit a little in the back of my mouth" and then rewrite the code so that it conforms to best practices. (Since it is highly unlikely that we'll meet the actual author during the semester, we sometimes find ourselves wondering "What in the world was he thinking???", which helps to make 75 minutes of evaluating code seem to go more quickly.) Modifying these code examples so that they adhere to good structure-programming practices becomes so routine that I find the students soon start looking at all code with a more critical eye – a wonderful assessment skill to master in the early stages of their education.

In advanced classes, students assess not only other students' projects, but their own work, too. My Rapid Application Development course is basically a user-interface design course which uses many Atari-style 2-D game programming assignments (Pong, PacMan, MineSweeper, etc.) to exercise the interface design theories covered in class. Students complete several projects throughout the semester. On the due date for each project, students begin the class period by completing an assessment form about their own implementation (see Figure 1).

The first few questions (How much time did you spend on this project? Did you submit the program? Was it well-documented? Did you include Help?) serve as basic reminders for the student. The remainder of the self-assessment form consists of many very open-ended questions so as not to limit the responses. For example, “Does your game work properly?” and “Have all game options been completed?” invite the student to assess the completeness of the project. In completing the form, they’ll often discover areas that are either missing or don’t necessarily work the way they intended. I remind the students to make these areas explicitly clear in their assessment so that I’ll know to grade those areas *last*, so that I can hopefully wait to test the areas that might crash.



Name: Approx. num of hours spent on this project: _____

CS3620 – Rapid Application Development
Pong/Hockey Evaluation

Has the program been submitted? On time?

Have you included Help? Program Documentation?

Does scoring work properly? Serving?

Have all game options been completed? If not, which ones are incomplete? Missing?

Is your game “pretty”?

Is your code efficient? Explain.

Does your game include anything extra?

What did you have to teach yourself in order to complete this project?

What was the hardest part of the program?

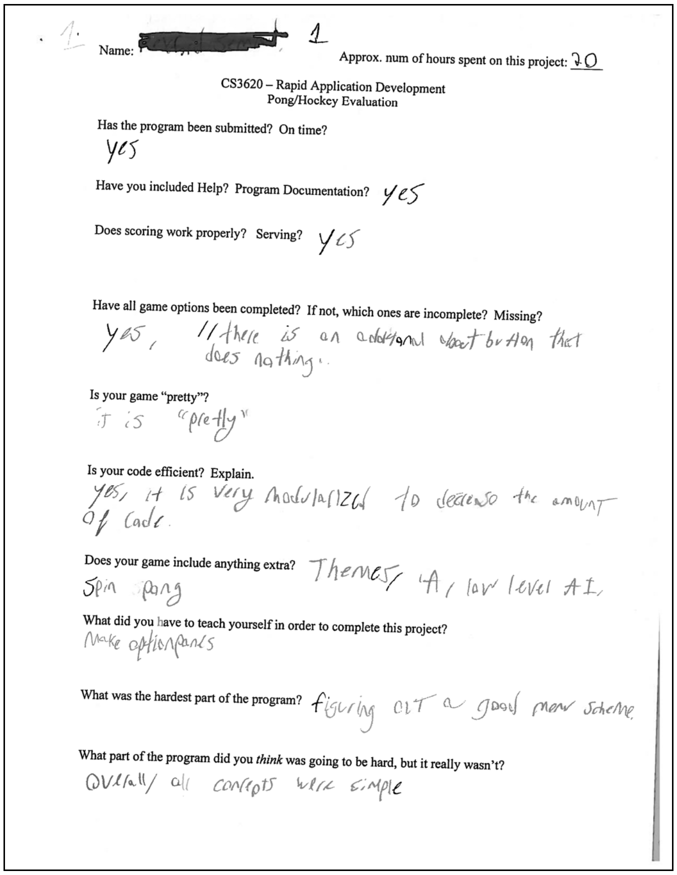
What part of the program did you *think* was going to be hard, but it really wasn’t?

Figure 1: Student Self-Assessment Form

The next few questions (Is your game “pretty”? Is your code efficient? Does your game include anything extra?) are truly “wide open” and allow the student to assess the user interface and aesthetics of the project. They also invite the student to assess the quality of the code itself and to explicitly describe any area(s) in which they may have gone beyond the requirements of the project. (I have found that, especially in projects that involve writing games, some students get very excited and inspired to take their projects well beyond the initial requirements. I want the students to know that I recognize when they do so. I call it the “wow factor” and invite them to make me say *Wow!* when I grade their work.)

I also ask about the process of creating the project and have the students describe the areas that they had to teach themselves in order to complete the project. (We require that *all* upper level assignments contain carefully-chosen components that are *not* covered during class, so that students become accustomed to researching and finding solutions to problems on their own.) The last two questions (What was the hardest part of the program? What part of the program did you think was going to be hard, but it really wasn’t?) allow the student to provide feedback about difficulties (and perceived difficulties) they encountered (or were afraid they might encounter) during the project.

This self-assessment exercise allows the students to provide honest assessments of their own work, which can be a very challenging skill to master. Students complete self-evaluations on every project, so they grow more accustomed to assessing their own work as the semester progresses. Figure 2 shows a completed self-evaluation form. (The Pong/Hockey project was the first project of the semester.)



1. Name: [redacted] 1 Approx. num of hours spent on this project: 20

CS3620 – Rapid Application Development
Pong/Hockey Evaluation

Has the program been submitted? On time?
yes

Have you included Help? Program Documentation? yes

Does scoring work properly? Serving? yes

Have all game options been completed? If not, which ones are incomplete? Missing?
yes, //there is an additional about button that does nothing..

Is your game “pretty”?
it is “pretty”

Is your code efficient? Explain.
yes, it is very modularized to decrease the amount of code.

Does your game include anything extra? Themes, 'A, low level AI, Spin pong

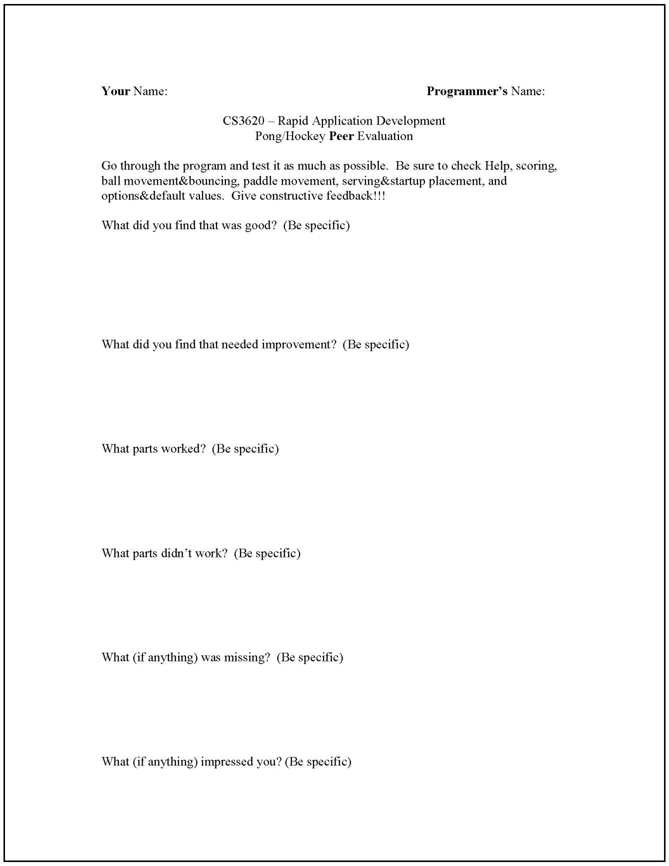
What did you have to teach yourself in order to complete this project?
Make optionpanels

What was the hardest part of the program? figuring out a good menu scheme.

What part of the program did you *think* was going to be hard, but it really wasn’t?
Overall, all concepts were simple

Figure 2 - Sample Self Evaluation

After completing their self-assessments, students receive forms for evaluating the projects of their peers (see Figure 3). The first question (What did you find that was good?) reminds students that “assessment” does not just mean “find out what’s wrong”. Explicitly describing the “good” parts helps to demonstrate how complete their assessment process has been. Along with the second question (What did you find that needed improvement?), together they invite responses about the overall project. The next two questions (What parts worked? What parts didn't work?) expand on the first two questions and invite specific responses and feedback. The next question (What was missing?) allows for an assessment of the completeness of the implementation. The final question (What impressed you?) allows the student to describe his or her own "Wow!" moments while running the program.



Your Name: **Programmer's** Name:

CS3620 – Rapid Application Development
Pong/Hockey **Peer** Evaluation

Go through the program and test it as much as possible. Be sure to check Help, scoring, ball movement&bouncing, paddle movement, serving&startup placement, and options&default values. Give constructive feedback!!!

What did you find that was good? (Be specific)

What did you find that needed improvement? (Be specific)

What parts worked? (Be specific)

What parts didn't work? (Be specific)

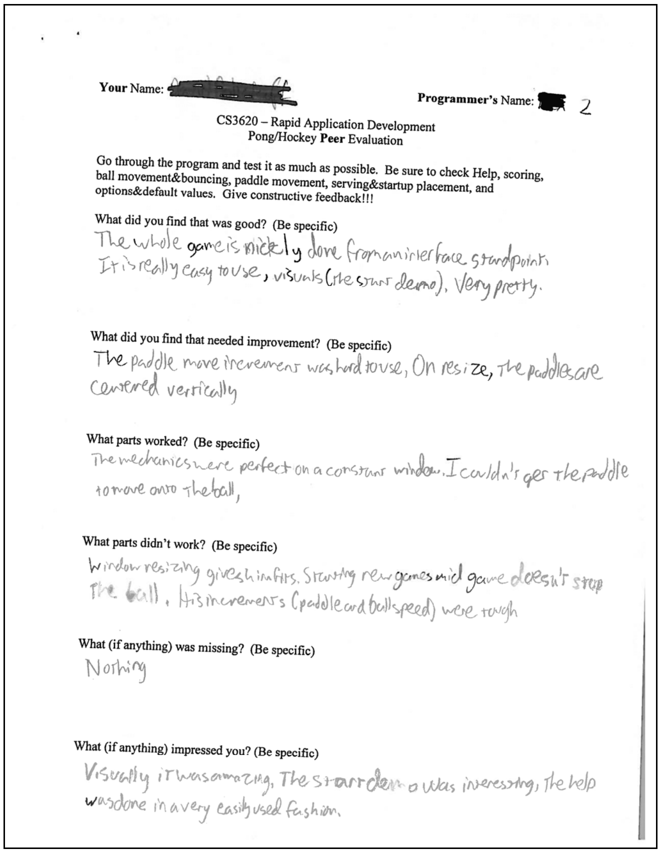
What (if anything) was missing? (Be specific)

What (if anything) impressed you? (Be specific)

Figure 3: Student Peer Assessment Form

Students will run other students' projects and assess their implementations. (Students do not assess the code of their peers, just the implementations.) They'll play several other students' games (pong, pacman, etc.), and complete the feedback sheets with their assessments. Figures 4, 5 and 6 show completed Peer Evaluations for the project developed by the student whose Self Evaluation was shown in Figure 2.

To ensure that this is an assessment exercise and not a play-a-game-for-a-while exercise, I'll remind students to provide specific feedback – not to just put "sometimes the ball bounced strangely" but rather "when the ball is moving down and hits the left side of the top paddle, it sometimes bounces strangely". I also remind them that I'll grade their assessments along with their projects. (I use the same set of questions for many different types of projects – not just for games. The same ideas hold for a variety of implementations!)



Your Name: [redacted]
Programmer's Name: [redacted] 2

CS3620 – Rapid Application Development
Pong/Hockey **Peer** Evaluation

Go through the program and test it as much as possible. Be sure to check Help, scoring, ball movement&bouncing, paddle movement, serving&startup placement, and options&default values. Give constructive feedback!!!

What did you find that was good? (Be specific)

The whole game is nickly done from an interface standpoint. It is really easy to use, visuals (the start demo), Very pretty.

What did you find that needed improvement? (Be specific)

The paddle move increment was hard to use, On resize, the paddles are centered vertically

What parts worked? (Be specific)

The mechanics were perfect on a constant window. I couldn't get the paddle to move onto the ball,

What parts didn't work? (Be specific)

Window resizing gives him fits. Starting new games mid game doesn't stop the ball. His increments (paddle and ball speed) were tough

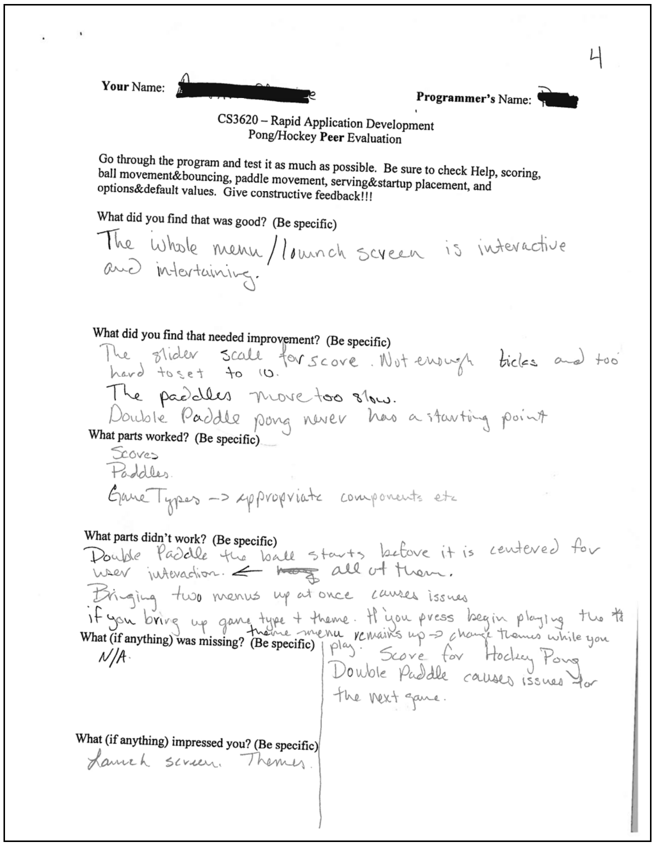
What (if anything) was missing? (Be specific)

Nothing

What (if anything) impressed you? (Be specific)

Visually it was amazing, The start demo was interesting, The help was done in a very easily used fashion.

Figure 4 - Peer Evaluation #1



4

Your Name: [redacted]
Programmer's Name: [redacted]

CS3620 – Rapid Application Development
Pong/Hockey **Peer** Evaluation

Go through the program and test it as much as possible. Be sure to check Help, scoring, ball movement&bouncing, paddle movement, serving&startup placement, and options&default values. Give constructive feedback!!!

What did you find that was good? (Be specific)

The whole menu/lounch screen is interactive and intertaining.

What did you find that needed improvement? (Be specific)

The slider scale for score. Not enough ticks and too hard to set to 10.
The paddles move too slow.
Double Paddle pong never has a starting point

What parts worked? (Be specific)

Scores
Paddles.
Game Types -> appropriate components etc

What parts didn't work? (Be specific)

Double Paddle the ball starts before it is centered for user interaction. ← ~~[illegible]~~ all of them.
Bringing two menus up at once causes issues if you bring up game type + theme. If you press begin playing the ~~th~~ theme menu remains up -> change themes while you play. Score for Hockey Pong
Double Paddle causes issues for the next game.

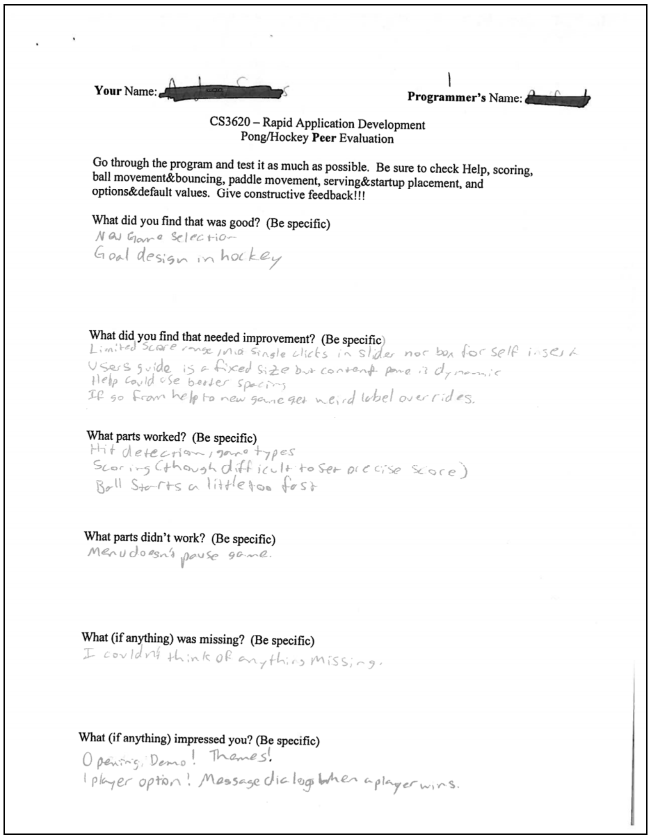
What (if anything) was missing? (Be specific)

N/A.

What (if anything) impressed you? (Be specific)

Launch screen. Themes.

Figure 5 - Peer Evaluation #2



Your Name: [redacted] **Programmer's** Name: [redacted]

CS3620 – Rapid Application Development
Pong/Hockey **Peer** Evaluation

Go through the program and test it as much as possible. Be sure to check Help, scoring, ball movement&bouncing, paddle movement, serving&startup placement, and options&default values. Give constructive feedback!!!

What did you find that was good? (Be specific)

New Game Selection
Goal design in hockey

What did you find that needed improvement? (Be specific)

Limited score range / no single clicks in slider nor box for self insert
Users guide is a fixed size but content pane is dynamic
Help could use better spacing
If go from help to new game get weird label overrides.

What parts worked? (Be specific)

Hit detection, game types
Scoring (though difficult to set precise score)
Ball starts a little too fast

What parts didn't work? (Be specific)

Menu doesn't pause game.

What (if anything) was missing? (Be specific)

I couldn't think of anything missing.

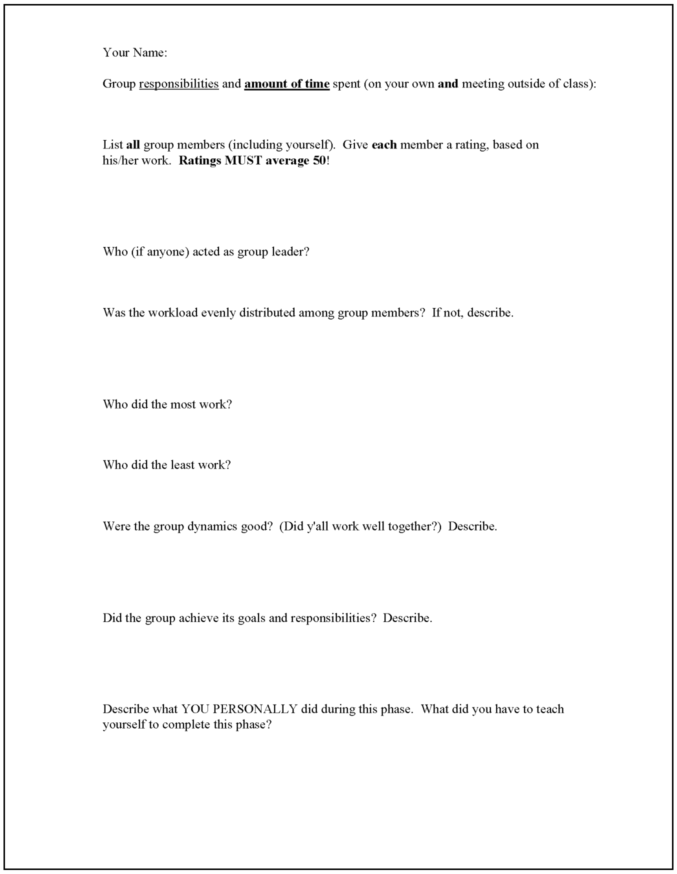
What (if anything) impressed you? (Be specific)

Opening Demo! Themes!
1 player option! Message dialogs when a player wins.

Figure 6 - Peer Evaluation #3

As with the self-assessment form, I keep the prompts on this feedback form as open-ended as possible, to allow for a wide variety of feedback. After completing a few peer project assessments, the students will then privately rank the projects, including their own. This allows students to assess their own level of success with that of their peers. This assessment exercise has an additional benefit for the professor: it often eliminates questions about how the score they received ended up being less than the score they felt they deserved, especially when they have seen and evaluated projects that may have been much better than their own.

An additional level of assessment arises when students work on group projects. (A much more complete description of how I manage and assess group work can be found in [8][9][10].) As students complete projects (or, in the case of Software Engineering, milestones), each will privately complete an assessment of the group dynamics and shared workload (see Figure 7). Students complete the assessment privately, away from their fellow team members, so that I get the most honest assessments possible. The first item helps me keep track of what project is being evaluated (and lets me know how much time students devoted to this project). The next item asks the student to list all group members and then give them a rating. I insist that all ratings average 50, which means that if some students in the group receive a higher rating, other student(s) must necessarily receive a lower one. (The original version of this item asked for an average of 75, but students immediately thought I was trying to "give everybody a C grade", so I changed it to a more arbitrary value. Students seem OK with the value 50.) The next item (Who acted as group leader?) lets the student identify who, if anybody, stepped up to take a leadership role. (I purposefully never "assign" a leader to any group.) The next item (Was the workload evenly distributed?) addresses the possibility that the workload might not have been equal for all team members. The open-ended nature of this question allows students to comment on *why* the workload might not have been the same – was it a matter of some students not doing their fair share, or was it the result of an initial distribution of responsibilities that appeared to be fair early on, but did not end up being similar.



Your Name:

Group responsibilities and **amount of time** spent (on your own **and** meeting outside of class):

List **all** group members (including yourself). Give **each** member a rating, based on his/her work. **Ratings MUST average 50**!

Who (if anyone) acted as group leader?

Was the workload evenly distributed among group members? If not, describe.

Who did the most work?

Who did the least work?

Were the group dynamics good? (Did y'all work well together?) Describe.

Did the group achieve its goals and responsibilities? Describe.

Describe what YOU PERSONALLY did during this phase. What did you have to teach yourself to complete this phase?

Figure 7: Student Group Assessment Form

The next two questions (Who did the most work? Who did the least work?) require that the student once again identify which team members, if any, ended up doing more (or less) than their peers. Ideally, these questions will reinforce the ratings given in second item on the form! The following question (Were the group dynamics good?) allows the student to assess how well (or how poorly) the team members worked together. The final item (Describe what you personally did during this phase. What did you have to teach yourself to complete this phase?) requires that students do yet another level of self-assessment by enumerating what they contributed to the group project.

Students in Senior Seminar (our highest-level offering) wrap up the course by providing a self-assessment narrative describing the work they completed, the new skill sets they mastered, how well they stayed on schedule, their level of effort and success, and how well their public Senior Project Presentations went. This narrative ends with the students themselves suggesting their own course grade, explaining their justifications for the grade. I have found that most of the students suggest a grade within 1/3 of the grade I had assigned to them in my own assessment of their work.

4 Conclusions

Beginning at the introductory level, we have introduced appropriate exercises and activities that allow our students to begin the process of developing the necessary skills to not only engage in active assessment of their own work and the work of others, but to provide useful feedback on their findings. As students progress through our curriculum, the expectations for these assessments rise accordingly. By the time they are ready to graduate, our students are very comfortable with the process and actually provide a self-assessment narrative and suggested grade for their Senior Seminar class.

5 References

[1] Paul, Richard W. and Linda Elder. “Structures for Student Self-Assessment”, The Critical Thinking Community. http://www.criticalthinking.org/pages/structures-for-student-self-assessment/458.

[2] Assessment for Learning: Strategies to Enhance Student Self-Assessment. http://www.assessmentforlearning.edu.au/professional_learning/modules/student_self-assessment/student_strategies_enhance.html.

[3] McMillan, James H. and Jessica Hearn. “Student Self-Assessment: The Key to Stronger Student Motivation and Higher Achievement”, Educational Horizons, volume 87 number 1, pp 40-49, Fall 2008. http://www.eric.ed.gov/ERICWebPortal/detail?accno=EJ815370.

[4] Rolheiser, Carol and John A. Ross, “Student Self-Evaluation: What Research Says and What Practice Shows”, Center for Development and Learning. http://www.cdl.org/resource-library/articles/self_eval.php.

[5] Schwartz, Donald R. "Beyond Programming and Software Development: Additional Teaching/Learning Goals throughout the Curriculum", Proceedings of the 2011 International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS-11), Las Vegas, July 2011.

[6] Anewalt, Karen. “Dynamic group management in a software projects course”, Journal of Computing Sciences in Colleges, Volume 25 Issue 2, December 2009.

[7] Anewalt, Karen and Jennifer A. Polack-Wahl. “Teaching an iterative approach with rotating groups in an undergraduate software engineering course”, Journal of Computing Sciences in Colleges, Volume 25 Issue 6, June 2010.

[8] Schwartz, Donald R., “Managing and Assessing Software Engineering Group Projects”, Proceedings of the 2005 American Society for Engineering Education Annual Conference and Exposition, June 2005.

[9] Schwartz, Donald R., “Service Learning and Software Engineering: Managing ‘Real World’ Group Projects”, Proceedings of the 2007 International Conference on Software Engineering and Data Engineering (SEDE-07), Las Vegas, July 2007.

[10] Schwartz, Donald R., “The Benefits (and Drawbacks!) of “Real World” Software Engineering Semester Projects: A Case Study”, Proceedings of the 2009 International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS-09), Las Vegas, July 2009.

Using a Dashboard as a Visualization Tool for Assessment Data

Courtney Lamar[1], **Velma Latson**[1], **Quincy Brown**[1], **Lethia Jackson**[1], **and Gail Fink**[2]
[1]Department of Computer Science, Bowie State University, Bowie, MD, USA
[2]Office of Planning, Analysis and Accountability, Bowie State University, Bowie, MD, USA

Abstract – *In this paper we describe the use of an instructional maturity model to develop a continuous improvement process for the Computer Science (COSC) and Computer Technology (CTEC) programs of the Computer Science Department of Bowie State University. This process was also used to meet ABET (Accreditation Board for Engineering and Technology) accreditation standards. The process involves the participation of stakeholders including faculty, students, alumni, employers, and external advisory board members. Data collected during the process is represented visually in a dashboard that identifies strengths and weaknesses of the programs. Objective, data-driven decisions are made at both the course and program levels to improve the effectiveness of the Department's curricula.*

Keywords: Assessment, ABET Accreditation, Continuous Improvement, Dashboard, Maturity Model

1 Introduction

Evidence of continuous improvement is central to the overall effectiveness of a curriculum, as well as to the ABET accreditation process. To ensure the curriculum for the computer science and computer technology programs demonstrate continuous improvement and meet the ABET requirements, a context-grounded evaluation method using a maturity model was developed. Maturity models identify strengths and weaknesses and establish standards for what works best in a program. This standard is then used to develop the best practices from the viewpoint of the organization (Jokela, Siponene, Hiraswaw, Earthy, 2006). Maturity models identify priority areas that need improvement and provide the basis for planning and implementing the actions for improvement (Jokela, Siponene, Hiraswaw, Earthy, 2006). Additionally, maturity models identify major categories for success. From these categories, levels are established that require specific goals that must be met as processes move through the levels to reach maturity.

Lesgold (2003) describes four types of maturity models: (1) Instructional Maturity; (2) Technology Infrastructure Maturity; (3) Educational Software Product Maturity; and (4) People Maturity. Instructional maturity (Lesgold, 2003) acknowledges the differences in subject area content and makes possible the development of rubrics for subject areas. Technology Infrastructure addresses the need to have adequate hardware and software to accommodate online courses. Educational Software Product Maturity monitors the software usability and examines the amount of effort involved with using appropriate software products. People Maturity ensures that the objectives of both populations of the school environment are being met for successful content and instruction of a course.

The Department of Computer Science utilizes an Instructional Maturity Model to develop its continuous improvement cycle. This model was selected because it provides a long-term approach for systematically achieving changes in learning and promoting efficient and quality student learning. This approach places the responsibility for assessment with curriculum subgroups that report to the Department Assessment Committee (DAC).

The continuous improvement cycle provides a systematic approach for faculty to collaborate on course and/or program modifications. This cycle streamlines the assessment process in a concise and clear way that is easily documented, reported and maintained. The four (4) phases in the continuous improvement cycle include (figure 1):

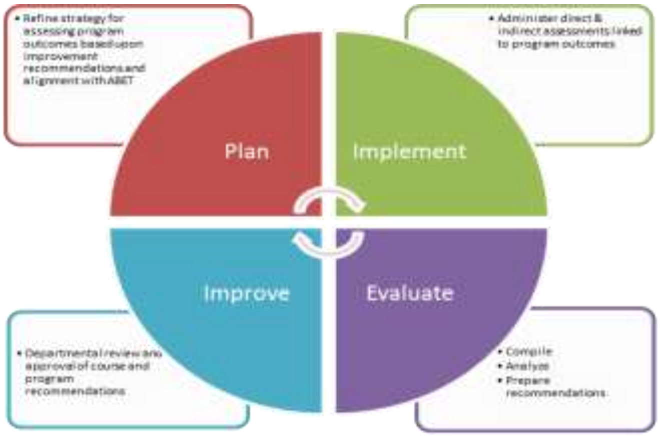

• Refine strategy for assessing program outcomes based upon improvement recommendations and alignment with ABET
• Administer direct & indirect assessments linked to program outcomes
Plan
Implement
Improve
Evaluate
• Departmental review and approval of course and program recommendations
• Compile
• Analyze
• Prepare recommendations

Figure 1 – Continuous Improvement Cycle

- *Plan Phase* - Refine strategy for assessing program outcomes based upon improvement recommendations and alignments with ABET
- *Implement Phase* - Administer direct and indirect assessments linked to program outcomes
- *Evaluate Phase* - Compile, analyze, and prepare recommendations

- *Improve Phase* - Departmental review and approval of course and program recommendations

2 Data Collection Methods

Both qualitative and quantitative data are collected throughout the department's continuous improvement process. During the *plan phase* of the continuous improvement cycle, course syllabi are updated based on the ABET student outcomes and approved department recommendations. The student outcomes are:

(a) An ability to apply knowledge of computing and mathematics appropriate to the discipline
(b) An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution
(c) An ability to design, implements, and evaluate a computer-based system, process, component, or program to meet desired needs
(d) An ability to function effectively on teams to accomplish a common goal
(e) An understanding of professional, ethical, legal, security and social issues and responsibilities
(f) An ability to communicate effectively with a range of audiences
(g) An ability to analyze the local and global impact of computing on individuals, organizations, and society
(h) Recognition of the need for and an ability to engage in continuing professional development
(i) An ability to use current techniques, skills, and tools necessary for computing practice.
(j) An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices.
(k) An ability to apply design and development principles in the construction of software systems of varying complexity.

The Computer Science (COSC) program's student outcomes include (a) – (k) and the Computer Technology (CTEC) program's student outcomes include (a) – (i).

In the *implement phase,* faculty administer both direct and indirect assessment tools such as quizzes, projects, exams, papers, group/teamwork or minute papers, surveys and questionnaires during the semester. Additional assessment tools are used to evaluate the program educational objectives (PEOs). These tools include senior exit surveys, employer surveys, alumni surveys, as well as an annual meeting conducted by an external advisory board. The senior exit survey is completed every semester; the employer survey and alumni survey are completed every three (3) years. The PEOs for COSC include:

1. Apply theoretical principles and practical tools and techniques in computing to solve real-world problems
2. Communicate effectively orally and in writing as an individual and as a member of a team
3. Become professionally employed and/or enrolled in advanced graduate studies in Computer Science or a related area

The PEOs for CTEC include:

1. Apply core technological principles to solve real-world problems
2. Communicate effectively orally and in writing as an individual and as a member of a team
3. Become professionally employed and/or enrolled in professional development, including advanced graduate studies in Computer Technology or a related area

At the end of each semester, faculty complete a course assessment report (CAR) using the student outcomes and course assessment tools. The faculty maintain a course portfolio which contains the course syllabus; good, bad, and average examples of student work products; and the CAR. Faculty enter the actual level of student attainment according to the course's student outcomes. The expected levels of attainment by course level are listed in table 1:

Table 1: Expected Levels of Attainment

Percent at Satisfactory or Higher	Level of Course
60%	100
65%	200
75%	300
85%	400

To execute the *evaluation phase,* each faculty is a member of a curriculum subgroup which meets twice each semester to discuss students' performance in a subset of related courses. The curriculum subgroups maintain meeting minutes, make course-level or program-level recommendations, and evaluate students' performance using end of semester CARs. Subgroup recommendations are forwarded to the Department Assessment Committee (DAC). The DAC meets to collate the direct assessment recommendations from the various subcommittees, as well as the indirect assessments to evaluate the extent to which the student outcomes are being met by the individual courses. The DAC submits the list of recommendations to the entire department faculty for discussion and approval.

During the *improve phase,* approved recommendations are brought forth to the faculty for implementation in courses and/or the programs.

3 Analysis of Data

Once all data has been collected, dashboards are generated to summarize the results for the student outcomes and the educational objectives of the department's programs.

3.1 Student Outcomes Dashboard

In the student outcomes (SO) dashboard, a white fill color indicates achieving or exceeding the expected level of attainment. A gray, textured fill color indicates the attainment level is less than six (6) percentage points below the expected attainment. A black fill color signals that attainment is more than six (6) percentage points below the expected level of attainment. A sample SO dashboard is illustrated in figure 2.

Analysis of assessment results for the sample SO dashboard reveals several key areas that can then be addressed at the course level. In this example, the use of the dashboard will enable the DAC to focus on improvement recommendations for student outcomes a, b, c and j in addition to specific course level recommendations.

ABET Outcome	*Course Level*	*Attainment by Course Level*	*Overall Attainment*
a	100	▨	▨
	200	▨	
	300		
	400	■	
b	100	▨	▨
	200		
	300		
	400	■	
c	100	▨	▨
	200		
	300	▨	
	400	■	
d	100		
	200		
	300		
	400		
e	100		
	200		
	300	■	
	400		
f	100		
	200		
	300		
	400		
g	100		
	200		
	300		
	400		
h	100		
	200		
	300		
	400		
i	100		
	200		
	300	▨	
	400		
j	100	▨	▨
	200		
	300		
	400	▨	
k	100	▨	
	200	▨	
	300		
	400		

Figure 2 – Sample Student Outcomes Dashboard

3.2 Educational Objectives Dashboard

A sample EO dashboard is provided below (figure 3). Analysis of its assessment data yielded a white fill color for all educational objectives. A white fill color indicates that expected attainment levels are being met.

Program Educational Objectives	*Instrument*	*Overall Attainment*
1 – Apply theoretical principles and practical tools and techniques in computing to solve real-world problems	Exit Survey	
	EAB	
	Alumni Survey	
2 – Communicate effectively orally and in writing as an individual and as a member of a team	Exit Survey	
	EAB	
	Alumni Survey	
3 – Become professionally employed and/or enrolled in advanced graduate studies in Computer Science or a related area	Exit Survey	
	EAB	
	Alumni Survey	

Figure 3 – Sample Educational Objectives Dashboard

4 Conclusion

Continuous improvement, more data-driven decisions, and an overall view of performance and impact are but three (3) benefits of the dashboard visualization tool described in this paper. This effective assessment tool uses an instructional maturity model inclusive of a continuous improvement cycle of planning, implementing, evaluating, and improving the curriculum. Using this assessment approach, which involves faculty, staff, students, alumni, employers, and an external advisory board, has enabled the department to enact incremental improvements to the computer science and computer technology curricula. Further, capturing a bird's eye view of assessment data across various data sources aids the department in quickly identifying (and correcting) performance/alignment issues related to student learning outcomes and educational objectives.

5 References

[1] Bres, M., Moore-Crawford, C. & Weisshaar, A. (2009). Streamlining assessment. *Journal of College Science Teaching* 38.5 (pp. 43). Retrieved from http://www.nsta.org

[2] Carnicom, S. & Snyder, C. A., (2010). Learning outcomes assessment in honors: an appropriate practice?[Electronic version] *Journal of the National Collegiate Honors Council* (pp. 69-82) Retrieved from http://digitalcommons.unl.edu/cgi/viewcontent.cgi?article=1278&context=nchcjournal

[3] Criteria for Accrediting Engineering and Technology Programs: Effective for the Evaluation During the 2006-2007 Accreditation Cycle (2005) [Electronic Version] Retrieved from http://www.abet.org/Linked%20Documents-UPDATE/Criteria%20and%20PP/T001%2006-07%20TAC%20Criteria%205-25-06.pdf

[4] COMAR (Code of Maryland) (2007). Office of the Secretary of State. http://www.dsd.state.md.us/comar/

[5] Fullmer, P. (2009) The assessment of a tutoring program to meet CAS standards using a SWOT analysis and action plan. *Journal of College Reading and Learning.* v40(1) (pp. 51-76). Retrieved from http://www.eric.ed.gov/PDFS/EJ867745.pdf

[6] Hall, R. H., Philpot, T. A. & Hubing, N. (2006) Comprehensive assessment of a software development project for engineering instruction. *Journal of Technology, Learning, and Assessment*, 5(5). Retrieved from http://jtla.org

[7] Henry, G. T., Kershaw, D. C., Zulli, R. A., & Smith, A. A. (2012). Incorporating teacher effectiveness into teacher preparation program evaluation. *Journal of Teacher Education* 63(5). (pp. 335-355). Retrieved from http://jte.sagepub.com

[8] Lesgold, A. (2003). Detecting technology's effects in complex school environments. In G. Haertel & B. Means (Eds.), *Evaluating Educational Technology* (pp. 38-74). New York: Teachers College Press.

[9] Marshall, S. & Mitchell, G. (2002). An E-Learning maturity model? Retrieved from Victoria University of Wellington, New Zealand, University of Teaching Development Centre Web site: http://www.utdc.vuw.ac.nz/research/emm/VersionTwo.shtml and http://www.ascilite.org.au/conferences/auckland02/proceedings/papers/173.pdf

[10] Neuhauser, C. (2004). A maturity model: does it provide a path for online course design? [Electronic version]. *The Journal of Interactive Online Learning*, 3.

[11] Pirnay-Dummer, P. Ifenthaler, D, & Spector, J.M. (2009). Highly integrated model assessment technology and tools. [Electronic version]. *Education Technology Research & Development.* (pp. 3-18). Retrieved from http://content.ebscohost.com/pdf23_24/pdf/2010/ETR/01Feb10/47968976.pdf

[12] Underwood, J & Dillon, G. (2004). Maturity Modeling Year 2: Institutional Self-Assessments, June 2004. Retrieved from http://www.evaluation.icttestbed.org.uk/methodology/maturity_model

A Heuristic Approach for Student-Outcomes Assessment Using Bloom's Taxonomy

Iraj Danesh
Department of Mathematics and Computer Science
College of Science, Mathematics and Technology
Alabama State University, Montgomery, Alabama, USA

Abstract - *Assessment techniques constitute parts of program accreditation processes and are of enduring interest to all educators, including computer science educators. A repeating Heuristic Student-Outcome-Level Cycle (SOL- Cycle) that includes student outcome assessment according to Bloom's Taxonomy, coupled with an enriching Course-Level cycle is introduced. Graphical analysis of selected topics as well as excerpts of student outcomes mapped with their performance indicators and related courses are discussed.*

Keywords: Assessment, ABET, indicators, rubric, outcomes, excerpt, taxonomy

General Terms: Student outcomes, performance indicators, technology in assessment, mapping and alignment

1. Introduction

The ABET [1] and other accrediting agencies such as SACS, NCACS, WASC grant status of public recognition to the program that meets the agency's standards, criteria, and requirements. Student outcomes (formerly program outcomes) are among the listed criteria. Many programs focus on data collection rather than information processing. Wading in through reams of paper containing raw data in an attempt to assess student learning outcomes is unproductive and inefficient. It reminds evaluators of finding the needle of proof within a haystack of evidence [5]. This paper introduces instruments for effective assessments and serves as the author's heuristic approach to simplifying and streamlining the student outcomes assessment processes.

2. Student Outcomes

The ABET 2011 definition of student outcomes states: "S*tudent outcomes describe what students are expected to know and able to do by the time of graduation. These relate to the knowledge, skills, and behaviors that students acquire as they progress through the program"[8]*. If students have achieved these outcomes, it is anticipated that they will be able to achieve the program educational objectives after graduation. Student outcomes indicate the ability of students to use the knowledge they gained at the time of graduation. ABET recommended student outcomes for computer science major are [4]:

"(a) An ability to apply knowledge of computing and mathematics appropriate to the discipline
(b) An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution
(c) An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs
(d) An ability to function effectively on teams to accomplish a common goal
(e) An understanding of professional, ethical, legal, security and social issues and responsibilities
(f) An ability to communicate effectively with a range of audiences
(g) An ability to analyze the local and global impact of computing on individuals, organizations, and society
(h) Recognition of the need for and an ability to engage in continuing professional development
(i) An ability to use current techniques, skills, and tools necessary for computing practice."

The Student-Outcome-Level Cycle (figure 1) illustrates the establishment, assessment processes, attainment and continuous improvement of student outcomes. This cycle aligns with the Course–Level cycle through a two-way channel (figure 5) that communicates useful information (feedback) back and forth between the two cycles. This promotes sustainability, provides continuous course improvement and attainment of student outcomes.

3. Performance Indicators

The ABET 2011 definition of performance indicators states: "*Specific measurable statements identifying the performance(s) required to meet the outcome; confirmable through evidence*"[8]. Performance indicators constitute dimension component of a rubric and have three critical attributes: *Content referent* is the subject content that is the focus of instruction (e.g., steps of design process), *action verb* which indicates specific performance (e.g., analyze, apply), and *value free* which indicates free of terms that express value (e.g., accurately, completely)[1]. They are concrete measurable performances students must meet as indicators of the achievement of the outcome [8].

4. Student Learning Outcomes According to Bloom's Taxonomy

Bloom's Taxonomy refers to a classification of the different objectives that educators set for student learning objectives. Bloom's Taxonomy divides educational objectives into three "domains" [3]. For each domain, performance indicators can be defined as follows:

Domain	performance indicators
Cognitive	*Knowledge, comprehension, application, analysis, evaluation*
Affective	*Receiving, responding, valuing, organizing, characterizing*
Psychomotor	*Perception, set, guided response, mechanism, complex overt response*

In this model, Bloom's Taxonomy is used in the design of four-level rubrics in two different ways: Domains as performance indicators (dimension), and domains as levels of performance (scale) [6]. Sample illustrative rubrics are given below:

A Rubric for Student Learning Levels According to Bloom's Taxonomy (cognitive domain)
Attributes framework for ABET student outcome c":
(c) "An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs."

Trait	Cognitive Domain Categories					
	1. Knowledge	2. Comprehension	3. Application	4. Analysis	5. Synthesis	6. Evaluation
Indicators						
Design a computer-based project	Identifies the necessary components	Indicate the methodologies and theories	Apply theories and knowledge to the solution	Breaking down problems to smaller problems	Put together the smaller codes to form a software	Detecting bugs and inconsistencies
Develop and Implement the Project	Defines the characteristics of the project	Interprets the characteristics of such a project	Implements the project	Discusses the features of the project	Develops the project	Evaluates alternative solutions
Improve actual computer-based system	Defines variables that improves actual system	Recognizes more parameters relevant to improve projects	Controls measures to make it more economical	Examines aspects that allows better user friendliness	Revises computer-based system and proposes alternative solutions	Evaluates economic feasibility of the solutions

An Excerpt of a Four-Level Rubric Using Cognitive Domain Categories as Performance Indicators

Trait	Categories			
	Excellent	Adequate	Minimal	Unsatisfactory
Scale	4	3	2	1
Indicators				
Knowledge				
Describe the four underlying principles of hardware design	Describes four principles of design	Only describes three principles	Only describes two principles	Describe one incomplete principle

Comprehension	………	………..	……….	……..
Application				
Analysis				
Synthesis	………	……….	……….	………..

The following three-scale rubric is designed to study graphical analysis for ABET outcome (f):

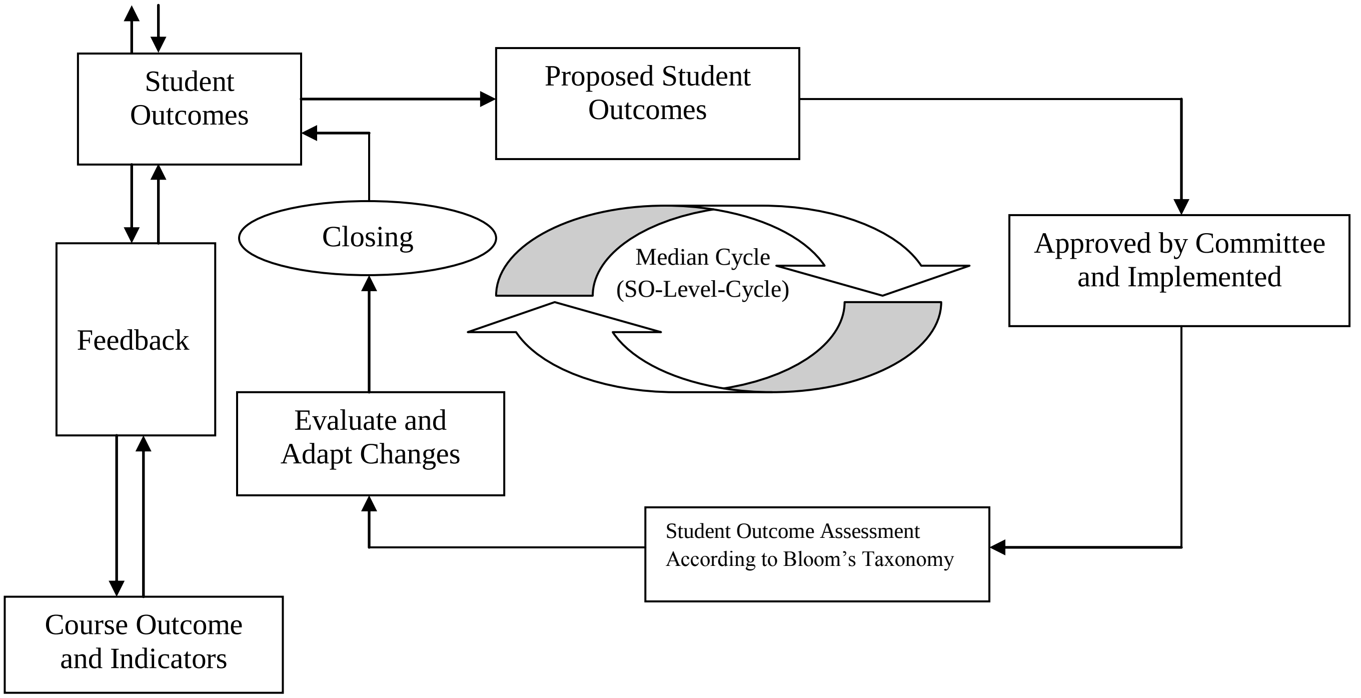
A Rubric for Student Learning Outcome

Attributes framework for ABET student outcome "f":

(f) "An ability to communicate effectively with a range of audiences"

Trait	Excellent	Adequate	Inadequate
Scale	5	3	1
Indicators			
Be able to write technical and scientific reports	Students are able to write articles and term papers	Student are able to write satisfactory project reports	Student are not able to write satisfactory reports
Be able to make oral presentations	Students are able to make outstanding presentations	Students are able to make satisfactory presentations	Students are not able to make satisfactory presentations

Figure 1. Student-Outcome-Level Cycle for Continuous Improvement (SOL-Cycle or Median-Cycle)


Student Outcomes
Proposed Student Outcomes
Approved by Committee and Implemented
Closing
Median Cycle (SO-Level-Cycle)
Feedback
Evaluate and Adapt Changes
Student Outcome Assessment According to Bloom's Taxonomy
Course Outcome and Indicators

The result of the graphical analysis of the performance indicators (written and oral communications) for a class of 25 students is shown in figure 2. Both direct and indirect measures [7, 8, 1] can be used for student outcome assessment. Figure 3 shows the role of these measurements. Figure 4 shows how the success target area is formed when direct, indirect, and performance lines meet. Figure 5 shows the author's double cycle model for continuous improvement and attainment of student outcome when student-outcome cycle aligns with course-outcome cycle. The feedback from student-outcome cycle is fed to the course-outcome cycle (and vice versa) for further improvement.

5. Use of Technology in Assessment

There are numerous commercial software tools available that enable instructors to achieve student artifacts for the purpose of assessing student learning. ABET does not endorse nor require the use of any of these commercial software products [1]. However, the following technologies, among others, have been used:

EvalTool® LMS – a simple and effective course portfolio system for LMS (features: Lessons, assignments, grade book, course syllabus, rubric, and objective evidence folder)

EvalTools® OAS – a fully tested program outcomes assessment tool (features: Course-embed assessment, senior-exit survey, alumni survey, employer survey, curriculum matrix, action item matrix, and academic administrative functions)

EFIS toolkit (Academic Evaluation, Feedback and Intervention System)[11, 2].

Figure 2. Graphical Analysis of Performance Indicators for Oral and Written Communication

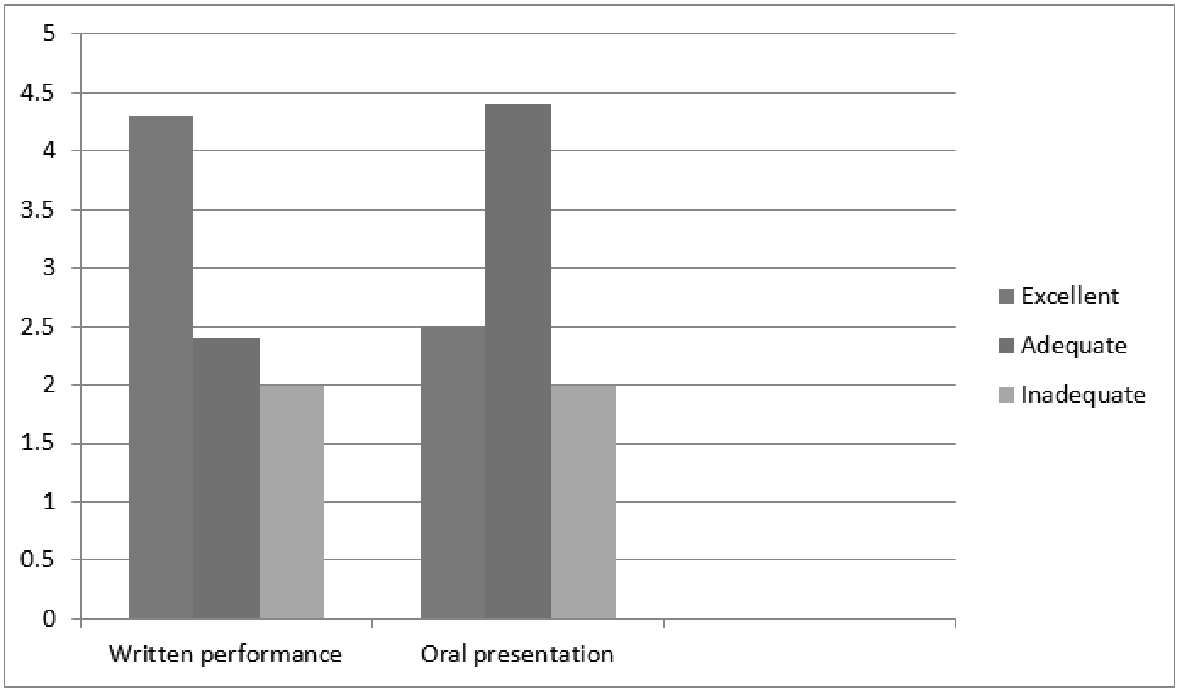


Figure 3. The Role of Direct and Indirect Measures in Student Learning Outcome Assessment

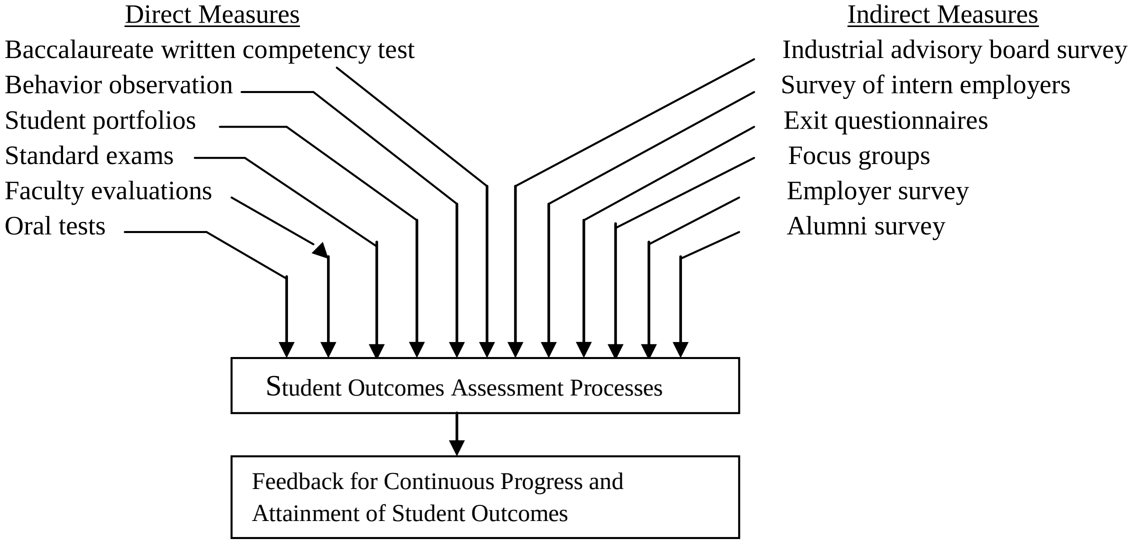

Direct Measures
Baccalaureate written competency test
Behavior observation
Student portfolios
Standard exams
Faculty evaluations
Oral tests
Indirect Measures
Industrial advisory board survey
Survey of intern employers
Exit questionnaires
Focus groups
Employer survey
Alumni survey
Student Outcomes Assessment Processes
Feedback for Continuous Progress and Attainment of Student Outcomes

Figure 4. Triangular Area Sets the Success Targets (Thresholds) for Student Learning Outcomes.

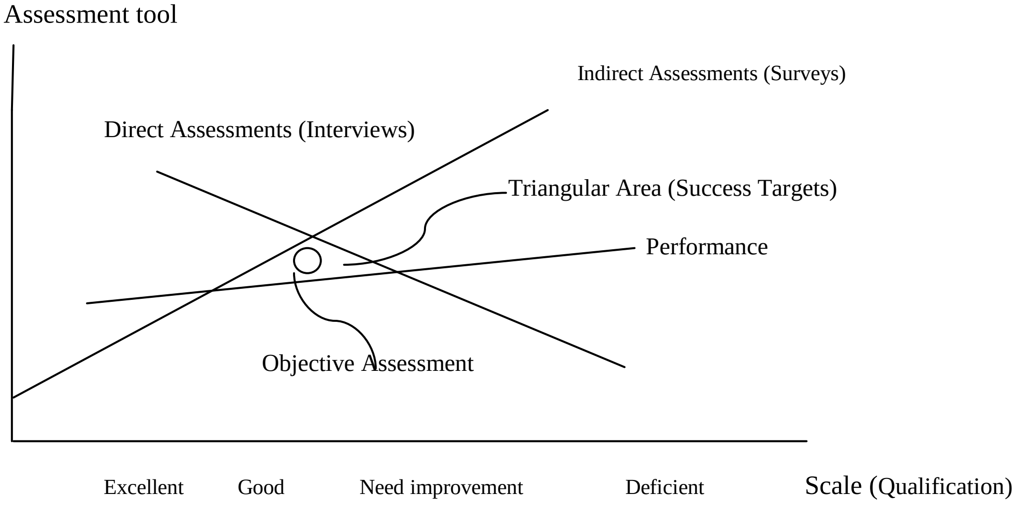
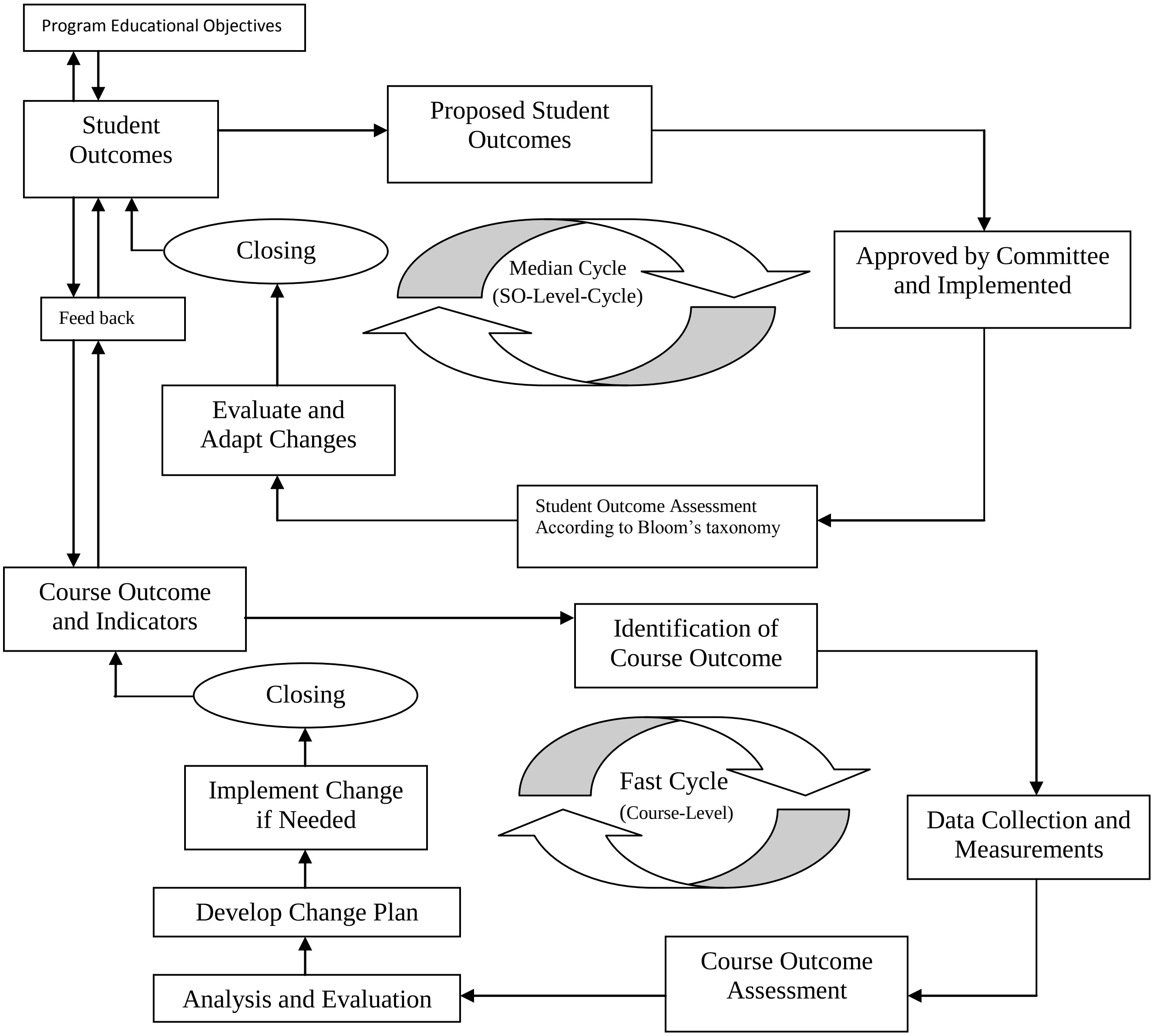


Figure 5. Alignment of Different Components of the Author's Double-Cycle Model


Program Educational Objectives
Student Outcomes
Proposed Student Outcomes
Closing
Median Cycle
(SO-Level-Cycle)
Approved by Committee and Implemented
Feed back
Evaluate and Adapt Changes
Student Outcome Assessment According to Bloom's taxonomy
Course Outcome and Indicators
Identification of Course Outcome
Closing
Fast Cycle
(Course-Level)
Implement Change if Needed
Data Collection and Measurements
Develop Change Plan
Course Outcome Assessment
Analysis and Evaluation

6. Mapping and Alignment

Each student outcome is to be matched with courses that contribute to that outcome directly and/or indirectly. However, not all courses can be assessed. Certain significant courses that have the most contribution should be selected for assessment. Table 5 shows an excerpt of ABET student outcomes mapped with content courses and selected courses in the author's CS department. Figure 5 shows alignment of different components of Student-Outcome and Course-Outcome cycles.

Table 5. An Excerpt of Student Outcomes Mapped With Relevant Courses

ABET Student Outcomes	Content Courses (significant)	Selected Courses for Assessment	Data Collected	Evaluation
(b) An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution	CSC 211, CSC 380,CSC 421, CSC 212, CSC 280, CSC 447	CSC 211, CSC 380, CSC 421	Odd academic years	
(c) An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs	CSC 311, CSC 312, CSC 330, CSC 414, CSC 442, CSC 460, CSC 280	CSC 311, CSC 312, CSC 330, CSC 280	Odd academic years	
…………….	……………….	…………….	…………	
(i) An ability to use current techniques, skills, and tools necessary for computing practice."	………..	……	…….	

7. Conclusion and Results

To have a successful student outcome assessment, for each student outcome (preferably ABET recommended CS student outcomes discussed earlier), affirm three or four performance indicators that are directly related to those outcomes, are measurable / ascertainable, and provide adequate actionable data for analysis and extraction of the results. For consistency and completion purposes, map and align them with significant contributing courses of the program. This approach assures simplicity, clarity, and facilitates reporting in tabular or graphical forms.

8. References

[1] ABET (Accreditation Board of Engineering and Technology) Program Assessment Workshop, 2012 ABET Symposium, April 2012, St. Louis, MO.

[2] AEFIS (Academic Evaluation, Feedback and Intervention System), http://www.goAEFIS.com accessed on 7/5/2012

[3] *Bloom's Taxonomy*, http://en.wikipedia.org/wiki/Bloom's_Taxonomy, accessed on 7/4/2012

[4] *Criteria for Accrediting Computing Programs*, 2012 – 2013, http:/www.abet.org/computing-criteria, 2012-2013, accessed on 7/11/2012

[5] Estel John, *A Heuristic Approach to Assessing Student Outcomes Using Performance Vectors*, ABET Symposium, St. Louis, MO, April 19-21, 2012

[6] Gomez Henry, *How to Assess the Student Outcomes*, ABET Symposium, St. Louis, MO, April 19-21, 2012

[7] Hobbs Margie, Harington Mary, hand book on Educational Program to include Student Learning outcomes, January 17, 2008

[8] Rogers, Gloria, Faculty Workshop on sustainable Assessment processes, Annual ABET conference, Baltimore, Maryland, Oct 26, 2010

[9] Shadaram Mehdi, *A detailed Process for Student Outcomes Assessment for Baccalaureate Level Programs*, ABET Symposium, St. Louis, MO, April 19-21, 2012

[10] Smith Terry, *An Efficient Approach to Sustaining Direct Student Outcome Assessment*, ABET Symposium, St. Louis, MO, April 19-21, 2012

[11] Sualp Mustafa, *Innovating Data Collection Practices to Promote Continuous Quality Improvement*, ABET Symposium, St. Louis, MO, April 19-21, 2012.

A General Course-Level Assessment Cycle for Computing Courses

Iraj Danesh
Department of Mathematics and Computer Science
College of Science, Mathematics and Technology
Alabama State University, Montgomery, Alabama, USA

Abstract *- Assessment techniques are of enduring interest to all educators, including computer science educators. They constitute parts of program accreditation processes. Course assessment is a core program assessment subset that is addressed comprehensively in this paper. A repeating course-level cycle is presented. For each component of the cycle, a survey of related methodologies and strategies including means and tools of assessments is introduced. Selection of processes that best fits the course under assessment including the use of multi-source/multi-method to maximize validity, reliability, and reduce bias of any approach is discussed.*

Keywords: Assessment, ABET, stakeholders, indicators, rubrics, caveats
General Terms: Multi-mean and multi-tool assessment, student outcomes, item analysis, success targets

1. Introduction

Course assessment is an integral part of CS program accreditation required by accrediting agencies such as ABET and SACS. Course assessment occasionally focuses on evaluation of individual student performance that usually leads to a letter grade. An assigned letter grade does not provide information regarding assessment and accreditation. The letter grade, in itself, does not have any place or role in ABET required assessments for accreditation purposes [14]. Therefore, the system of measurements used to assess the course outcomes should reflect this information and be directly linked to all the way through program educational objectives, (ABET) student outcomes, performance indicators of student outcomes, and ends up with course assessment which is the topic of study in this paper.

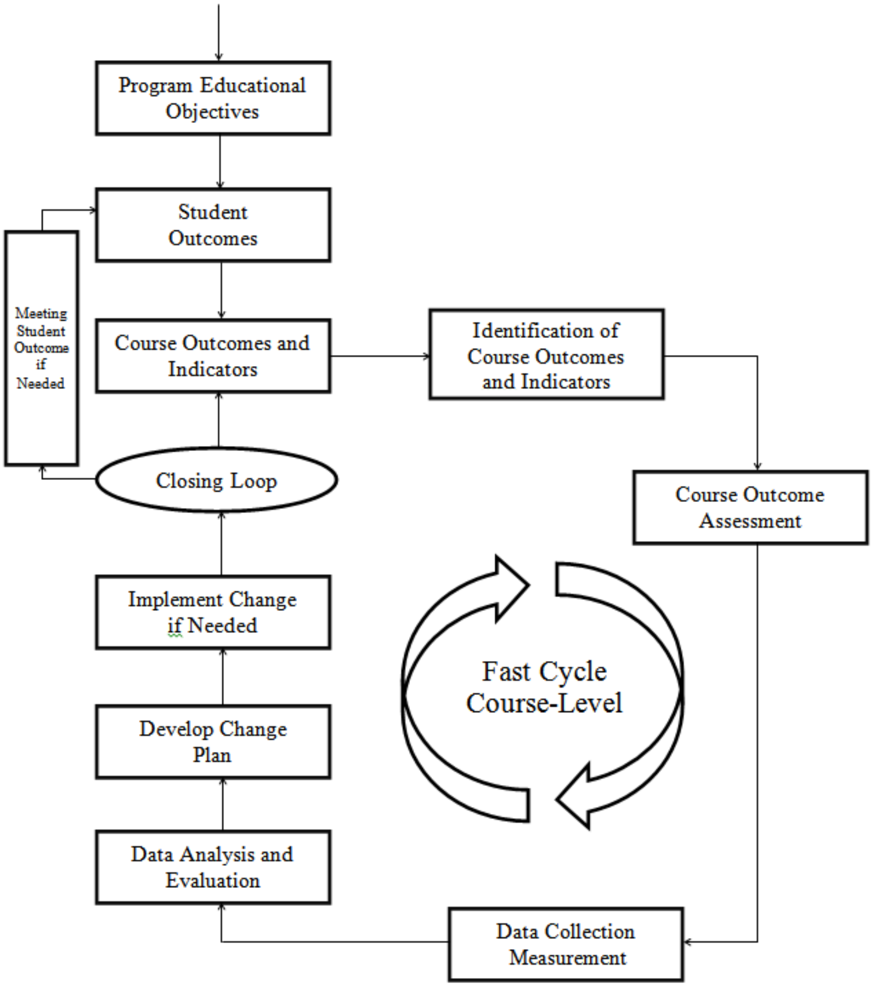
2. Course-Level Assessment Cycle

Figure 1 represents course-level assessment cycle which is a process for revision of course outcome and outcome indicators. This cycle establishes the connectivity between course-level assessment and program when course objectives are mapped and aligned with affirmed performance indicators of the student outcomes. Timeline (frequency) for this cycle is one semester or a quarter,[9]. In a course, assessment begins at concept level and goes all the way through levels of topic, course subject, actionable data collection, data analysis, evaluation, action taken by the result of the analysis. In a repeating process, weighted rubrics can be used to emphasize on detected weaker area followed by next cycle result.

2.1. Course Outcomes and Performance Indicators

Course Outcomes describe what students are expected to know and able to do by the end of the course. They are descriptions of what we intend for students to know and think (cognitive), or do (behavioral) when they have completed the course. Outcome performances are based on demonstration of Performance indicators, that are specific measurable statements identifying the performances required to meet the outcome, confirmable through evidence [1, 6, 9]. Performance indicators are concrete measurable performances students must meet as indicators of the achievement of the outcome.

Figure1. Process for Revision of Course Outcome and Outcome Indicators (Fast Cycle)


Program Educational Objectives
Student Outcomes
Meeting Student Outcome if Needed
Course Outcomes and Indicators
Identification of Course Outcomes and Indicators
Closing Loop
Course Outcome Assessment
Implement Change if Needed
Fast Cycle Course-Level
Develop Change Plan
Data Analysis and Evaluation
Data Collection Measurement

2.2 Course Outcome Assessments

ABET 2011 definition of assessment states: "*One or more processes that identify, collect, and prepare data to evaluate the attainment of student outcomes and program educational objectives. Effective assessment uses relevant direct, indirect, quantitative, and qualitative measures as appropriate to the objective and outcome being measured*." [9]. Assessment *methodologies* are instruments that are used to assess student learning**.** Assessment *means* are areas of measurement, and assessment *tools* are facilities used for the measurement of means. Together they can be used for data collection for later analysis and extracting results [6]. The following assessment methods are often used and are described below:

2.2.1 Direct Assessment.

This method is based on analysis of student artifacts and behaviors. Assessment means are tests, assignments, projects, presentations, portfolios, and papers. Assessment tools are item analysis, rubrics, and percentiles. Criterion-referenced and norm-referenced measures are important approaches to assessment work [5, 6, 7]. Qualitative assessments rely on descriptions rather than numbers (means: Journals, participant observation, writing, speaking, research, group project, and culminating (capstone) project [6]; tools: Rubrics, faculty panel). Quantitative assessments rely on numerical scores or rating (means: True/False tests, multiple-choice tests, standardized tests, department-constructed tests, and end of program capstones [6, 10]).

2.2.2 Indirect Assessment

This method is based on reported perception of student learning. Assessment means include attitudinal from students, alumni, employers, faculty, fieldwork, and supervision. Assessment tools include surveys, exit interviews, and focus groups [6, 9].

2.2.3 Other Assessments

Summative assessment is used to evaluate the effectiveness of instructional methodologies and can be described as a mean to gauge, at a particular point in time, student learning relative to content standards. They are given periodically to determine at a particular point in time what students know and do not know (means: End of the chapter tests and end of the term tests) [3, 4, 9]. *Formative* assessment is part of instructional process and is used to modify instruction. When incorporated in the classroom practice, it provides information needed to adjust teaching and learning within a set time frame (means: Criteria and goal setting, observation, questioning strategies, and student record keeping) [3, 4, 9]. *Triangular* is a *multiple-mean assessment* methodology that incorporates a combination of assessment methods. A course may have different components, and no single method may be good for all components. Assessment options have their own advantages and disadvantages, and caveats concerning methodologies and number of instruments should be dealt with accordingly.

2.2.4 Course Targets (Thresholds)

Targets are set up to achieve improvement. Table 1 shows the result of multi-mean, multi-tool assessment methods that used by the author in a course. It maps and aligns tests, course objectives, oral presentation, written report, and success targets. ***Percentiles in this table can be used for converting (translating) assessment data to a letter grade.***

Table 1. The Mapping and Alignment of Tests, Course Objectives, and Success Targets

Test #	*Objectives Evaluated*	*percentage*	*Targets or Thresholds*
# 1	*Objectives 1 and 2*	*10 %*	*At least 70% of students demonstrated excellent or good performance, and at least 90 % demonstrated acceptable or above performance*
# 2	*Objective 3 and 4*	*10 %*	*As above*
# 3	*Objective 5 and 6*	*10 %*	*As above*
# 4	*Objective 7 and 8*	*10 %*	*As above*
# 5	*Objective 9 and 10*	*10 %*	*As above*
# 6	*Objective 1 through 10*	*35 %*	*As above*
Oral presentation	*On assign topics*	*7 %*	*As above*
Written report	*On assign topics*	*8 %*	*As above*
Total		*100 %*	

2.3 Data Collection and Measurements

Rubrics and faculty panel tools are used to collect data qualitatively. Item analysis and percentiles are tools that can be used to collect data quantitatively. Scoring, in this case, is straightforward and efficient. Analyzing data is relatively easy. Rubrics commonly are used in scoring and grading (summative assessment). They are “scoring tools” that articulate the set of standards against which students’ performance are judged. They may also play an important role in instruction (formative assessment). A rubric states the expectations for student performance. It provides students with clear information how they have performed and what they need to accomplish to perform better. Rubrics provide understanding of areas of weakness, strength in student, and guide instructor in the assessment [3, 9].

A rubric generally contains three components: Dimension, scale, and descriptor. Dimension is the number of performance indicators. Attribute of dimension (content referent, action verb, value free) are used in the process of assessment. Scales indicate the levels of performance and show how good or bad they are. Descriptors spell out what is expected of students at each level of performance [9, 8]. The following table shows a generic format for rubric with a dimension of 5 indicators and levels of performance with 4 scales:

Trait	Unsatisfactory	Satisfactory	Good	Excellent
Scale	1	2	3	4
Indicator 1	Descriptor 1-1	Descriptor 1-2	Descriptor 1-3	Descriptor 1- 4
Indicator2	Descriptor 2-1	Descriptor 2-2	Descriptor 2-3	Descriptor 2-4
Indicator 3	Descriptor 3-1	Descriptor 3-2	Descriptor 3-3	Descriptor 3-4
Indicator 4	Descriptor 4-1	Descriptor 4-2	Descriptor 4-3	Descriptor 4-4
Indicator 5	Descriptor 5-1	Descriptor 5-2	Descriptor 5-3	Descriptor 5-4

Selected examples of rubrics for data collection and analysis [6, 8, 11, 12] is given below:

Oral Communication (Presentation) Assessment

Material Organization	Total point (10)
Subject	6
Content	8
Logical organization	9
Supporting material	7
Presentation and Delivery	total Points (10)
Language	6
Voice and enunciation	8
Eye contact	6
Gestures	7
Overall Effectiveness	Total points (10)
Speaker attitude	7
Audience appeal	9
Cumulative points earned	70 points out of 100

Criteria for success: Two criteria may be established for success. Overall (primary) criterion is 80% of students have an overall score of 85 or higher. Secondary (detailed) criterion is that on no component will average score be less than 7.

Computer Programming Grading Rubric

Trait	Excellent	Good	Satisfactory	Unsatisfactory
Scale	4	3	2	1
Delivery	Delivered on time	Delivered within one week of the due date	Delivered within 2 weeks of due date	Delivered more than 3 weeks of due date
Documentation	Clearly documented	Moderately documented	Barely documented	Not documented at all
Coding	Includes name, date and title assignment	Includes name and title assignment	Missing title assignment and few global variables	No name, date, assignment, or title is included, disorganized
Standards	Good use of variables	Ambiguous variables	Ambiguous variable naming	Poor use of variables
Readability	Very easy to follow	Fairly easy to follow	The code is readable	Poorly to follow
Efficiency	Efficient and, readable	Fairly efficient	Unnecessarily long	Inefficient and difficult

Written Report Grading Rubrics

Trait	Excellent	Adequate	Needs Improvement	Inadequate
Scale	4	3	2	1
Written Skills				
Clarity	The report can easily be followed	The report can be followed	The report is difficult to follow	No organization to follow
*Mechanica*l *(Sentence level)*	Concise and no mechanical Or grammatical errors	2 grammatical and mechanical errors	More than 3 errors	Poor grammar and mechanics
Organization	Writer leads the reader Gracefully	Writer leads the reader	Writer barely leads the reader	Reader is completely lost
Contents				
Format is followed	Always	Often	Usually	Never
Technical Issues	Concise and complete	Accurate but not complete	Misleading or inaccurate	Hardly explain it at all
Uses Analogies	Uses original analogy	Includes a good analogy	The analogy is not used well	No analogy at all
Utilitarian	Unusually revealing	Adequate utilitarian	Mostly one sided and confused	Entirely inadequate
Conclusion	Convincing results	Tries to justify the question	Does not justify responses	Does not answer at all

2.4. Data Analysis and Evaluation

ABET 2011 definition of evaluation states: "*Evaluation is one or more processes for interpreting the data and evidence accumulated through assessment processes. Evaluation determines the event to which student outcomes and program educational objectives are being attained*." [9]. Evaluation of data collected from summative assessment leads to final demonstrated achievements such as a letter grade. Evaluation of data collected from formative assessments leads to detection of areas of strength, weakness, and eventually is transformed to changes such as course modification, instructional methodology, and technology updates. Data and evidence accumulated through assessment processes will be analyzed to determine the event to which student outcomes, and course objectives are being attained. The following rubrics that are derived based on examples in reference [6] are sample rubrics that are used to measure intended criterion (criteria) established for success:

Summary Item Analysis* of Exam Question Assessment (across 25 students)

Test questions	# Correct answers	Total answers	% Correct answers	Criteria met	Flag of criteria
Q 1	23	25	92	Yes	below 75%
Q 2	20	25	80	Yes	
Q 3	15	25	60	No	
Q 4	21	25	84	Yes	
Q 5	13	25	52	No	

* There is a lot more to item analysis than this summary. Item analysis can be a powerful technique available to instructors for the improvement of instruction. Basic item analysis statistics which aid in evaluating the effectiveness of an item are: Index of difficulty, index of discrimination, discrimination efficiency, mean item difficulty, and standard error of measurement [13].

Analysis of a Project Assessment

Indicator's content reference	student 1	student 2	stuent3	student 4	student 5	student 6	Criteria/Percentage Student Scores 4 and above
Accuracy	4	5	2	6	4	3	66%
Efficiency	3	6	3	7	3	2	32%
Timeliness	7	4	2	8	2	2	50%
Effectiveness	6	6	3	9	6	1	66%
Total	20	21	10	26	15	8	
Student Grade	B	B	D	A	C	F	

Columns are used for individual student grading, rows can be used for assessment of intended educational outcomes accomplishment such as percentage of students that earn a score of 4 or more.

2.5 Develop Plan Change (Action Taken)

The result of the analysis establishes planned actions to rectify shortcomings and weaknesses for the next offering of the course and can lead to the modification of course objective, course instruction, technology used, target adjustment, and assessment methodology and eventually promotes pedagogy and the program it involves with.

3. Result of An Item Analysis of Exam Question Assessment

In a data structure course, the analysis of data collected in a test with 23 questions across 25 students showed that 52 % of questions were met (answered within the acceptable success target range). 2% of them satisfied the highest percent of correct responses (92%), while 20% of them satisfied the lowest (71%). 28% were distributed between highest and lowest range. The remaining 48% questions were below 70% correct response target.

In this test the expected target success criteria were not met satisfactorily. Based on the result of the analysis of the analytic rubrics used, faculty panel detected area of weakness and recommended addition of a unit in the course that emphasizes on solution of recurrence equations with expansion of recurrence relations.

4. CONCLUSION

Course-level assessment provides data and evidence, analysis, extract result from analysis, and evaluation suggesting method of improvements. While not all methodologies, means, and tools can be used for a single course, using only a single assessment method or tool is not good for measuring all outcomes of a course either. A combination of certain methodologies and assessment tools may become very effective for assessment of certain courses. Repeating process using weighted rubrics followed by second-cycle results is complementary to a thorough process of course assessment.

5 References

[1] Benta trudy, Hallmarks of effective Outcome Assessment, San Francisco, California: Jossey- Bass, 2004

[2] Cunningham, G K., Educational and Psychological measurement, New York: MacMillan publishing, 1986

[3] EbertMay Diane, Classroom Assessment Techniques---Scoring Rubrics, http://www.flaguide.org//rubrics/rubrics7.php, accessed on 11/2/2011

[4] Garrisn Catherine and Ehringhaus Michael, Formative and Summative Assessments in the Classroom http://www.amle.org/Publications/WebExclusive/Assessment/tabid/1120/Default.aspx Accessed on 10/31/2011

[5] Gronlund, N.E, Measurement and evaluation in Teaching, 4th ed. New York: Macmillan Publishing, 1981

[6] Hobbs Margie, Harington Mary, hand book on Educational Program to include Student Learning outcomes, January 17, 2008

[7] McBeach R.J. Instructing and evaluating in Higher education: A guide book for planning learning outcome, Englewood Cliffs, N: Educational Technology Publications, 1992

[8] Muller John, Rubrics, http://jfmueller.faculty.noctrl.edu/toolbox/rubrics.htm, accessed on 11/12/2011

[9] Rogers, Gloria, Faculty Workshop on sustainable Assessment processes, Annual ABET conference, Baltimore, Maryland, Oct 26, 2010

[10] Acessment Methods:Quantitative and Qualitative, http://its.fvtc.edu/langan/SCO/SCO0607/Screen1.htm, accessed on 10/31/2011

[11] http://www.csulb.edu/colleges/coe/cecs/views/programs/undergrad/grade_prog.shtml, accessed on11/3/2011

[12] http://www.teachervision.fen.com/teaching-methods-and-management/rubrics/4525.html, accessed on 11/12/2011

[13] Academic Technology Services, http://scoring.msu.edu/itanhand.html, accessed on 6/20/2012 [14] ABET Program Assessment Workshop, 2012 ABET Symposium, April 2012, St. Louis, MO.

SESSION

TOOLS AND SYSTEMS + STUDIES INCLUDING, PLAGIARISM, ATTENDANCE TRACKING, CLASS MANAGEMENT SYSTEMS, RESEARCH METHODS, CAPSTONE PROJECTS, AND OTHERS

Chair(s)

TBA

Online detection of source-code plagiarism in undergraduate programming courses

D. Pawelczak
Faculty of Electrical Engineering and Computer Science
University of Bundeswehr Munich (UniBw M), Neubiberg, Germany

Abstract - *Plagiarism in programming courses has increased in recent years. Therefore, courses need to be adapted in concept and organization. A relentless pursuit of plagiarizing or the ineffective attempt to completely suppress all communication in the classroom are certainly as inadequate approaches as the installation of an offline plagiarism detection tool without a proper feedback mechanism for the students. Ideally, teamwork is allowed, but each student submits her/ his own solution. A "real-time" plagiarism detection tool gives immediate feedback on a submitted solution during class time. Based on typical indicators, which course instructors use to manually detect plagiarism, a new software system is developed. This system allows submissions of programs, which are automatically tested for proper functionality and checked against other submissions. An evaluation of the system with data sets from an undergraduate C-programming course reveals a high rate of plagiarisms while false detection (coincident similarity) rarely occurred.*

Keywords: source-code plagiarism, coincident similarity

1 Introduction

Plagiarism is not only found in doctoral theses, but also a typical problem in undergraduate hands-on programming courses. A former survey by Sraka and Kaucic among students revealed, that 72.5% of the students admitted plagiarizing and even 75.5% of all students estimated, that students would plagiarize at least once during their study [1]. A random sample of 31 source files taken from the undergraduate C-programming course at our *Faculty of Electrical Engineering and Computer Science* (*ETTI/ UniBw M*, Munich) checked with JPlag [2] exposed 9 pairs of obvious plagiarism and even one pair with a 100% match [3]. Due to the small number of samples and according to the impression of the course instructors, the actual percentage of plagiarism is probably much higher. This selective testing raised the demand for establishing a tool for plagiarism detection in programming courses as well as further investigations on plagiarism at our university.

Today's technologies provide many simple means of sharing solutions among course participants. The technical means are certainly only one aspect that leads to plagiarism: a large list of reasons for plagiarism in computer science courses is found in [4]. Another reason especially for undergraduate courses is found in the wide range of different knowledge among the first-year students: some students have not yet seen a development environment while other students already earn money as programmers in addition to their studies. The challenge for a lecturer in undergraduate programming courses therefore is to simplify the learning contents for beginners on the one hand, and on the other hand, provide enough substance and diversification not to bore experienced students. Still the different knowledge of the students leads to totally diverging course preparation times. As a concrete example, the C-programming course at *ETTI/ UniBw M* requires according to the curriculum an average course preparation time of 10 hours per week, while students state in a survey their individual preparation time between 1 and 16 hours. The big effort for preparation arouses the desire among some students to shorten this time by taking a copy from a fellow student instead of preparing their own solution. This behavior has increased in recent years and leads to lower grades and a higher failure rate in the final examination.

This paper discusses different approaches at our faculty to minimize plagiarism (chapter 2-3), it presents an online detection tool, that checks for plagiarism during the class time (chapter 4-5) and summarizes the results of an evaluation of this tool with data from a C-programming course (chapter 6).

2 Prevent plagiarizing

At our faculty we utilize two different approaches (summarized in 2.1 and 2.2) to prevent plagiarizing [3]. A third approach (2.3) shall be established.

2.1 Prevent sharing

One typical but certainly not satisfactory approach is to prevent any import or sharing of files in the classroom. This typically involves disabling of Internet access, file-sharing access and ban on USB devices, mobile phones and so on. A further restriction is not to reveal the assignment before class time and not to reuse assignments. With all these precautions, plagiarizing is limited to class time. If a course needs preparation, the students are requested to bring in a printout or handwritten sheet of their prepared solution. The printout is typed in during class time and each student provides her/ his own solution (at least regarding the individual typing). One could say, that students might have a benefit from typing in a prepared program. In case the student types his/ her own solution, it might help as a review. But certainly a beginner typing in the solution of an experienced programmer does rarely give a benefit, because these solutions often require programming skills, which are not covered by the accompanying lectures yet. Cutting off the Internet during a hands-on course does not reflect the state of the art, as many professional programmers query the Internet for online tutorials, programming standards, etc. As cited in [5]: "(…) code copied from a website that assists in a specific task is potentially good practice." Last but not least, a cat-and-mouse game begins, as students might try to find leaks in the communication restrictions of the classroom.

2.2 Teamwork with an additional colloquium

A second approach is to ignore technical restrictions. This will certainly invite students to share solutions, so an additional colloquium is required to grade the individual accomplishment. Often a single question is enough to find out whether a student prepared the assignment on his own. The colloquium has the advantage that students can work in teams as long as each team member passes the colloquium. Allowing teamwork does not only provide the benefit of learning from each other, it also reduces the need for clandestine plagiarism. Ideally this approach requests a colloquium for each student after each lesson. Only the regular basis allows detecting asymmetric teams and enables the possibility to react quickly on cheating. Students have the most benefit of this approach because of the individual discussion with the lecturer. Unfortunately, this approach requires additional resources to perform the required colloquia.

2.3 Teamwork with individual solutions

The third approach takes some advantages of the second: ignoring technical restrictions and allowing teamwork. Instead of a colloquium, an individual solution is demanded from each student. This solution can be checked manually, tool-based afterwards or tool-based during the class time. Checking the solutions outside the class time requires a feedback mechanism while checking during class time enables an immediate feedback. Although it is often not difficult to detect plagiarism manually (see chapter 4.2), tracing the plagiarism is laborious and discussing the offense with the students is time-consuming. A tool-based approach allows passing the buck to the software tool, which claims the plagiarism.

3 Former and future course set-up

As up to now, the instructors manually checked the source codes of the students. It was often an individual decision of the instructor how to deal with cheating. One possibility was to request additional tasks or changes in the assignment, another was to completely reject a student's solution. The decision was often based on the student's level and hindsight. A similar problem arose with the completeness of an assignment. While an instructor might accept the solution of a skilled student although it does not catch every exception, the instructor might reject a solution of another student, which is more exhaustive. This situation was certainly non-satisfying neither for the instructors nor the students, but found its origin as so often in too less time and staff. So both, checking sources against plagiarism and testing the proper functionality of a student's solution should be tool-based in order to perform a fair-minded and uniform grading.

For our new didactical concept students prepare as before parts of their programming assignment at home and use the class time for electronic submission of their solutions. We will have the same class time and the same number of instructors, but due to the electronic submission, the instructors are relieved from the laborious and time-consuming task of checking sources manually for plagiarism. The instructors therefore should have more time for discussing different solutions or for answering questions regarding the assignment.

4 Plagiarism detection

4.1 Defining plagiarism

We define plagiarism as discussed in [5] as reproduction/ copying source-code without making any adaptions or just making moderate alternations. Comparing sources is restricted to sources submitted during class time. Checking against Internet sources or other databases is not performed. The level of alternations is represented by a threshold, which defines the maximum allowed similarity between two source submissions. The threshold shall be variable as it depends on the assignments: e.g. coding a function for calculating the factorial *n!* does not result in too many different solutions, compared to an implementation of a Sudoku-solver-algorithm.

4.2 Detecting Plagiarism

Many different methods how to detect plagiarism are implemented in existing tools. A detailed overview of different approaches is found in [7]. Apart from an implementation point of view, let us first focus on how instructors manually detect plagiarism, without actually comparing source files in detail. Typical indicators are:

- *Output*: Identical or similar output during run, which does not fully reflect the output as expected by the assignment, sometimes even with typos
- *Behavior*: Identical behavior when processing invalid input, e.g. entering letters instead of numbers
- *Tests*: Identical tests or test input provided by the students
- *Structuring*: Identical or similar structuring of the source code, like: defines before/ after includes, functions declarations at the beginning/ in header files/ no declarations at all, main as the first / last function, and so on
- *Indention*: Lack of indention (as typically the indention gets lost when sending source code per email)
- *Comments*: Numerous identical explaining comments, out-commented instructions reflecting a different version

- *Coding-Styles*: The source code represents different coding styles
- *Flipping*: The source code completely changes between submissions in a very short time
- *Questions*: Asking a question like: "Why does your program crash on that specific input?" often reveals, that a student is actually not the author

From a technical point of view, it is much easier to implement a token-based comparison on files as to focus on these indicators. Still every plagiarism detection tool has to deal with coincident similarity, i.e. a false detection. The challenge is to reduce on the one hand the information in order to be able to deal with simple alternations like renaming identifiers or changing string literals, and on the other hand prevent that the reduction leads to a false detection.

4.3 Course-specific requirements

As all course participants work on the same given assignment, sources will reflect a high degree of similarity and the risk of coincident similarity (false detection of plagiarism) is high [6]. Often programming assignments provide a project template, a code basis or skeleton to be completed in the assignment or have strict requirements regarding input or output of the program. These additional requirements further increase the similarity. Plagiarism detection therefore shall take code templates into account to distinguish between code fragments provided by the instructor and the actual implementation by the course participants. In case of false-detection (coincident similarity) students shall be given the chance to submit their solution (based upon the decision of the instructor) without plagiarism detection enabled. From where and how students obtained their sources in case of plagiarism is out of the scope of the detection system except with respect to the reporting: the system shall report, which participant has already submitted a similar source. Further investigations are up to the instructors.

Actually designing a tool for plagiarism detections raises the question: "What does the system expect from the student." It would certainly give a drawback, if students modify a copy of an efficient and well-programmed code towards a worse or erroneous code just to circumvent the plagiarism detection. The evaluation as discussed in chapter 6 shows, that plagiarizing rises with the difficulty of the assignment. If we resign to the fact, that some students are not able to cope with the assignment (these students probably will not pass the final examination, but they have to attend the course), we need to define, what qualifications (with respect to the course contents) students get from collaboration with other students. If a student starts working with a copy from a fellow student, but is capable of doing syntactical and semantical changes, fixing of errors (e.g. regarding invalid input), the student accomplishes certain skills, he might not have accomplished without using the initial copy. On the other hand there is certainly no learning effect by just performing editorial (lexical) changes.

5 System design

5.1 Overall system

With respect to the indicators listed in chapter 4.2, it is obvious, that a single tool would become too complex to accomplish all tasks. Neither is it possible to implement all features at once. Therefore the functionality is distributed among different sub-systems, which allows to further extent the system in future. These are:

- *PlagC2* (core plagiarism detection tool)
- *CGuide* (coding style test)
- *MOPVM* (a virtual machine for functionality testing)
- *IDE* (an integrated development environment for editing and debugging purposes), that includes the *TaskController* (an extension to the IDE, which controls the submission of an assignment)
- *Compiler & Linker*

Tab. 1 shows the responsible sub-systems for the different indicators and reflects the current development state.

Table 1. Responsibilities of sub-systems

Indicator	Responsible sub-systems
Output	*PlagC2* (checking string literals) *MOPVM* (checking for the expected output)
Behavior	*MOPVM* (checking for the correct I/O and exception handling)
Tests	*PlagC2* (checking string literals, token streams of test functions, overall source file structure) *MOPVM* (if requested by the assignment, check, if proper tests have been performed)
Structuring	*PlagC2* (checking the overall source file structure)
Indention	*CGuide* (checking the source code for proper indention: sources without indention can not be submitted)
Comments	*PlagC2* (currently only based on the file structure, a future extension is planned)
Coding-Styles	*CGuide* will warn on changing coding-styles. This indicator is not yet used for plagiarism detection.
Flipping	*TaskController* & *PlagC2* (flipping is not yet supported, as each submitted file is treated as a new without respect to previous submissions), see chapter 7.
Questions	*TaskController* & *MOPVM* (asking questions is not yet supported. In future, some prepared questions can be asked with the solution derived from the source code of the student, e.g. "is it call-by-value or call-by-reference?"
Token-based comparison	*PlagC2* (checking the token stream)

Fig. 1 shows the structure of the overall system and the interaction of the sub-systems. We use as front-end for the students the *Virtual-C IDE*, as it already implements a virtual machine (*MOPVM*), which is suitable for automated testing of C-source files [8]. All transactions between the webserver and the IDE are PIN and TAN based.

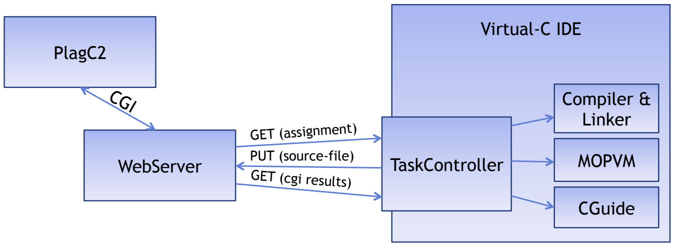


Figure 1. Structure of the plagiarism detection system

5.2 The core plagiarism detection tool

For a high flexibility and reusability the plagiarism detection tool is developed as a CGI script for usage within a webserver. The *TaskController* submits source files to the webserver and invokes the *PlagC2* tool via a CGI request (compare Fig. 1). A precondition for running *PlagC2* is, that the source file meets a minimum of requirements regarding its code style, compiles, links and fulfills most of the functional tests. This simplifies the implementation of *PlagC2*, as it gets well-formed source files.

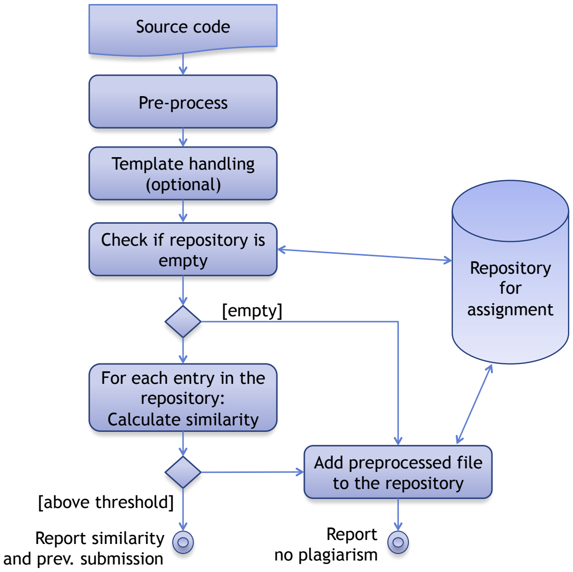


Figure 2. Schematic workflow of the core plagiarism detection tool

Fig. 2 shows the schematic workflow of the plagiarism detection tool. The source code is pre-processed and template code is optionally removed. This reduces size of the source code for a better storage within the repository as well as for a faster comparison with other submissions. In case the repository is empty, the pre-processed file is added. Otherwise, each entry in the repository is compared (see chapter 5.2.2). If a similarity is above the threshold, plagiarism will be reported with the similarity value and the TAN of the corresponding submission.

5.2.1 Preprocessing

The pre-processing of the source file reduces the C-source to four strings as shown in Fig. 3:

- *File structure*: A line-oriented representation of the code (each line is represented by a single character)
- *Identifiers*: A list of all identifiers using hash-codes
- *Literals*: A list of all string literals
- *Tokens*: A sequence of all tokens (each token is represented by a single character).

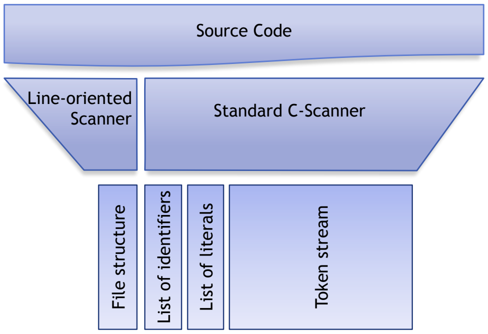
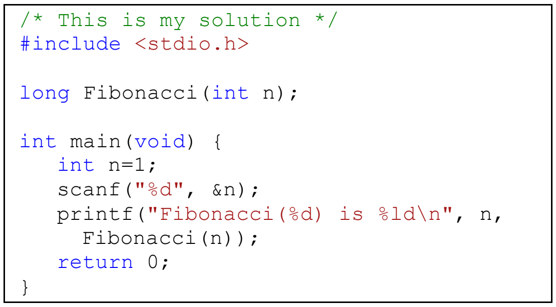


Figure 3. Pre-processing the source files

The pre-processor consists of two scanners. The first scanner is line-oriented and the second scanner is a standard C-scanner that removes whitespaces and comments, but does not perform includes. The *DLex*-Scanner generator is used to create table driven finite state automata from sets of regular expressions.



```
/* This is my solution */
#include <stdio.h>

long Fibonacci(int n);

int main(void) {
   int n=1;
   scanf("%d", &n);
   printf("Fibonacci(%d) is %ld\n", n,
     Fibonacci(n));
   return 0;
}
```

Figure 4. Example source file

Fig. 4 represents a short example file, which is used to demonstrate the four string representations in short:

File structure: The line-oriented scanner divides the source code into conceptual lines, which are represented by a single character, such as e.g.: I: includes, K: comments, L: empty lines, P: prototypes, R: return-statement, V: variable definitions, a: variable initialization, m: definition of function main(), p: usage of a printf-function, s: usage of a scanf-function, {: opening brace and }: closing brace and ?: for expressions, initialization lists, etc. In case a line contains more than one statement, only the first statement is detected, as a typical regular expression catches the whole line, like: "[\s\t]*while[^\n]*\n". The resulting string representing the structure of the example source is "KILPLmasp?R}".

Identifiers: The second string, which is derived from the source code, is holding hash-values of all identifier names (alphabetically sorted). While the source-file is tokenized, the C-scanner stores all identifiers in a symbol table. 32-Bit hash codes are derived from the identifiers in the symbol table and assembled as a hexadecimal representation in a string, as for example: "42e76a5a42fd9d16431337c1435069c145a21b63".

Literals: The third string holds trimmed versions of all string literals, i.e. all whitespaces (space-, tabulator-, carriage-return- and line-feed-characters) are removed. The result is derived from the symbol table, which keeps all string literals during the tokenizing phase. For our example the literal string results to: "Fibonacci(%d)is%ld%d".

Tokens: The forth string represents the tokens of the source file. A single character per token is used to reduce the original source file into a single string representation. Here the string is: "I<x>5x(4x);4x(#){4x;x(",&x);x(",x,x(x));R0;}". Most operators and punctuation marks are directly taken, while keywords, symbols and operators with multiple characters are reduced to a single character, as for example: 0: octal literal, 1: decimal literal, 2: char, 3: short, 4: int, 5: long, I: #include, R: return, x: identifiers, ": string literal, #: void.

5.2.2 Comparing

The four string representations are compared to the corresponding string representations stored in the repository. Except for the list of identifiers, the *Greedy-String-Tiling* algorithm is used [9]. The algorithm splits the strings into character-sequences of a minimum match length and tries to match these sequences. On a successful match the consecutive characters are checked until a different character is found. The algorithm returns the percentage of matching characters to the overall characters in the string. The advantage of this comparison is, that sequences equal or above the minimum match length can be rearranged in an arbitrary order, but the algorithm still returns a similarity of 100%. The identifiers are alphabetically ordered and provide a fixed structure, so a fast linear string comparison can be used instead of the *Greedy-String-Tiling* algorithm.

Comparing the file structure shall indicate a plagiarism as discussed in chapter 4.2. We use a minimum match length of 3 for the comparison. It is obvious, that inserting additional empty lines or putting all in one line will break the structure of the file and would reduce the similarity of two files although they are identical regarding the semantic. Detecting a high similarity in the files' structure can serve as an indication of plagiarizing but not vice versa. The same applies for identifier names. Changing the names of identifiers (e.g. using "search & replace") is regarded as an editorial change and represents plagiarism according to our definition (see chapter 4.1). It is different with string literals. A typical indicator is the output of a program that varies from the expected output. While the expected output usually is identical throughout an assignment, a match on a variation strongly indicates plagiarism. The same applies for test code, which contains identical string literals. If we can ignore string literals containing the expected output, we can put more weight on all other string literals.

The token stream is apparently the strongest indicator for plagiarism and is used in many existing solutions, e.g. such as JPlag[2] or YAP3[11]. A former approach for plagiarism detection at our faculty was also primarily based on token comparison [3]. Yet this method does not take the indicators as described in chapter 4.2 into account. For *PlagC2* we test the similarity of the token stream first. If the similarity is above the threshold, the remaining three strings are compared and the overall similarity is calculated as:

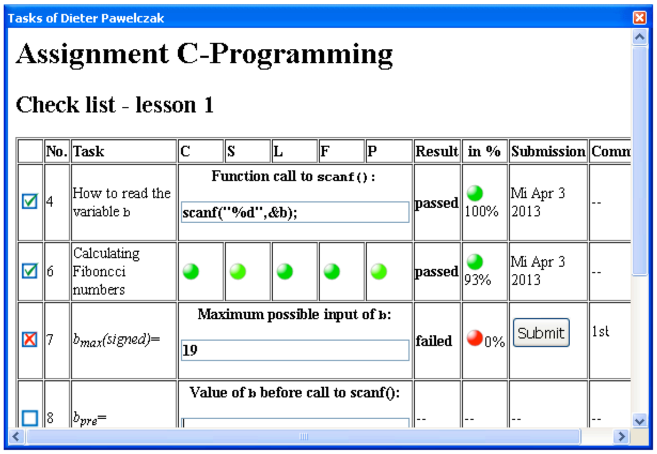
$$S = \frac{S_f + S_i + 10 \cdot S_t}{12} \tag{1}$$

$$S = \frac{S_f + S_i + 2 \cdot S_l + 10 \cdot S_t}{14} \tag{2}$$

where S represents the total similarity as a weighted average of the four similarities: S_f (file structure), S_i (identifier), S_l (literals) and S_t (tokens). Equation (1) is applied in case S_l is zero. This is typically the case for assignments with a graphical output. The equations implicitly define a theoretical threshold for a plagiarized source: In case of a 100% match on the token, but a complete different structure, different identifier names and string literals, S results for (1) in: 83.3% and for (2) in: 71.4 % respectively.

5.2.3 Template handling

A programming assignment often requests a specific output of the program or provides a file, which students complete. A template can be provided for *PlagC2*. This file represents the file as given to the students plus string literals of the requested output. The *PlagC2* tool performs the same pre-processing on the template as described in chapter 4.2.1. A variation of the *Greedy-String-Tiling* algorithm removes all sequences that match the template from the four string representations of the source code.



Tasks of Dieter Pawelczak

Assignment C-Programming

Check list - lesson 1

	No.	Task	C	S	L	F	P	Result	in %	Submission	Comm
☑	4	How to read the variable b	Function call to scanf(): scanf("%d",&b);					passed	● 100%	Mi Apr 3 2013	--
☑	6	Calculating Fiboncci numbers	●	●	●	●	●	passed	● 93%	Mi Apr 3 2013	--
☒	7	$b_{max}(signed)$=	Maximum possible input of b: 19					failed	● 0%	Submit	1st
□	8	b_{pre}=	Value of b before call to scanf():					--	--	--	--

Figure 5. Screenshot of the *TaskController* view.

5.2.4 The user interface

The *TaskController* extends the IDE by an additional web view, which allows to answer questions regarding the current assignment and to submit the source code for code and plagiarism checking. Fig. 5 shows the user interface. The results are presented in a traffic light scheme for the individual checks: Green refers to success (no errors, no warnings), yellow (no errors) and red (errors). The abbreviations for the checks are: C (source compiles), S (coding style), L (object file links), F (functional tests) and P (plagiarism detection).

6 Evaluation of the System

For the development and evaluation of the system, students were asked to provide in an anonymous form their assignment results of the undergraduate course *C-Programming* (spring 2012), which have not yet been checked against plagiarism. The course consists of 8 different assignments (see Tab. 2). 50 students submitted about 30 (different) source files per assignment. Instead of a submission during class time, the source files have been consecutively submitted with a random PIN to the webserver as a simulation of the real system.

Table 2. Plagiarism Results
(Programming in C, Spring-Course 2012)

Threshold for plagiarism: 80%	**Assignments 1-8**							
	Integer	***Floating Point***	***Strings I***	***Structures***	***Strings II***	***Arrays***	***Lists***	***Files***
	1	***2***	***3***	***4***	***5***	***6***	***7***	***8***
Detected plagiarism (Similarity above threshold)	25%	32%	37 %	52%	37%	57%	58%	65%
Average similarity	64%	74%	77%	75%	71%	78%	77%	79%
Average similarity above threshold	87%	89%	93%	89%	85%	89%	89%	90%
Linear gradient[a]	0.55%	0.51%	0.28%	0.06%	0.34%	0.16%	-0.11 %	0.04%
Linear gradient below threshold[b]	0.42%	0.56%	0.46%	0.24%	-0.02%	-0.40%	-0.07 %	-0.17%
Number of submissions	32	28	30	31	30	30	26	29

[a] gradient of a linear trend line from submission 2 to the last submission
[b] gradient without samples above the threshold of 80%

The results in Tab. 2 show, that the similarity of source files increases with the complexity of assignments. Although the number of tokens is much smaller in the first assignments, these diverge much stronger than the results of the later assignments (see also Fig. 7). The source lines of code are about 40 for the first and 220 for the last assignment. The highest similarity of assignment 8 can be explained, because this assignment had been reused from the previous year.

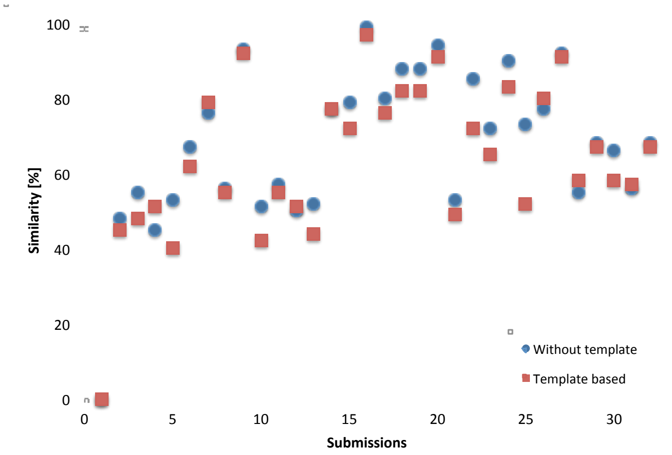


Figure 6. Evaluation of assignment 1 (Integer)

Fig. 6 shows the similarity results of the 1st assignment with and without applying a template file. The first submission always provides zero similarity, because the repository is still empty. The usage of the template file reduces as expected the calculated similarity. In Fig. 6 the average reduction is about 5%. The similarity increases with the submissions as the data in the repository grow. Nevertheless, as can be seen in Fig. 6, students can still provide individual solutions up to the last submission. This is an important outcome of the evaluation as a system-inherent drawback is, that the conditions for submissions are not equal for all students: with each submission, the repository might get a new data set for comparison. It obviously depends on the assignment, if the condition to pass the plagiarism detection gets tougher: Tab. 2 lists the linear gradient for the calculated similarity versus the number of submissions. This is high for the first (about 24 submissions let the similarity increase by 10%) and irrelevant for the later assignments.

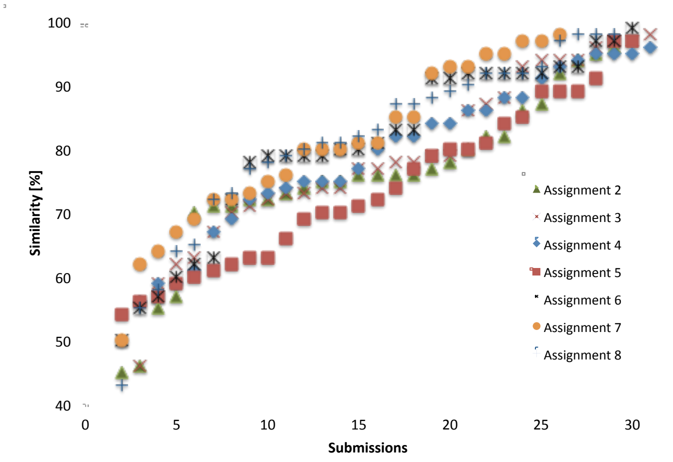


Figure 7. Similarity of assignments 2-8 (sorted)

The results in Fig. 7 reflect the increase of plagiarisms for later assignments. For a better representation the similarity is viewed in ascending order.

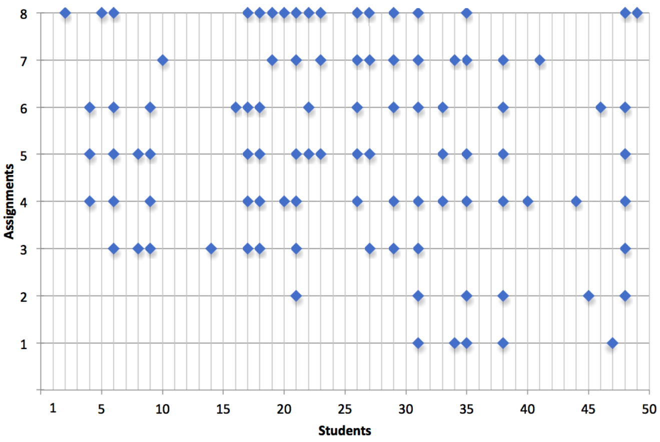


Figure 8. Distribution of plagiarizing (students versus assignments)

Unfortunately, the submissions to the assignments are incomplete. There are only a few students, who provided their source code for all assignments. Therefore Fig. 8 lets us expect, that some students (e.g. no. 31, 35, 38, 48) utilized unpermitted collaboration through out the course, while other students started taking advantage of collaboration in the later assignments (e.g. no. 17, 18, 26). The assignments 4, 5 and 6 fell in the time of re-examinations; this explains the accumulation here (e.g. no. 4, 33). Apart from assignment 8 the number of isolated cases is small (e.g. no 10, 14, 16, 40, 41). Checking these cases manually revealed, that except for two cases, the source files are nearly identical to previous submissions. The first case is obviously an earlier version of another submission, as it contains code fragments, which have been removed or out-commented. The second case (no. 40, assignment 4) is a false detection (coincident similarity) with a weighted similarity of 80%: the manual comparison revealed an individual work. As assignment 4 is a graphical application and does not provide text output, equation (1) has been applied. A high similarity on identifier names, structure and token streams obviously led to this false detection.

7 Conclusion and Future Work

Cheating in programming courses has widely spread. On the other hand prohibition of collaboration does not reflect the daily work of a software engineer. A compromise needs to be found between collaboration and individual accomplishment. A tool based plagiarism detection, which reports the offense during class time to the student, provides the advantage that students can discuss the offense with the instructors immediately. Furthermore we need to avert the danger, that students modify a well-programmed code towards an inefficient code just to circumvent the plagiarism detection. Therefore code-checking and detecting plagiarism should always be performed in combination.

Apart from the source code and token comparison several indicators have been defined, which call course instructors' attention to plagiarizing. Based on these indicators a software system is build to check source code automatically for proper functionality and against other submissions. The evaluation of the system shows that almost all cases of reported plagiarism actually represent copies with none or minor (mostly lexical) modifications or simple alternations. Due to the sound evaluation results, the system will go on stream for the upcoming C-programming course (spring 2013).

Some indicators for plagiarisms, which are discussed in chapter 4.2, have not been fully integrated into the system. An interesting approach as described in [6] is to rather report on a change in the coding style of a student, than to report a suspicious similarity. This approach can be integrated into the system in future and would further avoid coincident similarity. It would also reduce the so-called "flipping" we observed during the evaluation phase of the electronic submission system: A student tried at the end of class time hastily to submit different versions (received via e-mail) into the system, but all versions were rejected. Such a behavior is expected with the system in operation and should be prevented by the system in future.

8 References

[1] D. Sraka, B. Kaucic, "Source Code Plagiarism" Proceedings of the ITI 2009, 31st Int. Conf. on Information Technology Interfaces, June 22-25, 2009, Cavtat, Croatia

[2] L. Prechlet, G. Malpohl, M. Philippsen, "JPlag: Finding plagiarisms among a set of programs." Technical Report 2000-1. Karlsruhe: Fakultät für Informatik, Universität Karlsruhe; 2000.

[3] C. Engelhardt, "Implementation and evaluation of a software for plagiarism checking of program source code", Bachelor thesis, UniBw M (in German: "Realisierung und Evaluation einer Software zur Plagiatsprüfung von Programmquelltexten", 2012.

[4] E. Roberts, "STRATEGIES FOR PROMOTING ACADEMIC INTEGRITY IN CS COURSES," 32nd ASEE/IEEE Frontiers in Eductation Conference, November 6-9, 2002, Boston MA

[5] G. Cosma and M. Joy, "Towards a Definition of Source-Code Plagiarism," IEEE Trans. Education, vol. 51, no. 2, pp. 195-200, May 2008.

[6] A. Ohno, H. Murao, "Work in Progress - A Novel Methodology to Reduce Instructors' and Student's Psychological Burdens in Source Code Plagiarism Detection", 40th ASEE/IEEE Frontiers in Eductation Conference, October 27-30, 2010, Washington DC

[7] G. Cosma and M. Joy, "An Approach to Source-Code Plagiarism Detection and Investigation Using Latent Semantic Analysis", IEEE Trans. Computers, vol. 61, no. 3, pp. 379-394, March 2012.

[8] D. Pawelczak, "Virtual-C IDE - USER MANUAL", available online at: https://sites.google.com/site/virtualcide/ [2012-04-06]

[9] M. Wise, "String similarity via greedy string tiling and running Karp-Rabin matching", Department of Computer Science, University of Sydney, Technical Report Number 463, March 1993

[10] A. Aiken, "Moss: A System for Detecting Software Plagiarism," Univ. of California-Berkeley, available online at: http://theory.stanford.edu/~aiken/moss/ [2012-04-06]

[11] M.J. Wise, "YAP3: Improved Detection of Similarities in Computer Program and Other Texts," Proc. 27th SIGCSE Technical Symp., 1996, Philadelphia, Pennsylvania, USA, February 15-17, pp. 130-134

Some useful components of an effective class management system

Luby Liao
Department of Mathematics and Computer Science
University of San Diego
San Diego, CA 92110
liao@sandiego.edu

Abstract—**This paper describes a simple and effective class management system I developed over the years using the Google family of applications, and use in all my classes. The components of the system help gather students' information in constant time, create class mailing lists for discussions and announcements, organize students' work in a class portfolio that persists after class, enable collaboration among professor and teaching assistants in keeping students' performance record, provide students timely feedback of their course work, and allow them to provide feedback anonymously. While simple, it serves all my classes well, and offers functionalities such as grade reporting that complex systems such as Blackboard do not. Some components, such as the class portfolio and the mailing list can continue to be useful after semester ends.**

Keywords-class management system; CMS

I. Introduction

This paper presents a simple and effective class management system (CMS) I developed over the years and use in all my classes. The components of the system are Google spreadsheets, Google forms, Google groups, Picasaweb, and blogger.com blogs. Note that all these are incidentally Google products. Similar software from other vendors can be used in place of the Google's. While simple, it serves all my classes well, and offers functionalities such as grade reporting that complex systems such as Blackboard[1] do not. I expect readers will find it intuitive, and requires little training to put to effective use.

II. Day one of the semester: create a potentially permanent class portfolio, mailing list and/or photo album

On the first day of the semester, I create the following components with students' help.

2.1 A class portfolio

I ask students to provide basic information such as names, pictures, and their preferred email addresses. This is accomplished with little time and effort. I first create a Google spreadsheet and make it editable by anyone. To communicate the randomly generated, very long and incomprehensible URL of this spreadsheet to students, I tweet it and direct students to my tweeter home http://tweet.com/lubyliao, where they will click on the link of my most recent tweet to start providing data to the spreadsheet. So that others would not be tempted to click on this link, I start the tweet with the phrase *Please ignore*. Students enter their last names, first names and emails into the first three columns.

If the class roster provided by the university does not include students' pictures, or in my case, the pictures are too small, or they are not current, and some students do not even have pictures, there are a few options. The first option is to take students' pictures; upload them immediately to Picasaweb; tweet the URL of the album; and have students access my tweet, locate their pictures and enter its URL into the fourth column of the spreadsheet. In contrast to entering names and email addresses that takes basically constant time, because students can simultaneously enter data into different rows, physically taking students' pictures requires time proportional to the number of students. If the class is sizable, then use of a high-speed camera will help reduce the wait time between pictures. You could also delegate picture taking to others to speed up the process. Yet another option is to ask students to provide a URL to their pictures, and only take pictures of those who do not have publicly accessible pictures. Having a class album accessible to students helps foster community building; students can know one another quickly and become learning partners. An example of a class portfolio can be found at [2], and a class album at [3].

2.2 A class mailing list

Using the emails students entered into column three, I can create a Google group for the class in seconds, and add all students to the group. This Google group is used for announcements and discussions.

After the creation of the Google group, it is easy to change the setting of the class portfolio so that it is only accessible to the students by using the email address of the Google group such as cs305-spring-2012@group.google.com.

The readers should be warned of potential difficulties and annoyances while trying to add members to a Google group.

An easy way to populate a Google group with students is to ask them to request to join the group by, for example, visiting the group's web site. To prevent people from creating Google groups and adding members who do not wish to be in the group, Google has policies in place that make it difficult to add members directly. In September 2012, I was allowed to directly add no more than ten people to a group at a time, and no more than 30 people on any given day. Furthermore, Google requires an additional captcha test for any batch of direct adds, notwithstanding that to be able to direct add, one must have already been authenticated as a group manager. I suggest that for a large class, you provide a link that allows students to request to join the group, putting the burden on students. However, in such cases, you will find students who drag their feet in signing up.

There are many benefits to using a Google group as a class mailing list. All the group messages are archived, easily accessible and searchable from the group's web site. This frees everybody from having to save and manage group messages. A class mailing list in my institution's Blackboard vanishes by default when the class ends, rightly so because Blackboard is a class management system. In contrast, the Google group for your course can exist permanently, serving you and students years after the class. For example, I routinely send job and speaker announcements to multiple Google groups from previous years. This would be difficult to do using Blackboard. I suggest that when you use a Google group to communicate with a class whose students have long graduated, you provide a simple message footer enabling them to opt out of the group with one mouse click.

III. Class Gradebook

Just as easily, I use students' last and first names in the first and second columns of the class portfolio to create a grade book for the class. For this, I use a new Google spreadsheet. This spreadsheet is shared with the teaching assistants (TAs). TAs work on it throughout the semester to manage students' assignment, quiz and exam scores. After each exam, I access the spreadsheet to compute students' performance percentages and grades. Before Google spreadsheet became available, my TAs and I collaborated by mailing a Microsoft Excel file back and forth. Some obvious problems with this approach were:

1. My access to the grade book always had latency. That is, between when I wanted to access it, and when I actually received it from my TAs, could be a long time. In one dramatic situation, my TA claimed he had emailed me the Excel file, which I never received, and took off to vacation in the Bahamas.

2. In the current setup, when a student catches a mistake on the grade book, such as a score recorded incorrectly, either the TAs or I can correct it quickly without having to bother the other. In the Excel approach, how does the professor make such a correction when the Excel file is in the hands of the TAs?

3.1 Grade reporting

In order to give students feedback on how they are doing in class, I publish the grade book while masking students' identities. To make it easy for students to identify themselves in the grade book, I solicit a nickname from them early in the semester using a Google form, and add their nicknames to the grade book in a new column. The Google form can be constructed in seconds and emailed to the class mailing list to collect students' nicknames. For someone unfamiliar with Google form, it may take a few minutes to create one. But once created, it can be reused over and over again for other classes.

Note that a spreadsheet can contain any number of sheets. In my case, the published grade book is actually a separate second sheet of a Google spreadsheet that contains students' records in the first, private sheet. The published sheet only contains formulas that expose selected data in the private sheet while hiding whatever you wish to hide, including students' identities. For example, assuming the private sheet is named *private* and, in it, the third column contains students' nicknames, then the cell A1 of the published sheet may contain the formula *=private!C1*, meaning the value of cell C1 of the private sheet. As this formula is propagated downwards, the first column of the published grade sheet contains students' nicknames. Propagating the formulas eastward has the effect of publishing the whole grade book with the exception of the students' names. For an example of a published grade book, see [4].

In the same way that you can select which sheet and what data to publish, you can choose when to publish what you wish to expose. These are useful features of an effective class management system.

IV. A way to obtain anonymous feedback from students

When students wish to express their opinions, they can talk to professors in person, or email them or the class mailing list. What if students wish to remain anonymous for whatever reason? Such feedback can be useful to your class. For example, they may point out some of the things you are not doing optimally, or they may bring up things you wish you had thought of clarifying in class. As an example, when I taught a course on web development in 2011, I ignored PHP completely out of my distaste for it. This caused quite a few students to comment anonymously in the semester-end teaching evaluations that they wished I had given them some guidance on PHP. Short of an effective way for students to express such thoughts, these comments often turn up in the course evaluation at the end of the semester. By then the semester has ended, and that leaves professor no opportunity to either change the course, or clarify the matter in class.

An easy way to allow students to provide anonymous feedback is to use a blog that allows anonymous comments.

V. A WEB SITE TO TIE ALL THE COMPONENTS TOGETHER

It is important to provide a natural entry point for the class management system. This provides a one-stop service that you refer students to. I suggest that you create a home page for your class, and link to it from your home page. The class home page can contain any relevant class material including the class CMS. For an example of my class home pages, see http://zen.sandiego.edu.

VI. CONCLUSION

I am sharing the class management system that I developed, and use in all my classes, hoping that it will be as useful for you as it has been for me. The components are the Google family of applications, but other software can be used instead. Compared to other complex class management systems, it is intuitive, easy to use, and requires little training.

A system you create to manage a class can continue to serve you and your students years after the class. For example, the class portfolio stores the students' work permanently. It can be a useful reference when students apply for scholarships or jobs. Class mailing lists can also be used to maintain contact, and the associated Google group archive serves as a collective class memory.

REFERENCES

[1] http://www.blackboard.com/

[2] http://goo.gl/wfjWi

[3] http://goo.gl/Rdw59

[4] http://goo.gl/3aSab

Portal of Research Methods and Methodologies for Research Projects and Degree Projects

Anne Håkansson
Department of Software and Computer Systems,
The Royal Institute of Technology, KTH, Kista, Sweden

Abstract *Research methods and methodologies are extremely important when conducting research and degree projects. The use and application of the methods and methodologies are considered to be "necessarily vicious" and, unfortunately, often applied after the research has been conducted. The need for applying methods before the actually research and the reasons for doing so are often stressed in the literature and courses for research and scientific writing. This includes the aspects of selecting, understanding and applying research methods for a selected project. Unfortunately, it is difficult to choose well-suited methods and too often the selected methods and methodologies do not match each other. Instead, methods are applied without knowing about the consequences the applied method have both on the other chosen methods and on the results of the work or research. This paper provides a portal of research methods and methodologies to help the students to choose and apply the most suitable methods by illustrating which methods belong together and the distinctions between the different methods.*

Keywords: Research Methods, Methodologies, Research Projects, Capstone Projects, and Method Education

1 Introduction

When conducting research or degree projects, methods and methodologies are essential to plan and steer the work to achieve proper, correct and well-founded results. The research methods and methodologies are processes, or particular courses of actions, that assure the quality of results of the research and projects.

In some sciences, like the different disciplines within computer sciences, and computer engineering, the focus is often on the work with the research rather than methods or methodologies for the research. Although, while focusing on the work, it is important that the research is supported with research methods and methodologies in the beginning of the process, since these affect the outcome.

The deprivation of significant constituents in the research process, effect the outcome of the research and projects. This can lead to misleading results that, unfortunately, are disseminated to the reasearch community and deceive other researchers.

As soon as the project is chosen, the research methods and methodologies should be considered. Choosing includes the whole spectrum of methods from assumptions, via collection and analysis, to theses and report writing. Many education programmes, bachelor, master and PhD programmes, include research methodology and scientific writing courses where students are supposed to learn and apply methods for research, and investigations, as well as, theses and reports.

Lots of good and rigorous literature about methods and methodologies has been provided during the years [1-4; 6-8; 11-16]. Also, a lot of material for carrying out projects and writing the theses has been provided for the students, like the formalities, i.e., the thesis organization [17], the content and reference system [9; 10], methodologies [3], as well as, the language [5; 17]. This literature is only a small selection of the vast collection of written material for research methods, methodologies and writing and is not a comprehensive set of material.

However, methods and methodologies are commonly neglected, or avoided, and regarded as "necessarily vicious". Typcially, students consider the research methods and methodologies at the end of the research process, and enforce these after the work has been carried out and the theses need to be completed. Then, the research methods and methodologies, in general, are applied by "bending the methods" or, even worse, inventing new methods to suit their existing working process.

Moreover, when students choose methods for the research, projects and theses, they too often pick methods that do not match. Often methods that more or less suit the work are applied without considering the consequences these applied methods infer on each other.

Also, besides the lack of quality research methods, discussions about good foundations for the project together with acceptable arguments are often missing, as well as, discussions about validity, reliability, replicability and ethics. The students may not even be able to distinguish between the research method and data collection method.

This paper provides a portal of research methods and methodologies supporting students in selecting and applying the best suited methods by explicitly illustrating, which methods belong together and, thereby, showing distinctions between the different methods. The portal contains the methods and methodologies that are most commonly used in information and communication technology, which includes disciplines in computer sciences.

2 Choosing Methods and Methodologies

Carefully choosing methods and methodologies for a research or degree project is often a tedious and difficult task. There are many methods and methodologies, and studying all them requires a lot of effort. Some of these have the same name, which can make the selection confusing (for example experimental research methods and experimental research strategy) at the same time it can support the selection. A good example of a confusing definition is the term research methodology that refers to: the "systematic process of carrying out the research work and solving a problem including research methods" and "learning the logic behind the research methods, explaining why a certain method is used and not another one to support evaluation of research results". However, methodology also refers to "the systematic steps in data collection and data analysis" [13] including quality assurance, i.e., validity, reliability, replication and ethics [16].

Research methods, as a general concept, include a spectrum of methods. It is often defined as "the search for knowledge" or "systematic investigation to establish facts". The concept of research methods is also confusing because it can be considered to be a term for: "a collection of different methods, ranging from philosophical assumption to data analysis" but it can be considered as "a part of methodology denoting its own category of methods".

Early in the working process of research projects and degree projects, students know the subject to study but do not have enough background to know the problems within the subject. This requires a literature study and a planning of the activities to reach desired outcomes, or goals, and achieve the expected result(s). The literature study gives a foundation with background knowledge in the subject area and several possible problems that need to be solved. From the study and revealed problems, a problem statement is crystallized and a research problem formulated. Also, tentative goals and results are developed.

To reach the goals and results, a strategy (or strategies) for conducting the research or degree project is required, which must be implemented to carry out the research or degree project. The strategy consists of the different methods applied to steer the work that leads to correct, valid and reliable results. This does not imply that the results, from conducting the research, are the researchers desired results but will be correct results according to the work that has been carried out. The correct results, produced by the chosen strategy, are extremely important for the research and the society.

A main categorization of a research project and degree projects is to decide methodologies to apply to the research project. The methodologies are the processes, followed during the entire research activity. These are used as guidelines to carry out the research project. Quantitative methodology and qualitative methodology are considered to be polar opposites [12]. Roughly, the first decision is whether or not the research or degree project is about proving a phenomenon, by experiments or testing a system with large data sets (quantity), or if it is about studying a phenomenon, or an artifact, to create theories, products, and inventions by probing the terrain or environment (quality).

3 The Portal of Research Methods and Methodologies

To support selecting the best-suited methods, a portal of research methods and methodologies is provided. The portal makes a distinction between the main methods, i.e., quantitative research and qualitative methods, but also between other methods, ranging from assumptions to quality analysis.

The left side of the portal belongs to the quantitative research methodologies using experiments and large data sets to reach a conclusion; the right side is the qualitative research using investigations (or development) in an interpretative manner on, commonly, rather small data sets, to create theories or artifacts, see Figure 1.

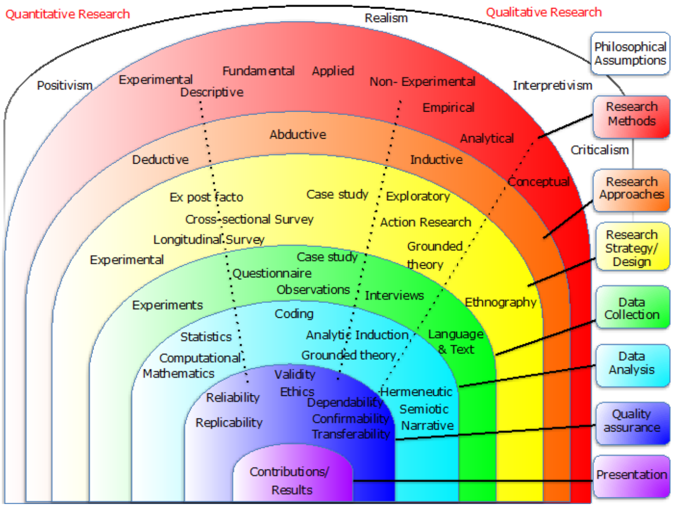


Figure 1. The portal of research methods and methodologies

The interpretation of the figure is that the left side includes the methods that work well with quantifications and the right side includes methods that work well with qualifications. Moving from the left side to the right, the methods move from being more quantitative research methodologies to more qualitative research methodology on the right side, with methods that more or less suit quantifications or qualifications.

The methods in the middle of the figure can move to both sides, left and right, deciding which part best suits the research or the degree project at hand. Once decided, it should not be a mix of methods from quantitative methods and quantitative methods within the different layers of the portal.

Hence, if the research is of the quantitative type, the methods that are on the right side on the second doted line, in the figure, Figure 2, shall not be used. This also applies for the opposite direction, from right to left, where the methods on the right side should not cross the second line, to the left, and, hence, the methods of qualitative type shall not use "pure" quantitative methods, as illustrated in Figure 3.

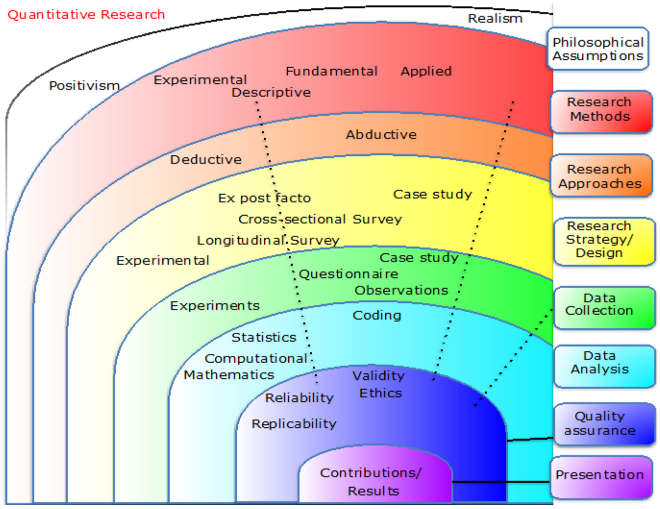


Figure 2. The quantitative research methods and methodologies

In the figure, Figure 2, all the research methods and methodologies that work well for quantitative research projects are presented. In figure below, Figure 3, all the research methods and methodologies that work well for qualitative research projects are presented.

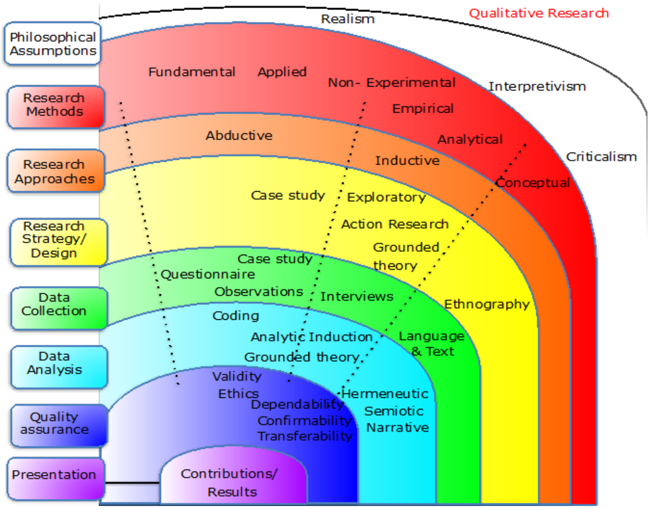


Figure 3. The qualitative research methods and methodologies

When selecting methods for the research project or degree project, every layer in the portal, starting from the top, must be investigated before entering the next layer and towards the port in the bottom. The first main category is quantitative or qualitative character. From this category, the first layer, to consider, is the outermost layer "Philosophical assumptions". Then, the selection moves inwards to the "Presentation" layer. At each level, all methods in the layer must be described, theoretically, before choosing a method and, then, applied to the research at hand. Hence, at least one method from each layer must be chosen, described and applied on the research project, or degree project, before moving to the next layer.

3.1 Quantitative and Qualitative research methods

The basic categories of research methods are, commonly, *Quantitative research method* and *Qualitative research method*. These two methods apply on projects that are either numerical or non-numerical. One of the research methods must to be selected, which decides whether the project is of quantitative character or qualitative character. This is the first choice of scientific standpoint and will affect the choice of research methods, strategies, data collection and analysis.

The method *Quantitative Research* method supports experiments and testing by measuring variables to verify or falsify theories and hypothesis, or computer systems' functionalities and interfaces. The formation of the hypothesis is that it has to be measureable with quantifications. Vague terms are unacceptable and the hypothesis must be evaluable and answerable. The method requires large data sets and use statistics to test the hypothesis and make the research project valid.

The *Qualitative Research* method concerns understanding meanings, opinions and behaviors to reach tentative hypotheses and theories or develop computer systems, artifacts and inventions. The method commonly uses smaller data sets that are sufficient enough to reach reliable results, where the data collection continues until saturation is reached.

It is possible to use both research methods as complement to get a complete view of the research area and situation. This method, called *triangulation*, is often used to ensure the correctness of results by increase credibility and validity of the results. Although the usage of several methods, triangulation is often carried out by applying one method at the time.

3.2 Philosophical assumptions

A philosophical assumption is the starting point for the research. The philosophical assumption is essential since it affects and steers the whole research. Hence, it constitutes assumptions about valid research and appropriate research methods and is the stand-point, which becomes the point of view for the project.

There are several core assumptions, or paradigms: Positivism, Realism, Interpretivism, and Criticalism [13; 11].

Positivism (objective) [13; 11] assumes that the reality is objectively given and independent of the observer and instruments. The researchers test theories, usually in a deductive manner, to increase predictive understanding of a phenomenon. The view of positivists is used in projects that are of experimental and testing character. They dismiss or evince a phenomenon by drawing inferences, about the phenomenon from the sample to a stated population, quantifying measures of variables, and testing hypothesis. The positivist assumption works well for testing performances within information and communication technology. *Post-positivist* is similar to positivist but post-positivists believe that the researchers' experience, knowledge and values can influence the observations, which makes it lean towards sociology and, hence, interpretation.

Realism (realistic) [13] assumes that things, in the reality, are known, or are of a perceived existence or nature. They do not depend on a person who is thinking about the things or is perceiving the things. The realists observe phenomenon to provide credible data and facts. From the collected data, the realists work with understanding the data and developing knowledge. The result is regarded as accurate worldly experiences (you see what you get) or as sensations. Realism can be used in interdisciplinary research in information and communication.

Interpretivism (interpretative) [13; 11] assumes that the reality is accessed only via social constructions. They attempt to understand phenomena by exploring richness, depth and complexity, often in an inductive manner, to discover the meanings people assign to the phenomenon. The view of interpretivists is used in projects with opinions, perspectives, and experiences characters to get context for phenomena. Works well in developing computer systems and artifacts.

Criticalism (critical) [13; 11] assumes that the reality is socially, historically, and culturally constituted, produced and reproduced by people. The critical assumptions focus on the oppositions, conflicts and contraditions in society and seeks to find and eliminate the causes of alienations, dominations and injustice and so on. The criticalism assumption can be used when learning about users' culture how it might affect the usages of computer systems.

From these assumptions, the researcher decides the perspective of the research, which is the approach taken to conduct the research, which are sometimes called assumptions or perspectives. Some examples of perspectives are: Pragmatism, Ontology, Epistemology, Axiology, Rhetoric, and Methodology.

Once the assumption is decided, the research question should be formed. From this question, the research method is chosen, which should provide a result (or results) for the given question.

3.3 Research Methods

Research methods provide procedures for accomplishing research tasks. These procedures are how to do the research with initiating, carrying out and completing research tasks. Hence, the research methods are the methods applied to the research project or degree project to support the process of conducting research.

In the portal, research methods and the research strategies and designs are distinguished. The distinction is the research methods are the framework for the research, which is theoretical frame for the methods. The research strategies and designs are the guidelines or methodologies, which are the steps carried out to conduct the research. The research method is, hence, on a higher level than the research strategies and designs.

Some of the research methods are also research strategies and designs, but can be described at different levels of details. This can be carried out by letting the research methods, first, introduce the method and, then, employ the research strategies and designs as methodologies that are applied to reach results.

The most common research methods are: Experimental or Quantitative, and Non-experimental including Descriptive, Qualitative, Analytical, Fundamental/Basic, Applied, Conceptual and Empirical.

Experimental research method studies causes and effects [1; 13]. The method deals with variables, establishes relationships between variables and finds causalities between the relationships. It can manipulate one variable and keep the other variables constant and check how the changes affect the result. This research method is often used when investigating systems' performances.

The *Non-experimental research* method examines existing scenarios and draws conclusions for the situation [1]. The method describes or predicts behaviour or opinions and can also describe relationships between variables but does not set out or test causalities between variables. This method has three different designs: relational (or correlational), comparative and longitudinal research and study. It can be used for studying users' behaviour or opinions of functionalities and interfaces.

The *Descriptive Research* method, also called statistical research, studies phenomenon and describes characteristics for a situation but not the causes or occurrences for the situation. The descriptive research method can use either quantitative or qualitative methods and often uses surveys, case studies and observations to produce accurate representations of persons, events or situations [6]. The method focuses on finding facts to establish the nature of something as it exists and can be used to find new characteristics, meanings and/or relationships in already existing data. It can be used for all kinds of research or investigations in computer science genre that aim to describing phenomenon or characteristics.

Analytical Research method tests pre-planned hypotheses based on existing knowledge and findings. It is concerned with validating hypotheses and uses facts and information that is already collected, and analyses the material to make a critical evaluation of the material. The method aids in decision making in areas, such as, product design and process design.

Fundamental Research, also called basic research or pure research, is a basic or applied research with a "curiosity-driven" focus observing a phenomenon to get new insights into the essence of nature. The method focuses on fundamental principles and testing theories. It generates new ideas, principles and theories. The result is innovations and new developments and solutions to old problems. This allows existing theories to be challenged and new theories to be developed. The fundamental research method is used for all kinds of research or investigations to generate innovations, principles and theories.

Applied research method involves answering specific questions or solving known and practical problems. The method examines a set of circumstances and the results are related to a particular situation. It often builds on existing research and uses data directly from the real work and applies it to solve problems and develop practical applications, technologies and interventions. Applied research is used for all kinds of research or investigations, which is often based on basic research and with a particular application in mind.

Conceptual Research method is used for developing new concepts or interpreting existing concepts [7]. It concerns theory development, historical research, literature reviews, and critical analysis and can be used to establish concepts in an area. The conceptual research can be used for investigating contents in a system with or without historic involvement. The literature reviews should not be confused with background data. Instead it is to study the literature to find, analyse and interpreting commonly used concepts.

Empirical research method relies on experiences and observations and derives knowledge from actual experience and tests predictions by focusing on real people and situations. It forms a body of knowledge and well-formed theories by involving collection and analysis of data and experiences to characterize, evaluate and reveal relationships between deliverables, practices and technologies. The empirical research seeks to gain knowledge by getting proofs based on evidence from experiments, observations or experiences. From the observable evidences that are collected, data are analysed with either quantitative or qualitative methods to explain the intrinsic situations.

Other important methods to be mentioned but not described in details in this paper are: Quasi-Experimental Research [13], Correlational Research and Historical Research [1].

If a single research method is not comprehensive enough or considered to be too weak to provide desired outcomes or results, several research methods, that complement each other, can be selected and applied to give a comprehensive picture of the situation or convincing and reliable research results.

3.4 Research Approaches

Research approaches are used for drawing conclusions and establishing what is true or false. The most common are inductive and deductive [15] but there is also a mixed approach, called abductive. The inductive approach establishes a general proposition from particular facts and the deductive approach derives conclusions from known premises. Abductive derives likely conclusions from an incomplete set of observations.

Inductive approach [13; 15; 4] or reasoning formulates theories and propositions with alternative explanations from observations and patterns. It can also be used in the development of an artifact. Data is collected, commonly with *qualitative methods*, and analysed to gain an understanding of phenomenon and establishing different views of the phenomenon. The outcome is based on behaviours, opinions, and experiences and must contain *enough* data to establish why something is happening, which are the reasons for the theories or requirements for an artifact.

Deductive approach [13; 15; 4] or reasoning tests theories to verify or falsify hypotheses. These theories are rigorously tested by deducing and testing a theory and hypothesis using, almost always, *quantitative methods* with large data sets. The hypothesis must be expressed in operational and measurable terms, explaining how the variables are to be measured and expressing the expected outcome to be achieved. The outcome is a generalisation that *must* be based on the collected data, and explanations of causal relationships between variables, establishing what is happening.

Abductive approach or reasoning uses both deductive and inductive approaches to establish conclusions. In the method, the hypothesis, that best explain the relevant evidence, is chosen. The approach starts with an incomplete set of data or observations and uses preconditions to infer or explain conclusions. The outcome is a likely or possible explanation and is, hence, useful as heuristics.

3.5 Research Strategies / Designs (Methodologies)

The research strategies and designs are the guidelines, or the methodologies, for carrying out the research. The guidelines (methodologies) are for the research, which includes organizing, planning, designing and conducting research.

The research strategies and designs for quantitative research commonly are Experimental Research, Ex post facto Research, Surveys (Longitudinal and Cross-sectional), and Case Study, and for qualitative research, the strategies and designs commonly are Surveys, Case Study, Action Research, Exploratory Research, Grounded theory, and Ethnography.

Experimental research strategy and design concerns control over all factors that may affect the results of an experiment. As the research method, Experimental strategy/ design verifies or falsifies hypotheses and provides cause-and-effect relationships between variables, i.e., correlations between independent and dependent variables. The amount of collected raw data is often enormous and the analysis is carried out with statistics. The method is used in experiments with large data sets.

Ex post facto research is similar to experimental research but does not control or change the independent variable since it is carried out after the data is already collected. Ex post facto = after the fact, means that it searches back in time to find plausible causal factors. The method also verifies or falsifies hypotheses and provides cause-and-effect relationships between variables but cannot provide safeguards to make strong inferences [2]. The method can be used to study behaviours and can, therefore, also use qualitative methods.

Surveys, cross-sectional and longitudinal, assess attitudes and characteristics of a wide range of subjects. It is a descriptive research method, which examines the frequency and relationships between variables [13] and describes phenomenon that are not directly observed. Cross-sectional surveys collect information on a population, at a single point of time. Longitudinal surveys collect data over a period of time. Due to surveys' characteristics and using questionnaires, the method can be used with quantitative and qualitative methods.

Case study is an empirical study that investigates a phenomenon in a real life context where boundaries between phenomenon and context are not clearly evident. The case study is a strategy, which involves an empirical investigation of a particular phenomenon using multiple sources of evidence. Case studies can be based on quantitative and qualitative evidences or of a mix of these methods and, hence, use either method or both methods.

Action research is performed by actions to contribute to practical concerns in a problematic situation. The method improves the way people address issues and solves problems, as well as, strategies, practices, and knowledge of environments. It is a systematic, cyclic method of planning, taking action, observing, evaluating and critical reflection. Action research often studies communities or settings with restricted data sets and, hence, qualitative methods are most suitable.

Exploratory research method provides a basis for general findings by exploring the possibility to obtain as many relationships between different variables as possible. It uses surveys to get an insight in the problem. It rarely provides definite answers to specific issues. Instead, it identifies key issues and variables to define objectives, using qualitative data collection.

Grounded theory seeks to develop a theory that is grounded in data. The method systematically collects and analyses data. The grounded theory is an inductive theory discovery method that allows the development of a theoretical account of general features of a topic.

Ethnography method is from anthropology, which means portrait of people or methodology to study people. It uses descriptive studies of culture and people that under investigation have something in common. Ethnography seeks to place the phenomena in their social and cultural context.

Other methods, not shown in the portal, is Creative Research and Design science. Creative Research method involves developing new theories, procedures and inventions, such as, artifacts or systems [8]. Design science is not in the portal since it is a framework for developing artifacts in the area of information systems and not a scientific research method. However, if design science is considered as a method, it should be noticed that designing an artifact is not science itself and using design science as a research strategy can create discussions. Also, design science includes parts of software engineering methods used when designing systems.

3.6 Data Collection Methods

Data collection methods are used to collect data for the research. The most commonly used methods for quantative research are Experiments, Questionnaire Case Study, and Observations; for qualitative research, Questionnaire, Case Study, Observations, Interviews, and Language and Text.

The data collection methods use samplings. Commonly, the quantative uses probability with stratification and theoretical samlings. Qualitative uses concenience, snowball, probability with stratification and theoretical samlings.

The different methods collect data for different purposes:
- *Experiments* collects a large data set for variables.
- *Questionnaire* collects data through questions, which are either quantifying data (closed, alternative questions) or qualifying data (open and reviewing questions).
- *Case Study* is an in-depth analysis of a single or small number of participants. The case study data colletion method is used with the case study research method.
- *Observations*, ethnograpy and participation, observe behaviour with focus on situations (participation) and culture (ethnograpy).
- *Interviews*, structured, semi-structured, and unstructured, give an deep understanding of a problem and capture participants' point of view.
- *Language and Text* are used for interpreting discourse and conversations, and meanings in texts and documents

3.7 Data Analysis Methods

The data analysis methods are used to analyse the collected material. It is the process of inspecting, cleaning, transforming and modelling data. It supports decision-making and drawing conclusions.

The most commonly used data analysis methods for quantiative research are Statistics, and Computational Mathematics and for qualitative research, the most commonly used are Coding, Analytic Induction, Grounded Theory, Narrative Analysis, Hermeneutic, and Semiotic.
- *Statistics*, descriptive and inferenctial statistics, is used to analyse data [13]. This includes calculating results for a population (or sample), as well as, evaluating the significance of the results.
- *Computational Mathematics* is used for calculating numerical methods, modelling and simlauations with an emphasize on algorithms, numerical methods, and symbolic methods.
- *Coding* analyses transcriptions of interviews and observations by turning qualitative data into quantitative data. Coding is naming and labelling concepts and strategies to numerate these concepts and strategies and apply statistics.
- *Analytic Induction* and *Grounded Theory* are iterative methods, that alternate between collections and analyses. The iterations continue until no cases dismiss the hypothesis or theory. Analytic induction stops when the hypothesis and grounded theory ends with a validated theory.
- *Narrative Analysis* concerns literary discussion and analysis. *Hermeneutic* (meaning of text/text-analogue), and *Semiotic* (meaning of signs and symbols) are used for analysing texts and documents and can be used to support traceability in requirements and interfaces.

3.8 Quality Assurance

Quality assurance is the validation and verification of the research material. The quantiative research, with a deductive approach, must apply and discuss validity, reliability, replicability and ethics [13; 14]. The qualitative research, with an inductive approach, must apply and discuss validity, dependability, confirmability, transferability and ethics [11]. Shortly, these terms mean:
- *validity*, in quantative research, makes sure that the test instruments actually are measuring what is expected to be mesured [7; 14].

- *reliability* refers to stability of the measurements [7] and is the consistency of the results for every testing.
- *replicability* is the possibiltiy, by another researcher, to repeat the same research and, hence, reach the same results [4]. This requires well-described procedures to be used for the research.
- *ethics*, independent of quantative research or qualitative research, is the moral principles in planning, conducting and reporting results of research studies [11]. Ethics covers protection of participants, maintenance of privacy, avoiding coercion and having consent in written form, and treating material with confidentiality [13].
- *validity*, or trustworthiness, in qualitative research, makes sure that the research has been conducted according to existing rules [4]. Respondents can validate and confirm that the results are correctly understood.
- *dependability*, corresponds to reliability, and is the process of judging the correctness in conclusions, using auditing [11].
- *confirmability* confirms that the research has been performed in good faith without personal assessments that have affected the results [11].
- *transferability* is to create rich descriptions that can become a database for other researchers [11].

3.9 Contributions and results

The contributions and results present the whole research. The presentation of the research is a rigorous description of the theory of the research methods and methodologies. From this theoretic description, methods and methodologies chosen and applied to the research project. In the best of all worlds, the philosophical assumptions, the research methods and the research approaches are introduced and discussed early in the thesis. Then, the appliction of the research method and research strategies/designs are separately presented discussing pros en cons for each method and strategy/design. Data collection should also be presented separately, as well as, data analysis and quality assurance. The most important issue is to be fair to the collected material and not present conclusions, theories or artifacts that lack of foundations or evidences in the research work.

4 Conclusions and discussion

This paper provides a portal of research methods and methodologies that can be used to support the students to choose and apply the most suitable methods by showing the usages and, to some extent, the distinctions between the different methods. Although the paper distinguishes between methods, some of the qualitative methods can be used in quantative research and vice versa but, then, consequences of using the different methods need to be considered.

There are a lot of research methods and methodologies. Since, it is hard to cover all of them, the next step in the work, is to tune the portal and the description of the research methods and methodologies to better apply to the research in information and communication technology. It is also to give a better description of the processes and how different methods and methdologies, at different levels, complement each other better than others.

References

[1] Donald Ary, Lucy Cheser Jacobs, Chris Sorensen, and Asghar Razavieh. Introduction to Research in Education. Wadsworh, Cengage Learning, 8th edition, 2010.

[2] Alan Bond. “Your Masters Thesis How to Plan, Draft, Write and Revise” Studymates Ltd, UK, 2006.

[3] Wayne Booth, Joseph M. Williams, Gregory G. Colomb. “The Craft of Research, 2nd edition, Guides to Writing, Editing and publishing.” University Of Chicago Press; 1 edition, 2003.

[4] Alan Bryman, Emma Bell, Business Research Methods, Second edition. Oxford University Press, Inc. New York, 2007.

[5] Dewey and Dewey. Decimal Classification, “A Research Guide for students“. Research, Writing, and Style Guides. MLA, APA, Chicago/Turabian, Harvard, CGOS, CBE), OCLC Online Computer Library Center, Inc, 1998. http://www.aresearchguide.com/styleguides.html

[6] Barry J. Fraser, Kenneth G. Tobin and Campbell J., McRobbie. Second International Handbook of Science Education. Springer Science, 2012.

[7] Pervez N. Ghauri, and Kjell Grønhaug. Research Methods in Business Studies (4th Edition), London: FT Pearson, 2010.

[8] Wayne Goddard and Stuart Melville. Research Methodology: An Introduction, Juta Academic, 2nd edition, 2004.

[9] Kim Kastens, Stephanie Pfirman, Martin Stute, Bill Hahn, Dallas Abbott, and Chris Scholz. “How to Write Your Thesis“, 2009. http://www.ldeo.columbia.edu/~martins/sen_sem/thesis_org.html

[10] Rowena Murray. “How to Write a Thesis.” Open University Press, 2006.

[11] Michael Myers. Qualitative research in Business and Management. SAGE Publication Inc. London, UK. 2009.

[12] Isadore Newman and Carolyn R. Benz. Qualitative-Quantitative Methodology: Exploring the interactive Continuum, 1998.

[13] Neil J. Salkind. Exploring research, 6th edition, Pearson International Edition. 2006.

[14] Mark Saunders, Philip Lewis and Adrian Thornhill, Research methods for business students. 5th edition, 2006.

[15] William Trochim, and James P. Donnelly. “The Research Methods Knowledge Base”, 3rd Edition, 2007.

[16] Gina Wisker. The Postgraduate Research Handbook. Methodology. 2nd edition. Palgrave Macmillian, 2007.

[17] Justin Zobel. “Writing for computer science”. Springer; 2nd edition, April 27, 2004.

Investigating Female Students' Attitude towards Cheating and Plagiarism: A Study in King Saud University

Manar Hosny and Shameem Fatima
mifawzi@ksu.edu.sa, sfatima@ksu.edu.sa
College of Computer and Information Sciences,
King Saud University, Riyadh, Saudi Arabia

Abstract - *: During their studies, many students commit some form of academic dishonesty, such as cheating and plagiarism, often to obtain higher grades than they are capable of. The current widespread use of the Internet, mobile and wireless devices has made it easier for students to illegally access information, and at the same time it has become difficult for academic institutions to control and discover such instances. Hence, it is essential that students become aware of the seriousness of these offences, and be encouraged to avoid them. In this paper, we study the attitude towards cheating and plagiarism among female students in the College of Computer and Information Sciences (CCIS) at King Saud University, Riyadh, Saudi Arabia. We aim to highlight the most prevailing practices, the underlying reasons, the popular sources of illegal information, and the conception of students towards the ethicalness of such practices. The results of the study indicate that both cheating and plagiarism are common among our students. After having analyzed the results, we propose some recommendations that may help combat cheating and plagiarism among students in higher education.*

Keywords: cheating, plagiarism, academic dishonesty, ethics, ethics in information technology

1 Introduction

Academic dishonesty at universities is a common phenomenon among students of all ages and specialties. Nowadays, the widespread use of the Internet and the popularity of mobile and wireless devices have made it easier for students to reach and transmit information in illegal and dishonest ways. Academic dishonesty can be defined as the students' use of illegal activities, techniques, and forms of fraud during their examination or evaluation processes, usually for the purpose of achieving better grades [1].

In the literature many types of academic dishonesty have been observed. For example, the collaboration when doing assignments, completely or partially copying an assignment from another student, using the Internet as a source for help for solving difficult problems, submitting the same work for multiple courses, copying text from another source (book, Internet, etc.), paying someone to do an assignment, using hidden resources during an exam, and many other forms [2].

In general, academic dishonesty can be divided into three main categories: cheating, plagiarism and collusion [3]. The first two categories seem to be the most common among students, especially those of younger ages. In general cheating is considered as an intended violation of rules in order to acquire illegal advantage or better academic results in exams or similar forms of assessment. This may happen by 'stealing' ideas and other material from different sources [4], [5]. On the other hand, plagiarism involves passing off someone else's work as your own without acknowledging the source [6]. Thus, plagiarism may or may not be intentional, due to some students' lack of knowledge regarding relevant standards of quoting. Hence why, plagiarism may not always be considered as cheating.

An increasing number of incidents of cheating and plagiarism are being observed daily. One reason could be that electronic communication through handheld and other popular devices makes it even easier for students to copy and transmit information both inside and outside the classroom. While doing so, students do not think much about the legality of this action, since improving their grades and passing the course come up as their only concern and ultimate goal that they hope to achieve. Although getting better grades may seem as the most compelling factor for making students cheat or plagiarize, other less obvious reasons could be: peer pressure, playing smart, making fun of the instructor, or just because they can!

The academic community is currently giving a lot of attention to increasing the awareness of students about ethical issues, including ethics of copying and using information in the electronic age. Publishing ethical codes of practice and teaching ethics courses have become an essential part of almost every discipline in the university, which indeed can help in fighting academic dishonesty prevailing among students. In addition, studying students' attitudes, what they think about academic dishonesty, and why they practice it may help overcome this phenomenon, while increasing the students' -as well as the faculty members'- consciousness about the danger and consequences of such acts.

In our department, which is a female only department in King Saud University, the phenomena of cheating and plagiarism are rather noticeable. This is despite efforts done by faculty members to combat cheating, and severely punishing students who are caught with academic misconduct. It would be interesting to know the extent of the spread of academic dishonesty among students, and the reasons behind committing it. It is also important to understand the attitude of students towards cheating and plagiarism, and what their conception is about their ethicalness. This may help in targeting the reasons behind the spread of these actions and could increase the awareness of students about the danger and consequences involved thereof. Overcoming academic dishonesty is essential in preparing students for a promising and successful professional future.

To this end, we conducted a study among the female students in our department. A survey about cheating practices and another about plagiarism practices have been distributed to students from different levels. This is the first time this type of study has been conducted in our college. In this paper we will analyze the results obtained by those surveys to determine the extent of the spread of academic dishonesty. We try to propose some recommendations about how to confront the phenomena of cheating and plagiarism, and to increase the awareness about them among our students.

2 Research Methodology

In the current research, two surveys were used to assess female students' attitude towards cheating and plagiarism in our college, where female students are segregated from male students. The Arabic language was used to write both surveys, since almost all students are native Arabic speakers and they are usually more comfortable using Arabic than English.

The cheating survey consists of three parts: the first part is composed of six questions intended to assess the frequency of some cheating practices among students. The second part consists of seven possible reasons for committing cheating and is intended to survey the students' opinion about the relevance of these reasons to practicing cheating. The third part consists of one question asking the student about her opinion regarding whether the cheating practices described are ethical or not, then giving the reason for her answer.

Similarly, the plagiarism survey consists of three parts: the first part tries to determine if the student knows the meaning of plagiarism, besides assessing the frequency of practicing some common plagiarism acts (three practices were included). The second part tries to examine the most common sources of plagiarism. The third part asks the student about her opinion regarding the ethicalness of such practices and giving the reason for her answer.

The target sample for the surveys was undergraduate students in levels 3 to 6 (a level is equivalent to one semester; so level 1 is the first semester of the student in the college, level 2 is the second semester, etc.). 148 students anonymously answered the cheating survey, and 115 students answered the plagiarism survey.

2.1 Surveys Results

Table 1: Assessing the frequency of some cheating practices

	Copy from another student	Pressured by a colleague to give answers	Use of body parts	Extract hints/answers from instructor	Paid someone for doing assignment	Changed answer after grading
%Frequently	0.0	3.3	1.3	2.0	0.6	0.0
%Sometimes	4.7	18.9	0.6	22.9	7.4	4.7
%Rarely	10.8	32.4	4.0	24.3	13.5	4.0
%Never	84.4	45.2	93.9	50.6	78.3	91.2

The results of part 1 of the cheating survey, shown in Table 1, indicate that 15.5% of students have previously copied answers from others during an exam. 54.6% have been pressured at sometimes to give answers during an exam. 5.9% of students have previously used body parts to hide written answers for the purpose of cheating during an exam. 49.2% of students have previously resorted to extracting hints from instructors during an exam. 21.5% have previously paid someone to do an assignment for them, and 8.7% of students have previously changed answers after grading and claimed an error in the marking of an exam.

Table 2: Reasons for cheating

	Difficulty of the exam	Not ready for the exam	Lack of time to study	Lenience of instructors	Having fun/ playing smart	Helpin g a friend	Importanc e of grades
%Agree	31.5	30.4	36.9	9.7	7.6	14.1	31.5
%Neutral	30.4	29.3	30.4	35.8	17.3	30.4	19.5
%Disagree	38.0	40.2	32.6	54.3	75.0	55.4	48.9

Among the 145 students who answered the cheating survey, 92 students answered Part 2[1]. The results of Part 2, shown in Table 2, indicate that 31.5% of students agree that difficulty of the exam is one possible reason to cheat. 30.4% agreed that not preparing enough for the exam made them cheat. 36.9% have agreed that lack of time to study is a reason of cheating. 9.7% agreed that having a lenient instructor encouraged them to cheat. 7.6% did it because they were

[1] Those who answered "Never" to all questions of Part 1 did not answer Part 2.

playing smart or because they had fun doing it. 14.1% thought that helping a friend was one reason to cheat, and 31.5% agreed that importance of grades justified cheating. Finally, 11.5% students thought that cheating was ethical.

Table 3: Meaning and common plagiarism acts

	Know what plagiarism means	Submitting another person's work	Replacing words without citing	Using exact words
%Yes	72.1	11.3	32.1	40
%No	11.3	66.0	44.3	41.7
%Uncertain	16.5	22.6	23.4	18.2

As shown in Table 3, 72.1% of students indicated that they know what plagiarism means. 11.3% of students have previously copied all or parts of another person's work and submitted it as their own without citing the source. 32.1% have replaced the words of someone else with their own words without citing the source. 40% of students have used the exact words of someone else without quotations and without citing the source.

Table 4: Common sources of plagiarism

	Electronic Resources	Printed Resources	Ideas/work of others
%Frequently	39.1	20	9.5
%Sometimes	44.3	42.6	41.7
%Rarely	10.4	27.8	26.9
%Never	6.0	9.5	21.7

As shown in Table 4, 83.4% of students have frequently or occasionally used electronic resources as a source of plagiarism. 62.6% have used printed resources, and 51.2% have used ideas and/or work of others like parents, colleagues, etc. as a source of plagiarism during their undergraduate studies. When asked whether plagiarism is ethical, 34.75% of undergraduate students believed that it is ethical!

3 Discussion

From the results of the cheating survey we can conclude that the majority of students (approximately 85%) have never copied answers from someone else during the exam, whereas a large percentage of them (approximately 55%) were pressured to give answers to someone during the exam. Most students did not resort to using body parts to hide written answers during an exam, and the majority also did not try to change the answer after grading. On the other hand, extracting hints from the instructor during the exam was relatively common, where approximately 50% of students had practiced it. Strangely enough paying someone to do an assignment seems to be a common practice among our students, where approximately 22% of students have admitted committing this act. This is surprising because in our community and especially among female students, getting access to professionals (a private tutor or a software company for example) who are paid to do a difficult homework like a project or a research paper is usually infeasible without the consent of a parent or a guardian. Thus, it seems possible to assume that parents themselves do not find this practice unacceptable!

Regarding the reasons for cheating, the most relevant reason for cheating is the lack of time to study, followed by the difficulty of the exam, importance of good grades, and not preparing for the exam (in order). On the other hand, helping a friend, having a lenient instructor, and playing smart, were less important factors that can lead to cheating.

When asked about the reason, 65% of students who believe cheating is unethical indicated that cheating is forbidden in Islam, and against Islamic values. Most of them cited the Hadith (saying) of Prophet Mohammed *"He who cheats us does not belong to us"*. 14% indicated that cheating gives one what is not right for them. 4% said it is not honest, and another 4% said that the benefit they get from cheating is "not blessed". Similarly 4% said that they are afraid of Allah (God). Other less frequent responses included: "who cheats can do more serious offences"; "it is like stealing"; "being honest is essential for the development of the society"; "a student who cheats is irresponsible"; and "because exams are intended to assess your understanding of the subject".

On the other hand, when asked about whether cheating is ethical, approximately 12% thought it is ethical! Among the reasons given for being ethical are: "I am not very religious"; "it depends on your values"; "I have to do it because the distribution of grades is not fair"; and "I have to do it to succeed".

From the results of the plagiarism survey, we can see that the majority of students know what plagiarism means. The most common practice was to copy exact words without citing the source. Paraphrasing without citing the source is another common practice. On the other hand, submitting someone else's work (an assignment for example) as their own is not a very common practice among students.

Regarding the source of plagiarism, most students agreed that electronic resources are the most popular source of plagiarism followed by printed resources. Slightly less common is the use of work or ideas of someone known to them like parents or colleagues.

A large percentage of students (approximately 35%) indicated that the act of plagiarism is ethical. However, most of those who said that did not give a relevant reason for believing so. Instead, they said that if the source is mentioned or it is an open source it is “okay” to use it. Some of those who believed it is ethical mentioned reasons like: “to help me”; “it depends on your intentions”; “it doesn't hurt to make use of some text in my research if I write it in my own way”; “I may have to do it because of lack of time”; “I am just using it without mentioning it is mine”; and one person said “I don’t know”!

On the other hand, 81% of those who believed it is unethical said that plagiarism is like stealing the work of others; 16% said that it violates intellectual property rights; and 3% said that it is against Islam. Some of the reasons given for being unethical are: “everyone should be given their right”; “it is a form of stealing/cheating”; and “it does not help me learn”.

To sum up, it appears from the results observed from the cheating survey that our students generally practice cheating by giving answers during an exam if pressured to do so by someone. They frequently resort to extracting hints from the instructor during the exam, and they sometimes pay money to get their homework done. As a result, we recommend that teachers be vigilant during the exam and avoid answering unnecessary questions by students. In fact, it is advisable to prohibit answering questions during the exam and to discourage the proctoring of the exam by the instructor of the course.

Regarding paying someone to do the homework, it is advisable that the instructor tries to get more acquainted with the students to recognize their individual levels. Also, having assignments of incremental nature, where every part is submitted separately, may help discourage this phenomenon. More interesting assignments that need creativity and discussion after submission may also help combat the students’ tendency to rely on others.

It can also be observed that importance of grades, competition with peers and expectations of parents are among the factors that can lead to cheating. Trying to motivate the students to learn and giving rewards that are not dependent on grades may help reverse the effect of these factors.

To increase the awareness of students, teachers as well as parents should stress the fact that cheating is unethical, not only because it is against religion, but because it is a form of fraud that makes someone gets what is not right for them, and it is harmful for the society. Policies against cheating and relevant punishments should be made clear at the introduction of every course, and implemented seriously by all faculty members.

From the results of the plagiarism survey, it seems that students think that it is less serious than cheating during an exam. This can be deduced from the lower percentage of correlating plagiarism with being ‘anti-religion’ compared to cheating. Most probably students think it is less serious because it is not directly done during the exam, or because it is more difficult to discover and the source of information is mostly unknown. Also, cheating by nature is often more stressed upon in the pre-college education than plagiarism. Therefore, students may be inclined to think that cheating is more serious than plagiarism.

We also believe that weakness in the English language, which is the language of instruction in our college, and the inability of students to express their thoughts properly in English, can lead to plagiarism. The most common forms of plagiarism are copying exact words and paraphrasing without citing the source. In addition, electronic resources stand out as the most popular source of plagiarism.

To combat plagiarism, students should be made aware of the seriousness of this act, and that it can affect not only their academic studies but also their future careers. Students should be warned about legal issues that may be involved with violating intellectual property rights. To help students combat their tendency to plagiarize, proper citation and referencing methods should be taught. Students should also try to improve their language and writing skills. They should practice quoting and paraphrasing techniques. Introductory research methods and writing courses should be given to all students starting from the undergraduate level. Plagiarism detection software should be used and the students can be advised to try them and review their essays to get rid of plagiarism before submission.

4 Conclusions and Future Directions

Motivated by the spread of practicing cheating and plagiarism among our students, we surveyed a sample from female students in different college levels about cheating and plagiarism practices. The surveys focused on the most prevalent practices, the reasons for doing them, and the awareness of students about the ethicalness of such acts.

The results indicated that, despite being aware of the immorality of these practices, students sometimes resort to cheating by taking or giving answers during the exams. However, a more common practice was to pay someone to do the homework on their behalf. This seems to be a serious offence that should be given more attention and severely punished by both the administration and faculty members.

Plagiarism practices seem to be even more common among students. It appears to be less serious in their view than cheating during exams. Increasing the awareness of students about the seriousness of this practice is essential. Moreover, helping them by improving their language and writing skills and teaching proper referencing, quoting, paraphrasing and citation styles are also important to discourage this phenomenon.

In the future we intend to examine whether the level of study is related to patterns and frequency of cheating and plagiarism. For example, it would be interesting to know whether students at introductory courses cheat more or less than students who are approaching graduation. Whether there is an effect of gender on the tendency and prevalence of cheating is also worth studying.

5 References

[1] Uhráková, E. & Podařil, M., The Attitude Of Students Towards Electronic And Non-Electronic Cheating" in proceeding of International conference the future of education. Retrieved on 21 Feb 2013 from www.pixel-online.net/edu.../ENT20-Uhrakova.pdf.

[2] Sheard, J., Markham, S., & Dick, M. Investigating differences in cheating behaviours of IT undergraduate and graduate students: The maturity and motivation factors. Higher Education Research and Development, 22(1),2003, 91-108.

[3] Moon, J. Academic Honesty, Plagiarism and Cheating: a self-instruction unit for postgraduate students.
(2006), Retrieved on 23 Feb 2013 from
http://www.hud.ac.uk/hhs/teaching_learning/plagiarism_handout3.pdf.

[4] Mareš, J. Tradiční a netradiční podvádění ve škole. (Traditional and non-traditional school cheating). Pedagogika. LV, 2005, No 4 (pp. 310-334). (2005). ISSN 0031-3815.

[5] Dobrovska, D., & Pokorný, A. Avoiding Plagiarism and Collusion. In *International Conference On Engineering Education-ICEE, Coimbra, Portugal.* (2007).

[6] Carroll, J. *A handbook for deterring plagiarism in higher education*. (2002). Oxford Centre for Staff and Learning Development.

Attendance Tracking

D. Deugo[1]
[1]The School of Computer Science, Carleton University, Ottawa, Ontario, Canada

Abstract - *We believe that a student's attendance in their university courses is important for the successful completion of their courses and that student attendance is a marker one can use to identify students in need. The problem this paper addresses is how to track student attendance in university courses in a fast and efficient manner, given class sizes can be as small as 10 students and as large as 300. Our approach uses easily found, inexpensive hardware and makes use of students' smartphones to help with the attendance tracking process. Our applications for the server and client portions of our system use open source software to minimize development and maintenance costs and do not require end users or system administrators to perform any installation.*

Keywords: Attendance, Tracking, Smartphone, Web

1 Introduction

The Science Student Success Centre (SSSC) at Carleton University is made up of students, staff, and faculty who know what students need to succeed in courses in the Faculty of Science. We collaborate with other on-campus departments to offer workshops, events, and activities that help to develop students' academic and professional skills. The SSSC also takes an active role in helping students struggling in their first-year computer science, math, and science courses. Its mission is as follows:

- Increase the engagement and retention of students in the Faculty of Science
- Foster the growth and achievements of high performing students
- Identify and support students who may be experiencing difficulty with their studies
- Inform students of professional and academic development opportunities
- Aid in the recruitment of outstanding students to the Faculty of Science

One of the methods the SSSC uses to identify first-year students in need is to look at their mid-term marks in their first-term science courses in mid to late October. Those students with marks less than 60% are sent an email and asked to come in and visit with the SSSC team to talk about the issues they are facing with their courses and possible techniques and actions they can take to overcome the issues.

The SSSC believes that a student's attendance in their university courses is important for the successful completion of the courses. We also believe that student attendance is an important marker that can be used to identify students in need. It is not that attendance is mandatory, but that students falling below a given level of attendance should be asked to meet with the SSSC team to talk about the reasons behind their low attendance and support given to help correct any issues the students are having.

The problem this paper addresses is how to track student attendance in university courses in a fast and efficient manner, given class sizes can be as small as 10 students and as large as 300. Taking attendance using paper and pen is one approach, but it is slow and prone to errors. In addition this method must go through a data entry phase in order to generate reports, which also suffers from the same problems. Therefore, using pen and paper is not an option.

1.1 Goal

Our main goal is to provide a fast and efficient attendance tracking system. In addition, the system must work in any and all classrooms at Carleton University, including its electronic classrooms – those with computers and projectors – and those classrooms containing no computers. Finally, our goal is to provide a system that requires minimum hardware, and as a result is built and maintained at minimum cost.

1.2 Objectives

To meet the goals mentioned in the previous section, we have the following objectives:

- Use easily found, inexpensive hardware
- Make use of students' smartphones to help with the attendance tracking process
- Use open source software to minimize development and maintenance costs
- Installation, either for end users or system administrators, should be fast and simple
- Attendance reports should provide end users with the greatest flexibility for manipulating the collected attendance data
- Support both Mac and PC platforms, and Android and iPhone smartphones

1.3 Outline

In section 2 we describe literature on existing attendance tracking systems. Section 3 describes our overall approach. Section 4 describes the current view of our system, including screen shots of the Web frontend of our system. Section 5 provides our conclusion and identifies our future work.

2 Background

Attendance tracking has been viewed as a method of improving student attendance and retention rates. Professor Mehmet Dicle and John Levendis at Loyola University in New Orleans created a computer program designed to electronically check students into class, eliminating the need for a lengthy roll call and saving valuable class time [1,2].

> *"The technology, called the DL-Electronic Attendance System, uses a quarter sized computer chip to track attendance by having individual students pass the chip in front of a scanner as they enter class. The scanner, hooked up to the professor's computer, records the students' attendance, eliminating the need for roll call."*

Dicle and Levendis' system is relatively inexpensive. The cost of the scanner is about $20 and is available online or in electronics stores. The chips that are scanned are also inexpensive, costing less than 30 cents each. The downside to this approach is that each student must be given a chip and the chip must be registered with the system.

Northern Arizona University (NAU) also implemented a similar attendance tracking system, based on RFID technology [3]. However, it was costly, with a price tag over $85,000, and created tension with the student body, raising the fear of invasion of privacy

> *"Northern Arizona University recently received $85,000 to fund a new tracking initiative that revolves around a series of scanners stationed outside lecture halls." [4]*

The RFID reader used by the NAU system, provided by HID (http://www.hidglobal.com/), cost $250 each, and the price did not include the cost of installing the devices and linking them to the school's network.

The Texas State University (TSU) has a Web-based attendance tracking system [5] that uses a student's ID card and a card reader to log attendance.

> *"Attendance tracking is a Web-based application that is designed to collect attendance data of individuals carrying a Texas State ID. By swiping one's ID card, the card reader logs the attendee as being in attendance."*

The cost of the card readers is minimal, only costing $50.00 in single unit purchases.

DeVry Univeristy [6] uses a system similar to TSU.

> *"A card reader is an automated system that collects class attendance by students swiping their DeVry ID cards for every scheduled lecture and lab session. The card readers are conveniently located in each classroom and lab."*

The University of Mississippi (UOM) uses a system similar in need to the SSSC at Carleton University.

> *"The Class Attendance Guidelines Policy discusses the importance of attendance relative to learning. Instructors are asked to report cases of freshman students missing three or more class periods as part of the Freshman Attendance-Based Intervention (FABI) program. The Academic Support Center follows up on these cases to connect students with resources. Yet, taking attendance for large classes can create an administrative burden for the instructor and also take away from critical class time."*

Their system uses the Symbol MK500, a small computer with a barcode reader and a network connection making it similar to what retail stores use for checking prices. As noted [7], the process for checking attendance is as follows:

- Students scan their student IDs by placing them under the scanner so that the ID barcode is readable
- The scanner reads the student ID information and sends it to SAP using the SAP Netweaver Gateway system.
- SAP processes the records to update class attendance based on the location, time, class, and student in near real-time
- The results are made available to the instructor using the new "Manage Attendance" option within the Class Rolls and Grades interface in myOleMiss

What was interesting about this system is the use of cell phones was considered, but ruled out.

> *"We considered having students check in with cell phones, but some instructors do not want cell phone usage to be part of the solution due to their potential to distract."*

In addition to attendance tracking system that are owned and operated by individual universities and colleges, open Web-based systems like myattendancetracker.com are also available to educators [8]. After creating an account, classes and students entries, users can start recording student attendance over the Web and generate Excel reports. The disadvantage of the system is the task of entering student information and the inability to quickly log student attendance using card readers or RFID tags based on student ID cards.

2.1 Summary

As described, there are different options available for attendance tracking system, each having a different hardware interfaces to where the student information resides. Hardware interfaces included RFID scanners, barcode scanners, card strip readers, and chip scanners. The choice of hardware depends on what students have available on their ID cards or if there is no ID card available. In some cases, the interface is the Web, and all data entry must be done manually. The cost of each system can vary greatly depending on the interface choice, and consideration must be given to the number of interfaces devices required by an institution. What is missing from the discussion is the time required to check in large classes using each of the approaches. For example, using 10 seconds as a check in time per student using a single reader, a class of 200 students requires 33.34 minutes to check in every student. In our case, this check in time is unacceptable for any class.

3 Approach

In the following sections we outline our approach. First we set the context for our approach, describing the environment that our attendance tracking system must operate in. The next section describes our database scheme. Our approach uses a combination of Web and smartphone logging and our database schema reflects this approach. The final section describes our implementation, interfaces and goes into a description of the system's features.

3.1 Context

At Carleton University, student ID cards have a magnetic strip on the back containing minimal information (no name or student id) but just enough to be able to access the student's record in the university database. The omission of name and id information is due to governmental Freedom of Information and Protection of Privacy Act regulations.

The university has Internet access in all classrooms. This access includes wireless connections for all students and faculty, and wired connections for instructors in electronic classrooms. Electronic classrooms include a locked computer and monitor, one or two LCD projectors, screen(s), chalkboards, and may include wired speakers. The classroom's electronic control system enable instructors to connect USB devices to the locked computer, or switch the LCD projector to connect to any laptop they decide to bring and operate in class.

3.2 Database Schema

Our database scheme contains four tables: Organization, User, Event, and Register. The Organization table is responsible from providing information on organizations, such as Carleton University, that want to track attendance. The Organization table contains the following columns:

- ORGANIZATION_PRIMARY_KEY: the primary key of the organization
- NAME: the text name of the organization

The User table is responsible for recording information about users of the attendance tracking system, such as instructors and administrators. The User tables contains the following columns:

- USER_PRIMARY_KEY: the primary key of the user.
- EMAIL: the email address of the user.
- FIRST_NAME: the first name of the user
- LAST_NAME: the last name of the user
- PASSWORD: the user's system password
- ORGANIZATION_FOREIGN_KEY: a foreign key reference back to the user's organization
- TYPE: the type of users, which is either instructor or administrator.

The Event table is responsible for recording the different events, such as a class or an exam, which an organization wants to track attendance. The Event table contains the following columns:

- EVENT_PRIMARY_KEY: the primary key of the Event
- USER_FOREIGN_KEY: a foreign key reference back to the user who will be recording attendance of the event
- NAME: the name of the event
- NUMBER: the number of the event
- YEAR: the year of the event
- DEVICE_ENTRY_TYPES: an or'd bit string indicating which hardware devices (Web, iPhone, Android) are permitted to log user attendance
- OTHER_INFORMATION: an open text field contain other information related to the event
- TERM: the term the event occurs in
- SECTION: the section of the event
- ORGANIZATION_FOREIGN_KEY: a foreign key reference back to the event's organization
- LATITUDE: the latitude of the event location

- LONGITUDE: the longitude of the event location
- ALTITUDE: the altitude of the event location
- ACTIVE: a boolean indicating if the event is active

The Register table is responsible for logging student attendance at an Event. The Register table contains the following columns:

- REGISTER_PRIMARY_KEY: the primary key of the Register
- USER_FOREIGN: a foreign key reference back to the user that recorded the attendance of a student
- EVENT_FOREIGN_KEY: a foreign key reference back to the event being logged
- DEVICE: the device used to log the attendance
- FIRST_NAME: the first name of the person attending the event
- LAST_NAME: the last name of the person attending the event
- PHONE_INFO: the phone information of the student attending the event
- STUDENT_NUMBER: the student ID of the student attending the event
- CARD_STRIP: the contents of the person's id card
- EVENT: a string representation of the event
- ORGANIZATION FOREIGN_KEY: a foreign key reference back to the user's organization
- ENTRY_TIMESTAMP: the time the attendance was logged.
- LATITUDE: the latitude of the were the log was taken
- LONGITUDE: the longitude of the were the log was taken
- ALTITUDE: the altitude of the were the log was taken
- ACCURACY: the accuracy of the were the log was taken, compared to the latitude, longitude and altitude of the event

3.3 Implementation, Interfaces and Features

3.3.1 Server

The server side of our application is a standard Web application setup. We use an Apache HTTP Server that communicates with a Tomcat application server when requests of the form srr.carleton.ca:8080/attendance/xxxx are received. xxxx is used to indicate what functionality of the attendance application is being requested. Our attendance server application is written in Java using Struts2 and Hibernate, the latter providing communication to our MySQL database server.

The URL needed to start the login process by a user is srr.carleton.ca:8080/attendance. This is the only URL that a user needs to remember.

The MySQL database is replicated on another slave MySQL database server ensuring that all data is backed up immediately. In addition, nightly snapshots of the database are taken. Database administration, such as user, event, and organization creation, modification, and deletion are done on the server using the MySQL workbench.

3.3.2 Interfaces

We have three interfaces to log attendance. The first is our Web interface that users access using the URL mentioned previously, and for this reason Internet access and a browser are required to use our system. This recording interface is provided to a user after they log in to the system, and for this reason the interface is only available to registered instructors for their associated events. Once logged in, a user selects a course, from here on synonymous with an event, they are responsible for and registers students by swiping students' ID cards.

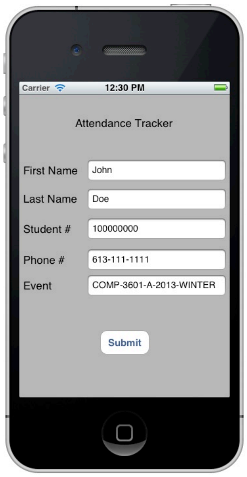
In order to swipe ID cards, users must have a magnetic card reader and access to a USB port on the computer they are accessing our Web interface. Using a Mac or PC, after plugging in the card reader, users should be able to swipe ID cards providing data to the corresponding required field in the Web interface. Our card reader of choice is the MagTek SureSwipe Reader USB HID Keyboard Interface, Model#: 21040145, shown in Figure 1. Single unit price is near $50.00 CDN, with cheaper prices available for multiple unit purchases.

> *"The SureSwipe Reader is available with a USB interface in either HID or a Keyboard Emulation mode and it can be reconfigured in the field. The SureSwipe reader captures 3 tracks of data from all ISO and AAMVA encoded magnetic stripe cards. A green/red LED indicator on the reader provides the operator with status of the reader operations. https://www.magtek.com/shop/sureSwipe.aspx"*

Figure 1: MagTek SureSwipe Reader, Model#: 21040145

Once the server receives the card swipe and event information, it accesses the university database in order to retrieve specific student information needed to complete the register record. If Web logging is permitted for the event, the register is completed and recorded in the database showing it was done with the Web as the device. In this case, altitude, longitude and latitude information is considered perfect giving an accuracy of 1.

The other interfaces we have are through iPhone and Android smartphone applications. These applications require students to enter their first name, last name, phone number and student number into the application, along with the course identification string. Once entered, it is a simple matter of students hitting the submit button at the bottom of the application. In addition to this information, the student's current location information (latitude, longitude, and altitude) on the phone is transmitted to the server. If iPhone or Android logging is permitted for the event, the register is completed and recorded in the database showing it was done with the corresponding smartphone device. The accuracy of the register is also computed and stored in the record by comparing the altitude, longitude, and altitude of the event with the corresponding values sent from the phone.



First Name John
Last Name Doe
Student # 100000000
Phone # 613-111-1111
Event COMP-3601-A-2013-WINTER

Figure 2: iPhone Attendance Tracker

3.3.3 Features

While the current version of the attendance tracking system has many features, we consider the following to be the most important of them:

- Web, iPhone and Android interfaces to log attendance: This features enables events to benefit from the growing existence of smartphones by students and their use to help speed up and personalized student attendance tracking. The Web interface allows for logging when smartphones are not available or desirable. The combination of the three supports attendance tracking in all sizes of courses.
- Registered users: Users of the system are identified as users or administrators. Users are only permitted to view results from events that they are responsible for, while administrators can view everything.
- Reporting: Real-time Excel spreadsheets are downloadable by users and administrators showing attendance results for events.
- Real-time: All logs and reports are done and generated in real-time.
- Inexpensive Hardware: Magnetic card readers are relatively inexpensive and less that $50.00 per unit if purchased in bulk.
- Other than server installation, users don't install any software to support the system.

4 Current System View

Our attendance tracking system is live and running at srr.carleton.ca:8080/attendance. Organizations, events and user can be added by contacting the author directly. Once added, users can log in using the startup URL and providing the information requested, as shown in Figure 3. A user's id is their email address, and their password is provided to them once they request a user id. The password is encrypted in our database so no one can ever view it directly as clear text.

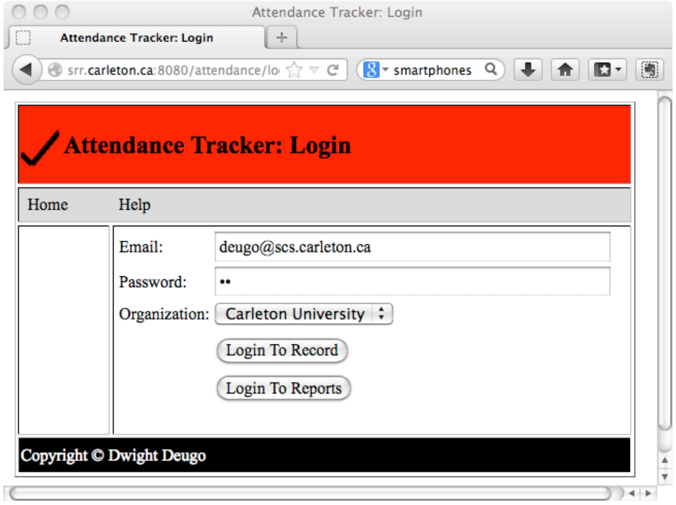
When a user logs in, they request to "Login To Record" or "Login To Reports". When they "Login To Record" and their id and password are correct, they are presented the screen shown in Figure 4. On this screen they can select any course they are associated with and can swipe a student's id card into the Swipe field. If desired, they can enter this information manually. Once they enter the information, by selecting the Record button, the corresponding log is made in the server and the Swipe field is cleared. As shown in Figure 5, the user is given an indication if the log was successful and what information was recorded.

When the user chooses to "Login To Reports" and their id and password are correct, they are presented the screen shown in Figure 6. In this screen they are presented a drop down list of all courses they are associated with. After selecting one of the courses and selecting the Download button, an Excel spreadsheet containing the following columns and associated event data are downloaded for them to save or review:

- First Name: student first name
- Last Name: student last name
- Student Number Phone: student phone number (smartphone only)
- Number : student id
- Event: event/class string
- Organization: event's organization string
- Card Strip : card swipe text (Web only)

- Time: log time
- Latitude: log latitude (smartphone only)
- Longitude: log longitude (smartphone only)
- Altitude: log altitude (smartphone only)
- Accuracy: log location accuracy
- Device: log device

Some column information is Web or smartphone specific.

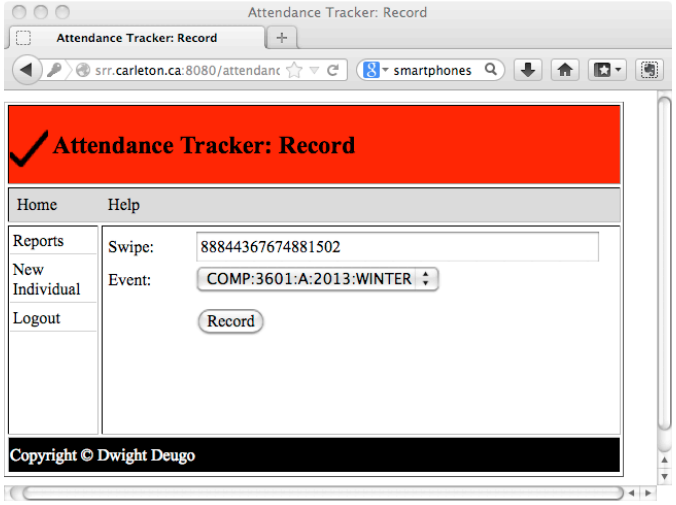

Attendance Tracker: Login
Attendance Tracker: Login
srr.carleton.ca:8080/attendance/lo
smartphones
Attendance Tracker: Login
Home
Help
Login To Record
Login To Reports
Copyright © Dwight Deugo

Email: deugo@scs.carleton.ca

Password: ••

Organization: Carleton University

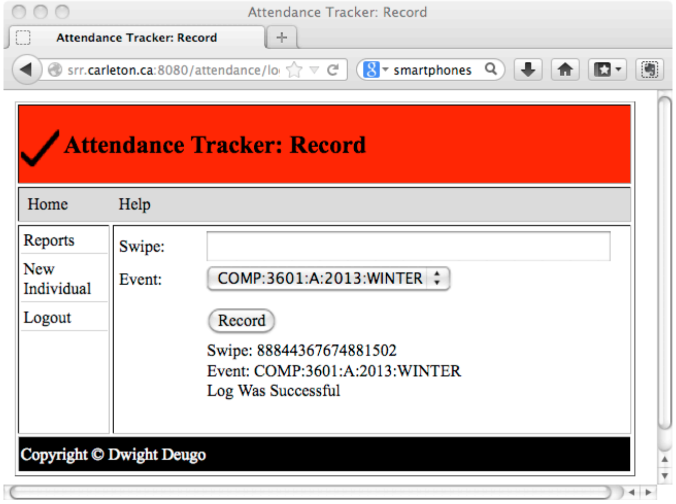
Figure 3: Attendance Tracker Login


Attendance Tracker: Record
Attendance Tracker: Record
srr.carleton.ca:8080/attendanc
smartphones
Attendance Tracker: Record
Home
Help
Reports
New Individual
Logout
Record
Copyright © Dwight Deugo

Swipe: 88844367674881502

Event: COMP:3601:A:2013:WINTER

Figure 4: Attendance Tracker Record


Attendance Tracker: Record
Attendance Tracker: Record
srr.carleton.ca:8080/attendance/lo
smartphones
Attendance Tracker: Record
Home
Help
Reports
New Individual
Logout
Record
Copyright © Dwight Deugo

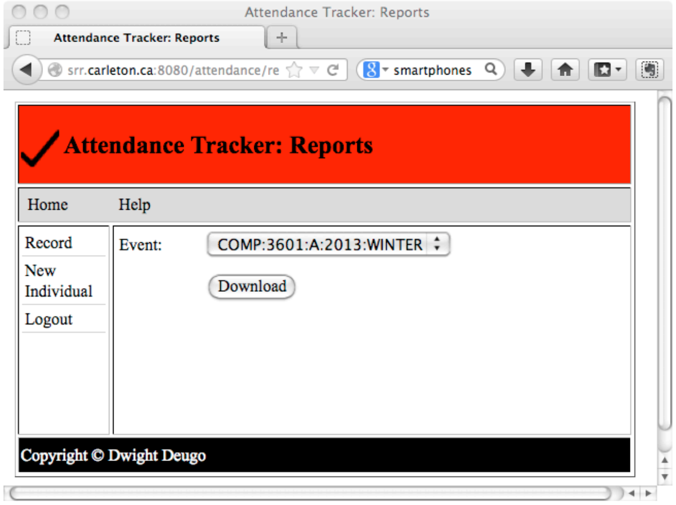
Swipe:

Event: COMP:3601:A:2013:WINTER

Swipe: 88844367674881502
Event: COMP:3601:A:2013:WINTER
Log Was Successful

Figure 5: Successful Log

To make navigation easy, the user can move from recording logs and generating reports at the click of a button on any screen. The user's session is timed and will logout the user after thirty minutes of inactivity.


Attendance Tracker: Reports
Attendance Tracker: Reports
srr.carleton.ca:8080/attendance/re
smartphones
Attendance Tracker: Reports
Home
Help
Record
New Individual
Logout
Download
Copyright © Dwight Deugo

Event: COMP:3601:A:2013:WINTER

Figure 6: Attendance Tracker Reports

5 Conclusion

Our goals for the attendance tracking system were to provide a fast and efficient system that was simple to use in any classroom at Carleton University while not requiring any specialized hardware or installation. To meet these goals we developed a system that had three different interfaces. The first one was a Web interface combined with a USB magnetic strip reader. Just plug in the card reader to any Mac or PC and they will automatically install the required software making the reader simple and easy to use. The Web interface also enables instructors to take attendance for events, combined with the card reader, in a fast and efficient manner. For large classes, the system includes applications for iPhones and Android phones that enables students to use their phones as logging devices. To ensure students are actually in class when logging their attendance, the smartphone's location services are used to provide location information that can be match against the locations of the corresponding events. The software for the server includes Apache HTTP Server, Tomcat, MySQL, Struts2, and Hibernate, all open source and making the cost of software for development $0.00. iPhone development and Android development use Xcode and Eclipse respectively, both freely available to developers. The smartphone applications are available to students at no cost, making the overall cost to them $0.00. Finally, reports are created in Excel format, giving end users flexibility in how they manipulate the data from their classes. Our objectives have been met, in the process of achieving our goals.

The driving force behind the development of our attendance tracking system is student retention and the ability to detect students, especially those in first year, that would benefit from coming to the SSSC and discussing issues they are having with their classes, especially the ones they are not attending regularly. As discussed, this is not only desired by Carleton University's SSSC, but by other university departments that are interested in student experience.

We believe participation, both by students and faculty, depends on how easy the system is to use and on how minimally intrusive it is to the overall class structure. We believe our system achieves these features with its combined interfaces and the simplicity in which both students and faculty access it.

5.1 Future Work

In our system, we are recording what both students and faculty do in order to determine the errors they make using it. By identifying these errors, it provides us with the opportunity to fine-tune the system to make for a better end-user experience. This fine-tuning will be a constant effort on our part in order to improved the overall success of the system.

Currently, server administration, such as the creation of new users, is done through the MySQL Workbench. Standard activities such as this will be moved to a Web interface, so that these reoccurring activities can be done faster and easier.

6 References

[1] Dicle, Mehmet F. and Levendis, John, "The DL-Attendance System: A 29¢ Solution to Electronic Attendance", September 13, 2011, http://ssrn.com/abstract=1926847, Web. 18 Apr. 2013.

[2] "Business professors save class time with automated attendance tracker", Loyola press release - November 8, 2011, http://www.loyno.edu/news/story/2011/11/8/2621, Web. 18 Apr. 2013.

[3] O'Connor, Mary Catherine, "Northern Arizona University to Use Existing RFID Student Cards for Attendance Tracking", May. 24, 2010, http://www.rfidjournal.com/articles/view?7628, Web. 18 Apr. 2013.

[4] Townsend, Allie, "Are Universities Digitally Tracking Students?", Aug. 31, 2010, TIME NewsFeed, http://newsfeed.time.com/2010/08/31/are-universities-digitally-tracking-students/, Web. 18 Apr. 2013.

[5] "Attendance Tracking", http://www.its.txstate.edu/departments/classroom_technology/attendancetracking.html, Web. 18 Apr. 2013.

[6] "Automated Attendance Tracking System", http://www.kc.devry.edu/Auto_AttendanceTrackingSystem.htm, Web. 18 Apr. 2013.

[7] Wiley, Deetra, "New! Attendance Tracking Scanners for UM Classrooms", January 16, 2013, http://technews.blog.olemiss.edu/2013/01/16/new-attendance-tracking-scanners-for-um-classrooms/, Web. 18 Apr. 2013.

[8] "Attendance Tracking Online", http://www.myattendancetracker.com, Web. 18 Apr. 2013.

Breaking the Programming Obstacles using an Automatic Tool.

Luis H. González Guerra and Armandina J. Leal Flores
Computer Science Department, Tecnológico de Monterrey
Monterrey, Nuevo Leon, Mexico

Abstract

There are some programming obstacles that the students have to deal when they are trying to solve a problem using a programming language, like the inexperience of the language, the unknoledge of some algorithms, and some cases the amateurishness of how to test a solution. The technological tools for developing programs and computer systems heavily explore the syntax and semantics of a program but do not provide support to determine its correctness and this is where students may failed by not exploring the correct scenarios and test cases. In this paper, we present the use of automatic evaluation tools as a way to help to remove these obstacles and improve learning experience, coding and testing skills in students.

Keywords: authomatic evaluation, programming skills, problem solving.

1. Introduction

In order to develop strong programming skills, it is critical for students to be constantly challenged by task and problems involving computational solutions [6]. Generally students face a large number of these during the semester [4] so they can build and acquire the necessary skills and techniques needed in program development [1].

To do their jobs, students use development tools for writing code and building computer systems. This set of tools reduce code related issues and strongly explore syntax and semantics of a program, but do not provide support in the analysis of the solution of the problem, this being a major drawback since, most students verify their solution with few cases -usually the most basic to the problem in question- and therefore produce incorrect or incomplete solutions [5].

Some of these exercises are made inside of the classroom where the teacher could guide to the students, and outside of the classroom the student made them without any support or help, and they do not could know how their solution is.

In this paper we present how automatic evaluation software can provide teachers and students a way to improve the programming learning experience, coding and skills, and how a tool like this can complement the problem-solving skills to students, and break programming obstacules inside and outside of the classroom.

2. Background

As part of the programming courses professors suggest using one of the multiple development tools on the market such as Eclipse, Visual Studio, XCode, among others, as they rely heavily on the development of programs from the perspective syntax and semantics in addition to which may be employed for programs in multiple languages.

For several semesters we have seen that most of the solutions of our students are not entirely correct, as they usually poorly tested solutions by only working with the basic scenarios and not exploring the results on corner cases.

Based on this fact, we researched on platforms that support the evaluation of the solutions given by students to their assigned problems. The tools found provided a set of problems that students could solve in several programming languages and validate their solutions by running well designed test cases. Through its use began to show good results with the students' performance, there was a major inconvenience: the existing tools do not allow professors to add new problems to the database so finding problems suitable to the multiple objective of the course became a hard task.

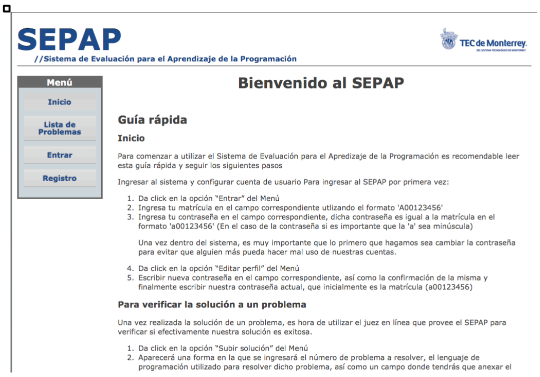
Given this problem and awared that the automatic evaluation supports to solve the above [2, 3], the development of computer system SEPAP (Spanish acronym of "Evaluation System for Learning Programming") started with the aim of helping students in reviewing their solutions. This tool has the functionality to verify the student's solution with multiple test cases including extreme cases where the student fails often related to their lack of experience in some areas.

3. SEPAP project

In January 2011 we undertook the development of SEPAP in collaboration with a group of four Computer Science and Technology students. During this period, we started the requirement engineering process by defining the scope and primary goals of the project. In the first development stage, the tool provide options to add problems and test cases designed by professors. It also include functionalities such as load groups, and above all, allow students to log in and upload solutions. The system was adapted to compile and accept solutions developed in Java or C/C++ and to provide feedback based on the solution submitted in case of acceptance or failure.

3.1 The functionality of the tool

SEPAP works with three kind of user account with different access privileges: student, teacher, and administrator. To enter the system prompts the user registration and password. Figure 1 shows the welcome screen.



Bienvenido al SEPAP

Guía rápida

Inicio

Para comenzar a utilizar el Sistema de Evaluación para el Apredizaje de la Programación es recomendable leer esta guía rápida y seguir los siguientes pasos

Ingresar al sistema y configurar cuenta de usuario Para ingresar al SEPAP por primera vez:

1. Da click en la opción "Entrar" del Menú
2. Ingresa tu matrícula en el campo correspondiente utlizando el formato 'A00123456'
3. Ingresa tu contraseña en el campo correspondiente, dicha contraseña es igual a la matrícula en el formato 'a00123456' (En el caso de la contraseña si es importante que la 'a' sea minúscula)

   Una vez dentro del sistema, es muy importante que lo primero que hagamos sea cambiar la contraseña para evitar que alguien más pueda hacer mal uso de nuestras cuentas.

4. Da click en la opción "Editar perfil" del Menú
5. Escribir nueva contraseña en el campo correspondiente, así como la confirmación de la misma y finalmente escribir nuestra contraseña actual, que inicialmente es la matrícula (a00123456)

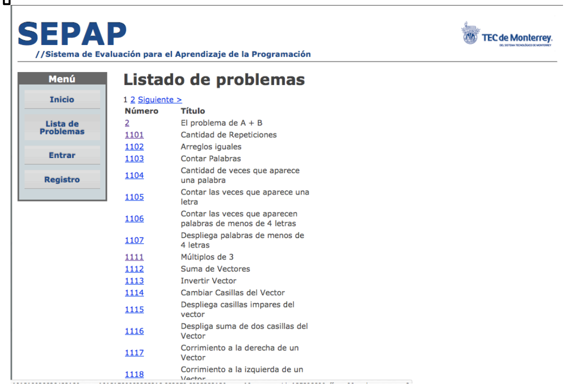
Para verificar la solución a un problema

Una vez realizada la solución de un problema, es hora de utilizar el juez en línea que provee el SEPAP para verificar si efectivamente nuestra solución es exitosa.

1. Da click en la opción "Subir solución" del Menú
2. Aparecerá una forma en la que se ingresará el número de problema a resolver, el lenguaje de programación utilizado para resolver dicho problema, así como un campo donde tendrás que anexar el

Figure 1: SEPAP main screen

SEPAP includes a list of problems that students can solve. The list of problems can be accessed in order to start working on a solution at the "Problem list" button. A screen of this list can be viewed in Figure 2.



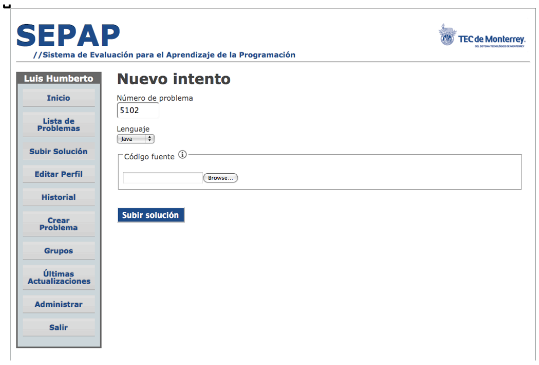
Listado de problemas

1 2 Siguiente >

Número	Título
2	El problema de A + B
1101	Cantidad de Repeticiones
1102	Arreglos iguales
1103	Contar Palabras
1104	Cantidad de veces que aparece una palabra
1105	Contar las veces que aparece una letra
1106	Contar las veces que aparecen palabras de menos de 4 letras
1107	Despliega palabras de menos de 4 letras
1111	Múltiplos de 3
1112	Suma de Vectores
1113	Invertir Vector
1114	Cambiar Casillas del Vector
1115	Despliega casillas impares del vector
1116	Despliga suma de dos casillas del Vector
1117	Corrimiento a la derecha de un Vector
1118	Corrimiento a la izquierda de un Vector

Figure 2: List of Problems

To upload a solution, it is required the number of the problem, the language used for the solution, and an attachment of file with the code. Figure 3 shows the screen to submit a problem.



Nuevo intento

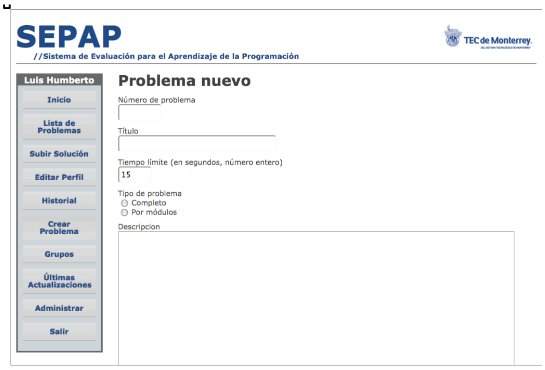
Número de problema: 5102

Lenguaje: Java

Código fuente

Figure 3: Upload a solution

First part to use SEPAP it that the professor design a problem, this problem should have a clear description, an appropriate group of cases to be evaluated, this cases should have a input and output files. The Figure 4 shows the form where a problm could add to the tool.



Problema nuevo

Número de problema

Título

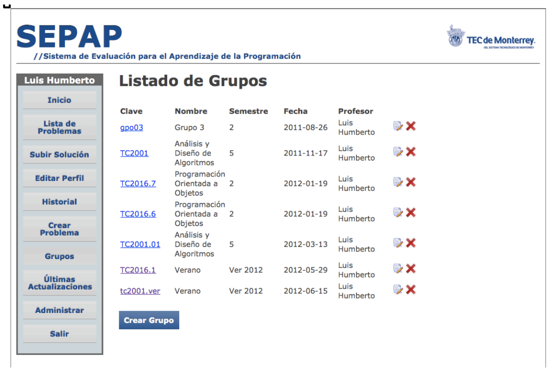
Tiempo límite (en segundos, número entero): 15

Tipo de problema
○ Completo
○ Por módulos

Descripcion

Figure 4. Adding a New Problem

A professor should add groups, add students. The Figure 5 shows how the professor could manage their groups

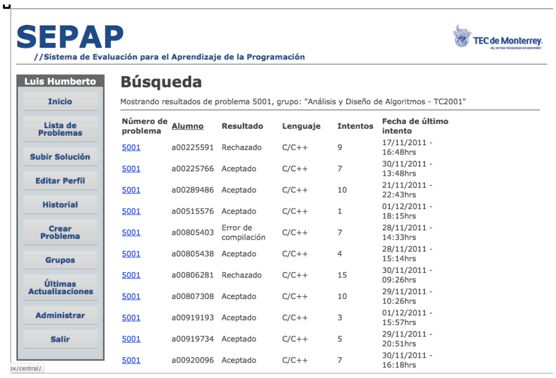


Listado de Grupos

Clave	Nombre	Semestre	Fecha	Profesor
gpo03	Grupo 3	2	2011-08-26	Luis Humberto
TC2001	Análisis y Diseño de Algoritmos	5	2011-11-17	Luis Humberto
TC2016.7	Programación Orientada a Objetos	2	2012-01-19	Luis Humberto
TC2016.6	Programación Orientada a Objetos	2	2012-01-19	Luis Humberto
TC2001.01	Análisis y Diseño de Algoritmos	5	2012-03-13	Luis Humberto
TC2016.1	Verano	Ver 2012	2012-05-29	Luis Humberto
tc2001.ver	Verano	Ver 2012	2012-06-15	Luis Humberto

Figure 5: Group Information Management

Once the student select "submit solution", SEPAP automatically provides feedback which can be ACCEPTED, COMPILATION ERROR, TIME LIMIT EXCEEDED or REJECTED. The professor could see the reports of the preformance of their students. The Figure 6 shows how SEPAP reports the preformance of a group.



Búsqueda

Mostrando resultados de problema 5001, grupo: "Análisis y Diseño de Algoritmos - TC2001"

Número de problema	Alumno	Resultado	Lenguaje	Intentos	Fecha de último intento
5001	a00225591	Rechazado	C/C++	9	17/11/2011 - 16:48hrs
5001	a00225766	Aceptado	C/C++	7	30/11/2011 - 13:48hrs
5001	a00289486	Aceptado	C/C++	10	21/11/2011 - 22:43hrs
5001	a00515576	Aceptado	C/C++	1	01/12/2011 - 18:15hrs
5001	a00805403	Error de compilación	C/C++	7	28/11/2011 - 14:33hrs
5001	a00805438	Aceptado	C/C++	4	28/11/2011 - 15:14hrs
5001	a00806281	Rechazado	C/C++	15	30/11/2011 - 09:26hrs
5001	a00807308	Aceptado	C/C++	10	29/11/2011 - 10:26hrs
5001	a00919193	Aceptado	C/C++	3	01/12/2011 - 15:57hrs
5001	a00919734	Aceptado	C/C++	5	29/11/2011 - 20:51hrs
5001	a00920096	Aceptado	C/C++	7	30/11/2011 - 16:18hrs

Figure 6: Reports

SEPAP version 1.0 was completed in its development stage in May 2011.

3.2 Courses and students involved

Following this, finding the most appropriate course for the semester August-December 2011 was needed to tests our tool. The pilot program started with two groups of the Programming and Data Structures class (TC1005) as its students already have a technical background from the previous required course (TC1002 Programming Fundamentals). By the time the students take this course (sophomore year), they already know the basic programming concepts and are more experienced allowing them to take fully advantage of the evaluation system still under development phase. It was decided that the first programming language accepted by SEPAP would be Java, the same one used in the pilot groups.

In the first test, at the beginning of the session we found out that the students struggled understanding the process to be followed in order to upload and verify solutions using the SEPAP problems. Nevertheless, after overcoming the learning curve, the students began to create and submit their own solutions with no further delays. During the hour and a half session, the students worked on the development of the solution of two problems. At the end of the class, SEPAP had accepted the solution of more than 75% of the students. All this empowered students' confidence, and after allowing late submissions during the rest of the day, a 90% of them got their solutions accepted into the system.

In the second and third test sessions, the students were already familiar with the platform and were encouraged to focus only on the development. In this way, the students were more attentive on working in their test cases before delivering a desktop solutions.

After working on these two groups, three more groups were added to the program so the tool was also used by students enrolled in the Analysis and Design of Algorithm class (TC2001) in which the official development language is either C or C++. The students enrolled on this course are in their junior year and therefor have a stronger technical background and are more experience so they can work with more sophisticated algorithms. While the students taking the Data Structure class faced problems more basic problems usually related to data sorting, the ones taking the design of Algorithms class were working with search techniques such as backtracking (Knapsack problem), DFS and BFS algorithms. In order to increment the challenge, SEPAP allows the professor to add memory and time constrains to the solutions so students are force to work on not only correct solutions but also fast, elegant, feasible and applicable to real scenarios. With this new constrains added, the students had to avoid unnecessary workload in their solutions.

3.3 Data analysis

We gather information focused on two area of opportunity: the way SEPAP helps students on the learning process and about the design and usability of the system.

We designed a survey where all students of the 5 groups answered the following questions:

SEPAP AS A TOOL FOR LEARNING PROGRAMMING

a. How many times have you used the system?
b. Does the use of this tool supports to reinforce the topics seen in class?
c. Do you think you could use SEPAP to increase your ability to develop programs?
d. Is SEPAP easy to use?

DESIGN PLATFORM

a. How would you evaluate the interface?
b. How would you evaluate the steps to find a problem?
c. How would you evaluate the steps to upload a solution?
d. How would you evaluate the response time to a solution?
e. How would you evaluate the history of your solutions?
f. How do you rate SEPAP overall?

4. Results for SEPAP

The feedback provided by the students enriched the next versions and growth of SEPAP.

The data was obtained from 5 different test from 3 different courses. The problems proposed go from a basic to an advance level involving solutions developed in Java, C and C++.

4.1 Results for learning programming

In the Figure 7 we can find the answer of the student to the question Does the use of this tool supports to reinforce the topics seen in class? And the analysis shows that more than 65% think is very helpful the use of a authomatic tool additional to the sections in classes.

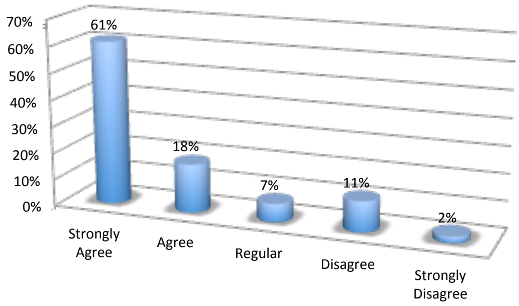


Figure 7: Does the use of this tool supports to reinforce the topics seen in class?

In the Figure 8 we can see that the tool is acepted perfectly of the students because to the question of Do you think you could use SEPAP to increase your ability to develop programs? The are 100% agree with this tool to increase their abilities to develop programms.

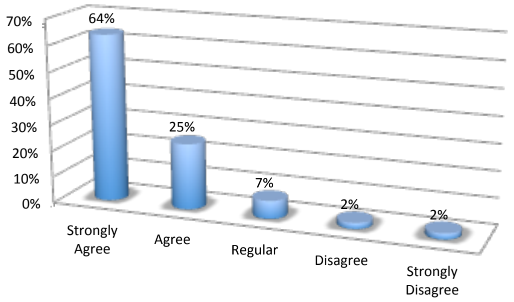


Figure 8: Do you think you could use SEPAP to increase your ability to develop programs?

In the Figure 9 we can discover that the tool is not so easy to use at the beginning becouse, the results to the question Is the SEPAP easy to use? Shows that the student have dificulties to understant the use of a tool of this kind, because the results are so similar between agree and disagree.

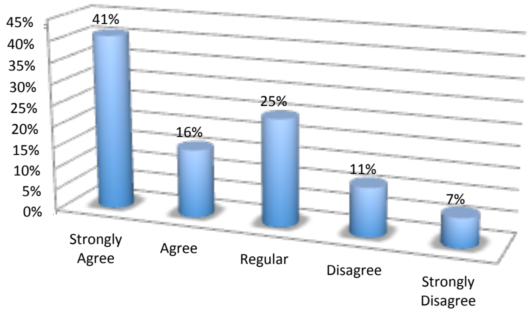


Figure 9: Is SEPAP easy to use?

We can discover that SEPAP is a very good tool to use in programming clases but we have to guide a litter more at the first use of this may be with a problem during the class.

4.2 Results for design plataform

We evalute the design plataform with five questions that tried to know the appreciation of the students about how they feel the use of the plataform in the interface use.

In the figure 10 we can see that the 100% of the student is agree or almost agree about the fisical interface, because the anwered in this way to the question How do you evaluate the interface?

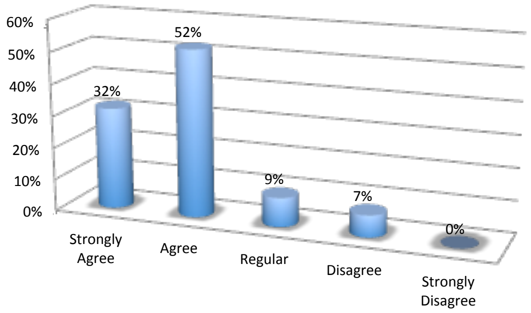


Figure 10: How do you evaluate the interface?

We can see that they take some minutes to find a problem, because the answered not so good to the question How would you evaluate the steps to find a problem? We can see the results of this question in the Figure 11.

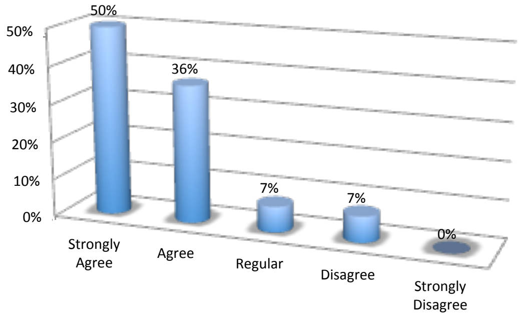


Figure 11: How would you evaluate the steps to find a problem?

Almost 90% thinks that is easy the way to upload their solution to a problem. In the Figures 12 we can see the results of this question.

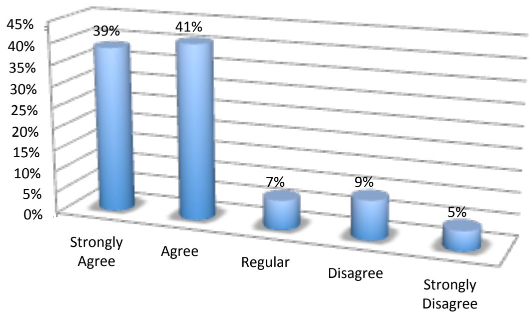


Figure 12: How would you evaluate the steps to upload a solution?

One of the key aspect when you are using a authomatic tool for solve problems, is the timing that te tool takes to answer a sumbition, in this case all the students were very agree that SEPAP gives a fast answer of their sumbitions.

Fortunately the students thinks that SEPAP provides excellent answer time, and this could help to them to apreciate faster how their solution was as the response regarding expected.

We can see in the Figure 13 the positve results of the question How would you evaluate the response time to a solution?

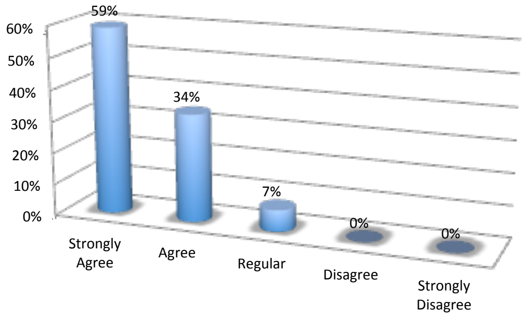


Figure 13: How would you evaluate the response time to a solution?

Another important aspect of this kind of tool, is how it save our submitions, and almost the 80% of the students think that SEPAP handle very good the historial of their submitions. In the Figure 14 we can see the results of this question.

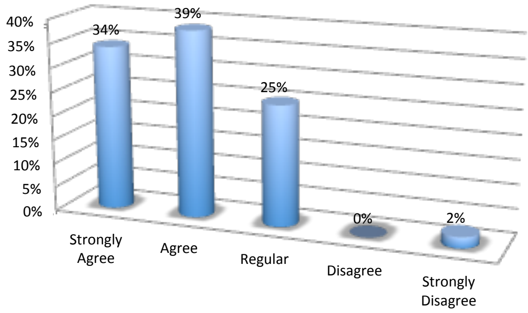


Figure 14: How would you evaluate the history of your solutions?

4.3 Summary of Results.

The results obtained from the end of course survey provided an excellent response from students to SEPAP:

- 82% of students believe that the tool supports the generation of strong skills to develop programs.
- 81% of students believe that the system is very user friendly and easy to use.
- 95% of students believe that the response time is excellent

For the first test, it took 4.7 submissions for a solution in order to get accepted to the students. In contrast to the third test where it only took an average of 1.4 submissions for their solution to get accepted.

These results reflect how students are gaining more awareness to send complete solutions for their activities, and we believe that SEPAP supports students in their learning of programming by providing a medium through which they can effectively develop their programming skills.

In the Figure 15 we can see the answer of the studen to the question How do you rate SEPAP overall? Where more than 90% think that in general SEPAP is an excelent plataform to support the topics seen in programming classes.

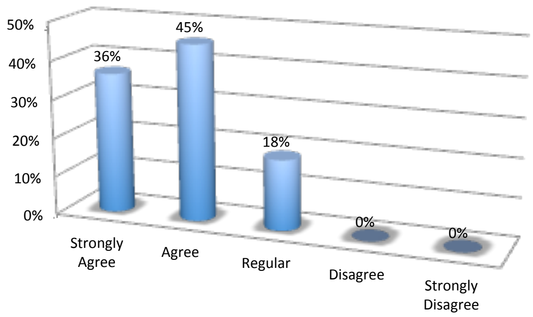


Figure 15: How do you rate SEPAP overall?

5. Conclusions and future work

The surveys reflected similar results regardless the fact that student students were enrolled in different courses with different programming languages and were solving problems of different nature and degree of complexity. They all agree that this type of tool can reinforce the course topics.

The results lead to the conclusion that a package of this nature will allow students to:

- Improve their programming skills as they can validate that programs developed are correct.
- Try different solutions to improve efficiency.
- Understand the importance of checking their solutions with all test cases.
- Receive automatic feedback on the faults their program presents.
- Increase their confidence in their programming skills so they find out their potential to develop good and correct solutions.

On the other hand, SEPAP allow and help professors to:

- Have a set of problems covering the study material in their multiple classes.
- Develop problems related to specific topics and increase (or diminish) the complexity by creating the correct test cases or setting up certain constrains.
- Obtain automatically a review of the solutions of the students.

The SEPAP is being widely spread within the institution and adopted by more professors willing to test the increased programming skills related to the use of automated evaluation tools. In addition, this provides more feedback and expands the range of reports to support the tool and the resolution of existing problems.

6. Acknowledgments

We thank Rafael García, Humberto Garza, Alan Gandarilla, and Roberto Plancarte for their valuable work on the development of the project.

We also thank Yolanda Martínez for being part of the professor who helps testing the tool in her groups and Juan A. Nolazco for his unconditional support for this project.

7. References

[1] Vasconcelos-Santillán, Jorge, "Development of a web-based surveying instrument to identify problem-solving abilities related to…", *ProQuest Dissertations & Theses* (PQDT).

[2] Reguera, Jorge López, Hernández Rivas, Cecilia, Farran Leiva,Yussef. "Una plataforma de evaluación automática con una metodología efectiva para la enseñanza/aprendizaje en programación de computadores". Universidad de Concepción, Ingeniare, 2011, Vol. 19 No. 2, pp. 265-277, Concepción, Chile.

[3] Petri, IHANTOLA "Creating and Visualizing Test Data from Programming Exercises", *Informatics in Education,* Institute of Mathematics and Informatics, Vilniu., 2007, Vol. 6, No. 1, pp. 81–102.

[4] Riku Saikkonen, Lauri malmi, Ari Korkhonen "Fully Automatic assessment of Programming Exercises" *ACM SIGSE*, September 2001, Vol. 33, No. 3, pp. 133-136.

[5] David Jackson "A semi-automated Approach to Online Assessment" *ACM SIGCSE*, September 2000, Vol. 32, No. 3,1, pp. 64-167.

[6] Misook Heo "A Learning and Assessment Tool for Web-based Distributed Education, CITC4 2004, pp 151-54

International Computer Science Capstone Project Exchange Program: A Case Study Report

Dean Knudson
Computer Science Department, North Dakota State University
Fargo, North Dakota, United States

Abstract - *Real-world capstone projects that are done for regional industry are very important for computer science students. However, once the students leave for jobs in industry they often find themselves working on internationally distributed teams. Thus, it would be of great benefit for students to have the opportunity to work remotely on a project that is being done for a company in another country and mentored by employees of that company. For the past two years North Dakota State University (NDSU) has included capstone projects from Germany and Sweden in its computer science program. A new unique project exchange pooling concept has been used to set up the projects, which has worked very well. This paper will describe the exchange pool methodology and discuss in some detail the projects that were completed. Lessons learned as well as potential improvements and long term goals will also be described.*

Keywords: capstone, international, case study, industry partnership

1 Introduction

Over the past several years it has become clear that including a capstone team experience in computer science degree programs has great value for students. This typically means implementing a significant software development project. Having this project be a *real-world* experience done for an industrial sponsor adds a level of reality and practicality to the learning. However, it is increasingly common for teams in industry to work with teams in other countries. Thus, although these capstone programs are very successful, they could be improved by including an international experience which would make them an even stronger learning experience. Working across international boundaries provides many learning opportunities (e.g. understanding the global context of situations and cultural differences in the workplace, dealing with language barriers, time zone issues, and different technology preferences).

In 2012, NDSU began working with universities in Germany and Sweden to coordinate capstone exchange projects. Teams of students in each of these countries worked for companies in the US while in the US, student teams worked for companies in Germany and Sweden. This was facilitated by establishing an exchange pool model where faculty in each country found companies which would be willing to sponsor capstone projects to be done by international teams in a different country (see Figure 1). The student teams worked remotely from their local universities. The results were very successful and were reported at the Capstone Design Conference in Champaign-Urbana, IL last summer. [1]

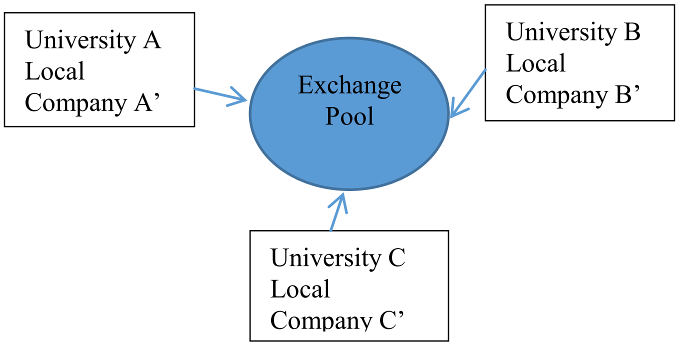


Figure 1 – Exchange Pool

2 Initial Exchange Program

Each participating university had already been carrying out industry-based capstone projects in their curriculum for some time. In those instances each university had its specific characteristics which varied by institution. The capstone course is highly appreciated by students since it provides an opportunity to gain some experience from all phases of software development. The capstone experience also covers methods, documentation and inter-group communication. In the exchange pool model it is important that the university be able to follow its own process when working for companies in host countries.

2.1 Hannover Program (Germany)

The CS department at the University of Applied Sciences in Hannover offers a one-year project in the final year of their bachelor's program. The theory of software engineering is taught in classes outside the capstone class. Several projects are offered each year by different faculty members (most in cooperation with industry) and students are assigned to the projects based on a bidding and auction process. External partners provide general project ideas and requirements while faculty members provide technical and organizational leadership. Student teams consist of 6-10 students and the students are expected to spend one day a week on the project, which counts for 8 ECTS credits. Typical characteristics of a capstone course in this program are shown in Fig. 2.

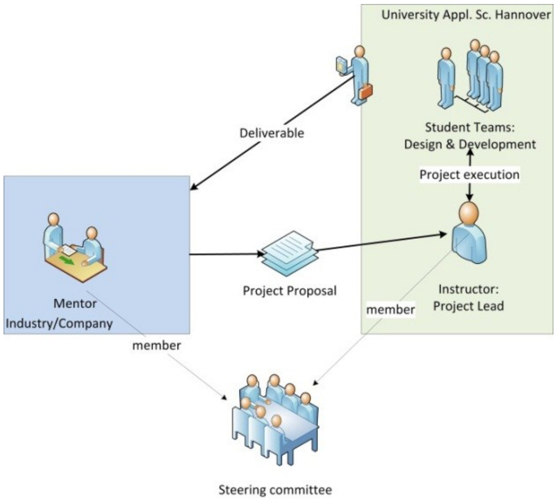


Figure 2: Cooperation in capstone at Hannover

2.2 NDSU Program (United States)

The NDSU B.S. program requires undergraduates, working in teams of 3-4 students, to complete a one-semester, three US credit capstone course. The course is centered on team-based software development projects of immediate practical value for local or regional businesses. Typically, several international students are distributed throughout the teams so students gain experience working with people from other countries. Students also gain experience with remote development by working projects for companies in other cities. Company sponsors/mentors prepare project proposals which are reviewed by the capstone instructor. Projects are presented to the students who then bid on which project they would prefer. Teams are assigned by the instructor. The company mentors interact with their teams on a weekly basis providing technical and business guidance to the student team. Typical characteristics of a capstone course in this program are shown in Fig. 3.

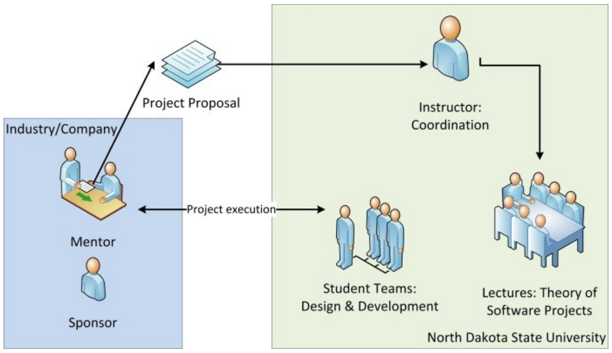

Project Proposal
Industry/Company
Instructor: Coordination
Project execution
Mentor
Sponsor
Student Teams: Design & Development
Lectures: Theory of Software Projects
North Dakota State University

Figure 3: Cooperation in capstone at NDSU

2.3 Linköping Program (Sweden)

At Linköping University in the third year of the curriculum, there is always a software development project performed in groups of 6-8 students with an external customer. The course runs over the spring semester and is worth 8 ECTS credits. Typically some 30 proposals are available for 10 groups to select from. In the proposal the customer normally presents the general purpose of the product and 3-5 important, high-level requirements. Customer contacts are supposed to work about 25 hours, giving interviews in requirements analysis, evaluate prototypes, and provide contacts to end-users, if applicable. Each group has a tutor, with whom they meet approximately once-a-week. The tutor supports the group with answering questions about tools and processes. Issues with the customer are handled by the examiner. Typical characteristics of a capstone course in this program are shown in Fig. 4.

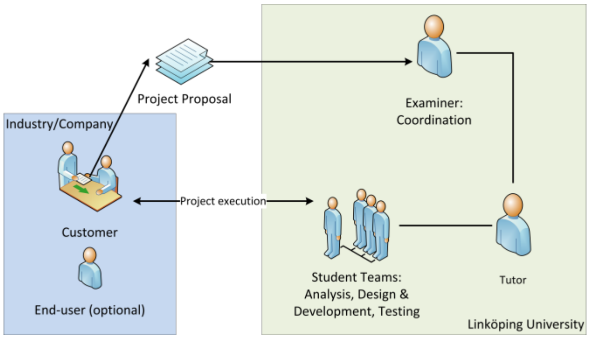


Figure 4: Cooperation in project at Linköping

3 Capstone International Exchange Projects at NDSU

In this section four projects will be described. These are the four international projects done by NDSU students during the 2012 and 2013 spring semesters. In each year one project was done in Germany and one in Sweden.

.

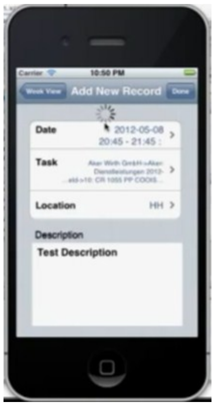
3.1 Spring 2012 - Germany

Company G1 - iPhone App

Company G1 is an SAP consulting and development company. The internal time recording system used by the company was outdated and had an antiquated look and feel. The student team was tasked with building an iPhone application to sync data with a company's SAP modules using SOAP-based web services provided by the company's SAP system.

Features:

- Uses VPN services from Company G1
- Calendar and list views
- Allows addition, modification, and deletion of records
- Can remember user name and password for a user
- Client-side handling of SAP restrictions



Date 2012-05-08 20:45 - 21:45

Task Aker Wirth GmbH->Aker Dienstleistungen 2012- ...ald->10: CR 1055 PP COOKS...

Location HH

Description

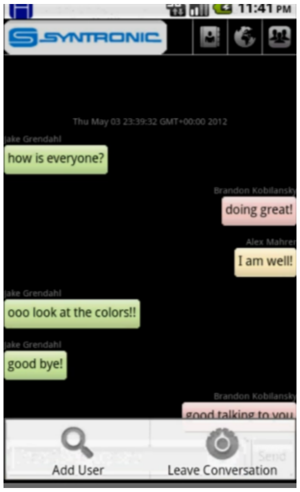
Test Description

3.2 Spring 2012 - Sweden

Company S1 is a design house that works on both software and hardware development, and the testing and maintenance of complex technical systems. The company strives to develop complete solutions, starting with ideas and ending with a complete system. As a result the company has many different projects and groups at any one time. The purpose of this project is to ease communication between members of a group, by indicating where members are located and providing multiple communication methods.

Features:

- Text-based chat uses Java sockets to avoid texting fees
- Java servlets provide functionality to add and modify conversations
- Uses session management protocol for multimedia
- Produced a SIP planning document for future groups to continue work



Thu May 03 23:39:32 GMT+00:00 2012

Jake Grendahl: how is everyone?

Brandon Kobilansky: doing great!

Alex Mahrer: I am well!

Jake Grendahl: ooo look at the colors!!

Jake Grendahl: good bye!

Brandon Kobilansky: good talking to you

3.3 Spring 2013 - Germany

Company G1 (repeat company) – AutoStore to Jive

Company G1 is a heavy user of Jive and wanted the ability to use NSi AutoStore to transfer documents to Jive for collaboration.

- Built components to save data to Jive.
- Components captured information from the Jive Instance (as an example, where documents can be saved).
- Made use of the RESTful WebService to save the document to Jive.

NSi
Powered By AUTOSTORE
j
jive

3.4 Spring 2013 - Sweden

Company S2 - Ping utility with added flexibility

Company S2, a large telecommunications company, wanted a ping utility that offered more flexibility than existing ones in order reduce the amount of time spent making scripts.

Features

- Released to open source
- Platform independent
- No installation required
- Windows command line interface (flags as follows)
  - Sample command line flags/parameters
  - Sets the time between sending multiple requests in milliseconds.
  - Sets the time between when a reply is received and when the next ping is sent out
  - Sets the size of the entire ICMP packet
  - Sets the size of the payload of the ICMP packet
  - Randomizes the size of the payload between min and max, uses discrete uniform distribution
  - Randomizes the size of the payload using an average and standard deviation distribution
  - Randomizes the time between sending multiple requests between min and max, uses discrete uniform distribution
  - Randomizes the time between sending multiple requests using an average and standard deviation distribution
  - Sets sequentially increasing size of pings
  - Sets the number of pings to be excluded from statistics
  - Sets the number of pings to be sent out

4 Lessons Learned

After two years of running international capstone projects using the exchange/pooling process there are a number of lessons that have been learned.

- Using an exchange pool is very easy to manage if the associated professors are excited about the program
- International capstone projects have been very popular among students, faculty, industry and university administration
- You must be flexible when dealing with alternative ways of running a project (e.g. the company in the US had to be willing to have no team assigned to their project if the students did not pick their project – this happened the second year working with Sweden)
- Free/cheap video conferencing tools like Skype or Google+ work well, improve understanding and simplify discussions (although they are not as good as face-to-face meetings, which students miss)
- Need to account for lack of equipment (e.g. Mac laptop)
- Only small timeslots to hold video conferences, may want Wikis or forums
- Daylight savings times and holidays vary
- Language has not been a major issue – some smaller issues with NDA documents
- Companies should spend extra time defining the projects up front
- Shared documents, shared desktops allow for better understanding (diagrams are useful since they are less language dependent)
- Although documentation may all be in English, it is not uncommon to see variable names in the local language
- Having limited times when teams can communicate often leads to better thought-out questions

5 Potential Improvements and Long Term Goals

There are several areas where improvements could be made. Students really love the experience and seem to get a lot out of working internationally. However, there are very few empirical studies that can document what the students really learn. Many of these studies are currently being started and results should be coming out soon. [2][3]

In the long term, this model could be expanded to other universities and departments. This could possibly involve the establishment of a pool of industry/university partners that are interested in international capstone exchanges. Industry/university partners could then be paired with other industry/university partners that meet some level of compatibility (one or two semesters, CS/EE/ME/etc., free/fee, etc.). This would allow interested universities a fairly easy mechanism for setting up international capstone exchanges. It is to be determined who would administer such a pooling/matching system.

6 References

[1] D. Knudson, C. Kleiner, K. Sandahl. "A Preliminary Report on Establishing an Industry Based International Capstone Exchange Program, Capstone Design Conference 2012, Champaign-Urbana, IL (July 2012).

[2] J. Aidoo, S. Sexton, J. Hanson, R. Houghtalen, M. Lovell. "International Design Project Experiences: Assessing the Longterm Impact on Students", Capstone Design Conference Champaign-Urbana, IL (July 2012).

[3] S. Thoe, J. Summers. "Survey Comparison of National and International Capstone at Clemson University", Capstone Design Conference Champaign-Urbana, IL (July 2012).

A JFLAP Extension for Checking Context-Free Grammars

Priyanka Bansal[1] **and Viraj Kumar**[2]
[1]Computer Science and Engineering, NIIT University, Neemrana, India
[2]Department of Information Science and Engineering, P.E.S. Institute of Technology, Bangalore, India

Abstract - *JFLAP is a freely available and popular software package used in formal languages and automata courses. We develop a JFLAP extension for checking homework assignments involving the design of context-free grammars (CFGs). Determining the language equivalence of arbitrary CFGs is undecidable. As a result, instructors as well as students typically resort to manual checking that is slow and error-prone. For instructors, we automate this process by verifying that student CFGs match the instructor's CFG for all strings of length up to L (an instructor-supplied parameter). Our tool is easy to use and is practical for large batches of students. It can also be used by students (e.g., to test whether two approaches to a particular problem are equivalent up to length L), giving valuable feedback quickly and reliably.*

Keywords: Computer-aided assessment, Formal Languages, Context-Free Grammars, JFLAP.

1 Introduction

Homework assignments in formal languages and automata (FLA) courses are often paper-and-pen based. They typically include problems of constructing automata or grammars for a given formal language. Testing (for students) and grading (for instructors) is usually a laborious and error-prone process. The advent and adoption of tools such as JFLAP [1] and L-FLAT [2] is, however, changing this landscape. Students can use such software to code solutions, and easily design and run complex unit-tests. Failing such tests gives students valuable feedback and enables them to correct their solutions, whereas passing such tests gives confidence in the correctness of their approach. JFLAP is extremely popular world-wide (see [3] for usage statistics) because it covers nearly every topic in the FLA course. Furthermore, the community actively encourages the development of grading support for instructors [4]. The present paper is similar to [5] in its contribution: whereas [5] is a tool for checking equivalence of finite automata, our tool focuses on context-free grammars (CFGs).

It is well known that CFG language equivalence is an undecidable problem [6]. Most instructors, therefore, apply a heuristic similar to this: If there is a short counter-example x, then mark *INCORRECT*, else if the student CFG "looks OK" then mark *CORRECT*, else mark as per the grading policy.

A "counter-example" is any string x that is generated by one of the CFGs, but not the other. One can easily automate the search for such a string up to a finite, instructor-specified length L [1]. On the other hand, it is not clear what it means for a CFG to "look OK". Indeed, this subjective judgment is where errors and inconsistencies in grading are most likely to creep in. We can eliminate this imprecise test to obtain the following modified grading heuristic: If there is a counter-example of length $\leq L$ then mark *INCORRECT*, else mark *CORRECT*.

A student solution that simply generates the requisite strings up to length L will, of course, be deemed "correct" under our heuristic. However, students are unlikely to attempt such tailored solutions for at least two reasons: (1) the instructor need not reveal L, and (2) if L is large enough, the tailored CFG is likely to be conspicuously larger (in terms of variables and rules) than an honest attempt.

Computer-aided assessment tools such as ours offer a reliable and effective way of grading computer science assignments, with a history of success spanning over five decades [7] (see [8] for a detailed overview). In addition to speed and accuracy, automation has two further benefits, both of which are particularly relevant in FLA courses: (1) By streamlining the grading process, our tool encourages instructors to assign more problems than time-constraints would allow under manual grading, and (2) our tool is also available to learners, providing automated feedback which is not prone to the errors and delays of manual feedback. This encourages learners to experiment with a variety of approaches to problem-solving, which is a prized skill in this domain.

In order to leverage these benefits, we have tried to make our tool easy to install and immediately usable. (A much more sophisticated and faster prototype application is available [9], but this is not designed as a computer-aided assessment tool and is therefore harder to use.) Also, to keep our tool as general as possible, we do not assume that the CFGs in question have efficient parsers. Our tool does, at present, have two primary limitations: (1) memory and running-time constraints limit L to about 15, and (2) there

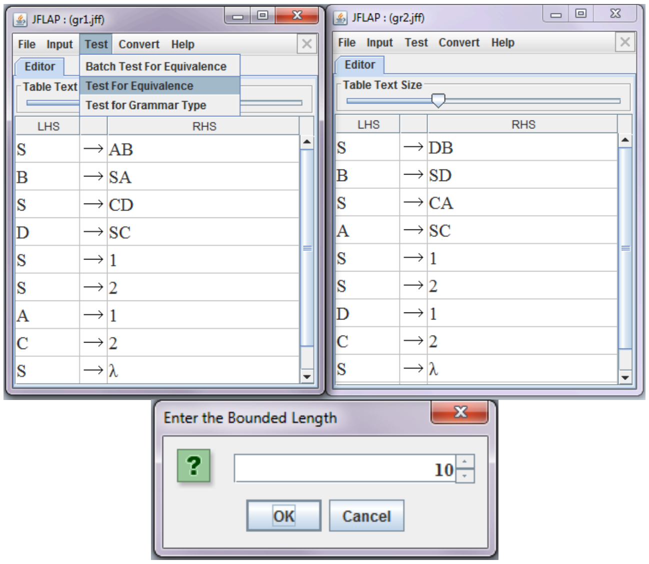
[1] Since there are exponentially many such strings, some instructors test only a small random sample of such strings.

cannot be more than 64 variables in our internal Chomsky Normal Form (CNF) representation of the given CFGs [2]. Both these limitations are the consequence of our rather naïve initial implementation, and are being addressed.

The remainder of the paper is structured as follows. In Section 2 we explain how our tool integrates with JFLAP, and demonstrate features that are helpful to both students and instructors. Section 3 details experiments to determine the practicality of our tool. Finally, in Section 4 we conclude with our future research directions. The latest version of our tool is available at www.niituniversity.in/projects/CFGeq.zip, with a README file describing the installation steps.

2 Integration with JFLAP and usage

Our tool integrates with JFLAP via an intuitive access-point under the **Test** tab for Grammars. We first demonstrate the simplest usage of our tool, **Test for Equivalence** (Fig. 1). The user selects two CFGs and a length bound L, and the tool compares the CFGs on all strings of length ≤ L.



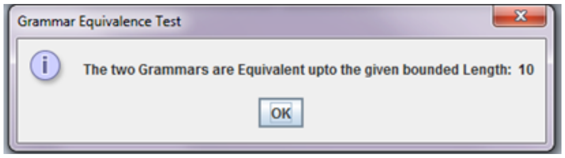
LHS	RHS
S	→ AB
B	→ SA
S	→ CD
D	→ SC
S	→ 1
S	→ 2
A	→ 1
C	→ 2
S	→ λ

LHS	RHS
S	→ DB
B	→ SD
S	→ CA
A	→ SC
S	→ 1
S	→ 2
D	→ 1
C	→ 2
S	→ λ

10

Figure 1. Testing two CFGs for equivalence

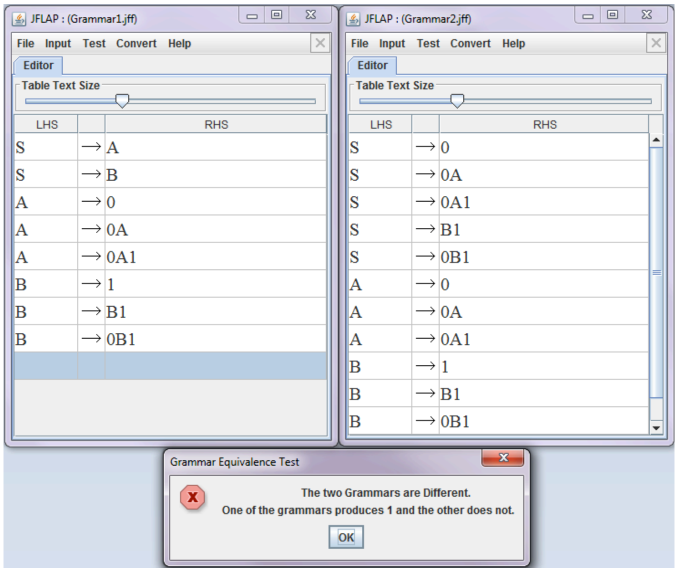
The CFGs in Fig. 1 are obviously equivalent (the only difference being that variable A in the first CFG has been renamed D in the second CFG), and the tool reports the expected result (Fig. 2).



The two Grammars are Equivalent upto the given bounded Length: 10

Figure 2. CFGs equivalent up to tested length $L = 10$

This simple functionality can nevertheless be extremely valuable for learners. Consider the somewhat more complex example in Fig. 3 where a student is attempting to remove so-called "unit rules" ($S \rightarrow A$ and $S \rightarrow B$) from the left-hand CFG to obtain an equivalent CFG without such rules. In following a well-known algorithm for this, our student has made a small mistake in the CFG on the right.



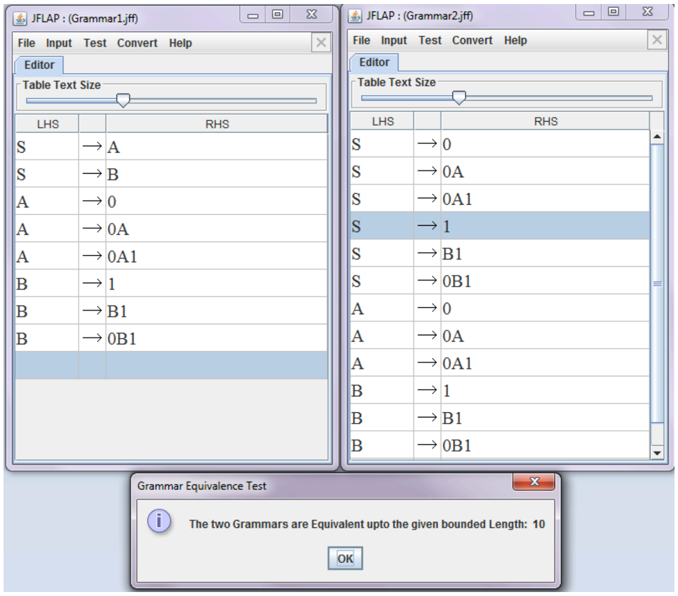
LHS	RHS
S	→ A
S	→ B
A	→ 0
A	→ 0A
A	→ 0A1
B	→ 1
B	→ B1
B	→ 0B1

LHS	RHS
S	→ 0
S	→ 0A
S	→ 0A1
S	→ B1
S	→ 0B1
A	→ 0
A	→ 0A
A	→ 0A1
B	→ 1
B	→ B1
B	→ 0B1

The two Grammars are Different.
One of the grammars produces 1 and the other does not.

Figure 3. Two inequivalent CFGs

Our tool reports that the CFGs are inequivalent, and also displays a *shortest* counter-example establishing this fact [3]. Our student may now guess that one extra rule ($S \rightarrow 1$) fixes the problem, and she can quickly test this hypothesis (Fig. 4).



LHS	RHS
S	→ A
S	→ B
A	→ 0
A	→ 0A
A	→ 0A1
B	→ 1
B	→ B1
B	→ 0B1

LHS	RHS
S	→ 0
S	→ 0A
S	→ 0A1
S	→ 1
S	→ B1
S	→ 0B1
A	→ 0
A	→ 0A
A	→ 0A1
B	→ 1
B	→ B1
B	→ 0B1

The two Grammars are Equivalent upto the given bounded Length: 10

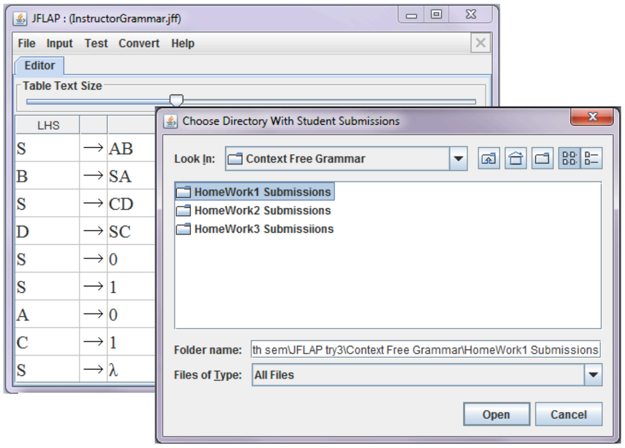
Figure 4. Correcting the error by adding one more rule

[2] JFLAP restricts CFG variable names to the 26 uppercase letters of the alphabet, so ours is not a severe limitation. Also, although JFLAP provides an automatic conversion of CFGs into CNF, it uses an inefficient process. Our internal CNF representation uses the more efficient process detailed in [10].

[3] As per Occam's principle, a shortest counter-example is likely to be most informative.

We believe our hypothetical student can greatly benefit from this kind of clear and rapid feedback. Having found the fix, she can easily discover that her earlier mistake was (in this case) caused by improper book-keeping while executing the algorithm, and not by a fundamental misunderstanding. In the latter case, she would probably need to devote significant time to properly understand the process, whereas here she knows that she merely needs to be watchful.

Instructors are likely to find the second feature of our tool more helpful: **Batch Test for Equivalence** (Fig. 5).



JFLAP : (InstructorGrammar.jff)

LHS		
S	→	AB
B	→	SA
S	→	CD
D	→	SC
S	→	0
S	→	1
A	→	0
C	→	1
S	→	λ

Choose Directory With Student Submissions

Look In: Context Free Grammar

HomeWork1 Submissions
HomeWork2 Submissions
HomeWork3 Submissiions

Folder name: th sem\JFLAP try3\Context Free Grammar\HomeWork1 Submissions

Files of Type: All Files

Figure 5. Batch testing: instructor CFG and student submissions folder

To perform a batch test, the instructor first opens the sample solution and clicks **Batch Test for Equivalence**. The tool then asks the instructor to choose a folder containing student submissions and the bounded length L. The tool checks each student's submission against the instructor's sample solution on strings of length ≤ L, and reports the results sequentially using dialog boxes similar to Fig. 2 and Fig. 3.

3 Experiments

In this section, we report running times for our tool on "difficult" inputs. We focus only on the **Test for Equivalence**, since batch testing merely performs this equivalence test on multiple inputs.

If both CFGs are defined over an alphabet with m letters, there are $\Theta(m^L)$ strings of length at most L. The most naïve approach would be to examine each such string x and determine whether both CFGs can derive x, or neither can, using the classical CYK algorithm [11][12][13]. This would yield a worst-case running time of $O(rL^3m^L)$, where r is the number of rules in the CNF grammar. The current version of our tool uses a modified CYK algorithm that remembers the derivations of short strings (in an m-ary tree data structure), and can reference this information while parsing longer strings. A careful implementation yields a worst-case running time of $O(rLm^L)$, which is about 50–100 times faster than the naïve approach on typical inputs.

For our experiments, we generate random CFGs (in CNF) with m letters and v variables, and set $r = mv$ (this choice tends to generate "hard" test cases). For each such CFG, we generate a new CFG by randomly permuting the names of the variables (except the start-variable). The new CFG is clearly equivalent to the original, and forces our code to run for the longest possible time. For such adversarial inputs, we report the run-times (in milliseconds) obtained by running our code on a low-end Windows XP version 5.1 machine, with a Pentium® Dual-Core 2.6 GHz CPU and 2 GB RAM. The data in Tables 1 to 4 below is averaged over four runs.

		L						
		8	**10**	**12**	**14**	**16**	**18**	**20**
v	**3**	0	0	16	78	344	1719	7766
	5	0	4	24	98	450	2164	9270
	10	4	8	28	141	471	3231	14848
	15	4	8	39	180	887	5048	18062
	20	4	12	55	235	1142	5891	23676

Table 1. Run-time in milliseconds ($m = 2$, $r = mv$)

		L						
		7	**8**	**9**	**10**	**11**	**12**	**13**
v	**3**	4	16	43	145	405	1636	5723
	5	8	16	59	200	735	2230	8012
	10	12	28	117	371	1274	4184	13223
	15	16	43	149	515	1684	6524	18265
	20	21	75	227	829	2180	7656	33375

Table 2. Run-time in milliseconds ($m = 3$, $r = mv$)

		L						
		3	**4**	**5**	**6**	**7**	**8**	**9**
v	**3**	0	0	8	24	149	813	4750
	5	0	0	8	39	217	1364	7016
	10	4	4	8	82	383	2305	15225

	15	4	8	20	106	571	4043	20254
	20	4	8	28	129	926	5289	29935

Table 3. Run-time in milliseconds (m = 5, $r = mv$)

		L					
v		**1**	**2**	**3**	**4**	**5**	**6**
	3	0	0	0	16	180	2325
	5	0	0	4	24	292	3958
	10	0	4	8	51	723	8590
	15	0	4	9	67	1082	13317
	20	0	4	16	110	1621	19391

Table 4. Run-time in milliseconds (m = 10, $r = mv$)

4 Conclusions

The present research was conducted as part of a semester-long project by the first author in her junior year, and there are several opportunities for follow-up work. To begin with, our experiments yield tolerable run-times only when L is quite small ($L \leq 18$ for $m = 2$, $L \leq 12$ for $m = 3$, $L \leq 8$ for $m = 5$, and $L \leq 5$ for $m = 10$). This may be suitable for the kinds CFGs found in typical FLA courses, but we propose to implement the incremental SAT solver approach in [9] for the next version of our tool, which is vastly quicker. We noted in the introduction that our CFGs are internally converted into CNF using a more efficient process than the one used by JFLAP. However, as pointed out in [10], Binary Normal Form (2NF) is an even more compact normal form (the normalized CFG is at most a constant factor larger than the original CFG). We therefore propose to augment JFLAP with a 2NF normalizer for CFGs, and use this in our implementation. Lastly, we can enhance the functionality of **Batch Test for Equivalence** by recording outputs in a log-file, instead of displaying the result of each test sequentially.

5 Acknowledgements

This research work was funded by NIIT Ltd., and was conducted wholly at NIIT University, Neemrana.

6 References

[1] S. H. Rodger, B. Bressler, T. Finley, and S. Reading, Turning automata theory into a hands-on course. In *Thirty-seventh SIGCSE Technical Symposium on Computer Science Education*, pp. 379–383. SIGCSE, March 2006.

[2] P. Moura and A. M. Dias, L-FLAT: Logtalk toolkit for formal languages and automata theory. In *Proceedings of the 11th Colloquium on Implementation of Constraint LOgic Programming Systems* (*CICLOPS*), 2011.

[3] S. H. Roger, JFLAP website. www.jflap.org

[4] S. H. Roger, J. Lim, and S. Reading, Increasing interaction and support in the formal languages and automata theory course. In *Twelfth annual SIGCSE conference on Innovation and Technology in Computer Science Education* (*ITiCSE* 2007), Pages 58–62, ACM New York, USA, 2007.

[5] D. A. Norton, Algorithms for testing equivalence of finite automata, with a grading tool for JFLAP. Masters Project, Department of Computer Science, Rochester Institute of Technology, 2009. ritdml.rit.edu/handle/1850/8712

[6] J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation*, p. 281, Addison-Wesley, 1979.

[7] J. Hollingsworth, Automatic graders for programming classes. In *Communications of the ACM*, vol. 3, no. 10, pp. 528–529, 1960.

[8] T. Winters and T. Payne, Computer aided grading with Agar. In *Proceedings of the 2006 International Conference on Frontiers in Education: Computer Science & Computer Engineering, FECS 2006*, Las Vegas, Nevada, USA, 2006.

[9] R. Axelsson, K. Heljanko, and M. Lange, Analyzing context-free grammars using an incremental SAT solver. In *Proceedings of the 35th International Colloquium on Automata, Languages and Programming* (*ICALP* 2008), Part II, pages 410–422, Lecture Notes in Computer Science 5126, Reykjavik, Iceland, 2008.

[10] M. Lange and H. Leiß, To CNF or not to CNF? An efficient yet presentable version of the CYK algorithm. In *Informatica Didactica*, vol. 8, 2009.

[11] J. Cocke and J. T. Schwartz, *Programming Languages and Their Compilers*. Courant Institute of Mathematical Sciences, New York, 1970.

[12] T. Kasami, An efficient recognition and syntax analysis algorithm for context-free languages. Technical Report AFCRL-65-758, Air Force Cambridge Research Laboratory, Bedford, Massachusetts, 1965.

[13] D. H. Younger, Recognition and parsing of context-free languages in time n^3. In *Information and Control*, vol. 10, no. 2, pp. 372–375, 1967.

SESSION

LEARNING MODELS, METHODOLOGIES, TOOLS AND CASE STUDIES

Chair(s)

TBA

The Know-How Kit Software

ATTORNEY MARCELLA COCCANARI
Coccanari & Partners Law Firm –
You Man Right –
City Council of Rome - Schools Commission
Roma, Italy
MR. ROCCO LANATA'
BKC Business Company –
Milano, Italy

Abstract

The project presented here was born as an upgrade of the project presented by the same authors last year in Worlcomp 2012 entitled "New Active Learning Tools". Currently this project is being implemented through a partnership actually born during the Worldcomp 2012 - FECS session with the university URI - Univesidade Regional Integrada – Erechim, RS, Brazil, which submitted a project entitled "URI Online Judge: New Classroom Tool for Interactive Learning".

So the very interesting element of development is the connection and mix of two papers subjected, approved and presented in FECS 2012.

The upgrade of the software project proposed last year is the interaction with an instrument of great success in teaching: the know-how kit. The possible partnership with the University of Brazil founds on the possibility to enclose their "Judge" as a final check to our software and educational tool to be able to actually put it into practice in some Brazilian universities.

The software is able to follow the practical application of the study kits and verifies the final results will appeal to teachers, educators and guides them in the formulation of educational programs effective active ingredients. The software also caters directly to students who want to learn and apply this method to study innovative because they want to raise their standards of quality and accelerate learning.

There is a general belief that the reading of a text or the loquacity of a person, both in teaching and explain concepts in both the exams taken by a student, is sufficient to confirm the actual assimilation and understanding of the studied or show the validity of what is taught. Unfortunately this is a false belief. Then he speaks of "disorders caused by deficiencies of attention" or "learning disability" and the whole thing gets serious when it occurs, as is happening in some cases, the administration of psychotropic drugs as a possible remedy.

Our proposed solution is a practical response which aims to create and strengthen all existing educational system, it is not expensive and is easily transmittable. The software that we propose is to demonstrate the effectiveness of the method.

The final test of teaching and learning is invariably not successfully overcome in oral exams, art or oratory in the personality of skilled teachers, however, are limited to theory, it takes a concrete demonstration of practical competence (especially in technical and science to which we tend to turn to be pragmatic).

Keywords

Teaching
Learning
Effectivness
Know-How
Interacting
Self-improvement
Judge

I. THE START: THE "NEW ACTIVE LEARNING TOOLS" SOFTWARE

Last year, in WorldComp 2012, we proposed the developing of a very specific software addressed to teachers and students, based on a method that is changing the entire vision of teaching and learning. The key word is: proven effectiveness.

We can demonstrate that the same subject given to two different classes, one with our method and one not, the first can be faster and students are able to understand and apply.

In the last 20 years, as instructors of teachers in different countries, and at different levels of education, in short we have observed the following problems:

- Teachers try to apply different methods of teaching, but they don't work uniformly with everybody,
- Teachers cannot fully assist a single student in a classroom,

- Students even if they pass their exams, later don't remember the given information and they are not able to apply it. Besides student's most frequent protest regarding school, is that it is boring, they cannot keep attention more than few hours.
- In connection with many business consultants we have proof that in the workplaces the new workers are enable to show and applied what they were supposed to learned.

In helping the teachers to solve such problems, and with their experiences, we have demonstrate, with tests, documents and exams, that these specifics practical tools are really working; they can be applied in any subject (from math to arts, for increasing technical skills or preparing a person for ESL). We have also several proofs of successfully applications in industries.

Teachers gave us several positive feedback on this method of teaching and learning: they proof to us that the method can be easily learned and applied. The software allow students and teacher to personalize the study path or to guide a distance learning too. In this way the teacher stop to teach to a group and start to supervise the single student.

The active learning tools are quite new on the educational world, they basically consist in three different kind of practical solution to the traditional study:

1) to give the exact definition of the words and the tool to really assimilate them,
2) Suggest always a practical application for study,
3) Utilize a gradual approach to any kind of personal study.

We are not able to give more specifics on the method used as we are waiting the legal approval for using the concerning copyrights (that we are licensed for the use).

Purpose of the software: A) demonstrate to students in which way they can easily understand a subject utilizing the new learning method we propose. B) To give a tool in order to study. Than to study can became fun, easier, faster and most important, effective.

The tool:

- The software at the beginning is programmed to show different subjects that is far away from his/her understanding. The student following the method, learn something about the subject chosen, that he or she first did not like or though it could not be understood. Going through the instructions one can realize that every subject can easily be learned, faster than ever.
- Some test are available for each subject. Students need to proof their competence before entering into the final software section which allows them to learn the given subjects part of the school program.
- The software can verify if the students is doing well or it can bring the person back on the correct gradient (in a particular case the teacher can intervene and assist the student on the method).

- For the practical application, the core of the method, we studied different options that bring the student to study the theory and after proof the real understanding with specifics actions which demonstrates the effective of the learning path.

II. THE STUDY TECHNOLOGY AND ITS RESULTS

The Study Technology developed by American author and educator L. Ron Hubbard.

Study Technology consists of tools and techniques teachers can use to improve the learning rates of students. These same tools and techniques can be used by students themselves to improve their ability to understand and to use the materials they read and study.

There had never before existed a true technology of study that can guarantee high standard of results.

The Study Technology is a breakthrough in self-paced, individualized instructional methodology and in education, that emphasizes, as the name states, a practical technology that is in fact a system of learning how to learn.

Study Technology undercuts the reasons people are illiterate or cannot apply materials they have studied. It is not just another system, it is a workable methodology that makes it possible for a person to recognize and handle the barriers to a successful study.

Purpose of this Study Technology: Converting traditional group lesson plans into personalized learning modules and creating effective and adaptable checksheets for each and every subject..

Several comparative studies and case studies have been conducted on Study Technology programs, documenting their substantial and measurable results. More than 28 million people have now been aided by Study Technology, across some 70 nations, with real emphasys on capability to apply the learned data.

We can demonstrate that teaching the same subject to two different classes, with Study Technology it can be faster and students are able to understand and apply!

Study Technology is not a collection of study tips or memory tricks. It is not speed-reading or the latest method of taking better notes. Rather, in its full range, it is a powerful educational philosophy with clear techniques of application. It provides a system of learning how to learn every and each subject.

THE THREE BARRIERS

A. Absence of mass (physical object) of what is being studied

If one is attempting to understand the function and operation of a car or a computer or a solar system, the printed page and spoken word are no substitute for the object itself. It would be difficult to understand how to use a computer for the first time if you did not have the computer there in front of you. In fact, lacking the object associated with a word can inhibit all understanding. If the mass of a subject is absent, you can actually feel squashed**.** It can make you feel bent, sort of spinny, sort of dead, or bored. A person studying a subject without the objects related to that subject will experience these and several other specific reactions. Knowing how to identify and handle these reactions is vital to a student's ability to grasp and use a subject—and more than vital to a teacher's ability to get a student to learn the subject.

B. Skipped gradient

A gradient is a way of learning or doing something step by step. A gradient can be easy where each step can be done easily, or it can be hard where each step is difficult to do.Too steep a gradient consists of not having mastered prior skills before going on to more complicated or detailed steps.A student who has skipped a gradient may feel a sort of confusion or a feeling of reeling (i.e., moving or swaying like you might fall).These are two reactions a person will have when they have missed a step or hit too steep a gradient in the subject they're studying.This is often referred to as "missed basic skills" or "insufficient basic skills."

C. A word not understood or Wrongly understood

The third and most important barrier is the misunderstood word. "Mis" means *not* or *wrongly*. "Misunderstood" means *not understood* or *wrongly understood*. A misunderstood word is a word which is *not understood* or a word which is *wrongly* understood. Have you ever been reading a book or a report, gotten to the end of the page and couldn't remember what you read? Therein lies the phenomena of the misunderstood word—all becomes distinctly blank beyond a word not understood or wrongly understood. It can make you feel blank or washed out. It can make you feel "not there" and a sort of a nervous upset feeling can follow after that. The matter is far more critical than one might surmise and of the three barriers it is the misunderstood that bears most upon human relations, the mind and understanding. It is the misunderstood word that establishes aptitude—or lack of it. It produces a vast panorama of reactions and is the prime factor involved with stupidity. It also determines whether or not one can actually perform a learned skill, and to what degree of proficiency. All of these are the result of one or more words or symbols not understood or wrongly understood. The misunderstood word can stop a student in his tracks completely. Knowing how to determine when there is a misunderstood word or symbol, how to find it and how to handle it are critical to the success of any student.

III. THE UPGRADE: A FOCUS ON NEW EDUCATIONAL TOOLS

The software that we propose here is combined with a practical educational kit called "Know-How KIT" which is also designed for the occasion and already tested.

Who should the software: To parents, teachers, instructors and students at any level. A business mainly in the productive sectors of the industry.

The aims of the "Know-How KIT": 1) To raise drastically the study results so that a student is able to understand and immediately apply what they studied. 2) Make the most practical and effective teaching in any subject. 3) Raising the productivity and reduce errors in the industry.

The goals of the software: Allow the versatility of the teaching kit and its practical applications. The software, the demonstration kit and the methodology used, they turn to the individual person, not to an entire class (given that understanding to which we are most interested in is that of the individual student). The proposed system makes sure that the student becomes more causative towards learning, and moves the current predominant importance of the teacher (important but not essential if the texts of study are correct and well set and replaceable often with a supervisor expert in this method).

How does "The Know-How KIT *" It is a practical exercise that demonstrates the effectiveness of the Study Technology developed by the educator ** American L. Ron Hubbard to learn and effectively teach any subject. The "Know-How KIT" uses the most effective teaching techniques in the world to learn how to do things, no matter what (our goal is to test it further).

This is not a generic method of study, or of the many approaches to the study, mnemonic or quick read, but a system widely experienced in the world that shows concretely how to teach, understand and do, is comprised of various precise techniques and closer to science than to psychological methodologies and traditional teaching (www.appliedscholastics.org).

The "Know-How KIT" is part of the project "NEW ACTIVE LEARNING TOOLS" based on "Study Technology" Educator L. Ron Hubbard and approved by the "World Academy of Science" (Las Vegas, USA, July 16, 2012, International Conference on Frontiers in Education: Computer Science & Computer Engineering).

IV. THE "KNOW-HOW KIT" A NEW ACTIVE LEARNING TOOL

Suitable for parents, teachers, students of any level, which is useful to increase productivity and reduce errors in the industry.

The exercises in the "Know-How KIT" demonstrate the effectiveness of the foundations of "Study Technology" developed by the educator L. U.S. Ron Hubbard to learn and effectively teach any subject, even that which seems difficult. The "Know-How KIT" uses the most effective teaching techniques in the world to learn how to do things and learn how to study. This is not a generic method of study, or of the many approaches to the study, mnemonic or quick read, but a proven and effective system that shows concretely how to teach, understand, do. This has endless applications in school, after school, in life and in industry to increase productivity and reduce errors.

The goals of the Know-how Kit:

1 - To demonstrate the existence of a study method useful for parents who really want to contribute to the progress of their children.

2 - Making education more practical and effective.

3 - Learn how to study using the basic principles of "Study Technology" by L. Ron Hubbard.

4 - Raising the results in the study so that the person is able to understand and apply immediately what it seems difficult.

5 - Improve productivity and reduce errors of the people in the industry

V. WHAT IS THE “KNOW-HOW KIT” SOFTWARE

The user first chooses a topic from a list that is unknown or even hated, on which the individual person is not prepared or competent: chemistry, electronics, mathematics, molecular cooking etc..

a) The person then reads a prepared text

b) These instructions, deliberately difficult, are not at all understood and the person is not able to "actually do" what is required with the kit (which contains the substances and equipment needed).

c) At this point the student is using the proposed method (taken from the Study Technology) and without further explanation, instructions or suggestions outside, is able to understand the text and to actually do what is required to obtain the final result: a cosmetic product very simple.

d) The cosmetic product that is made with the "Know-How KIT" Chemistry Base is a zinc oxide paste for delicate and sensitive skin, which protects, calms redness or irritation (but the procedure to create each sort of kit: electronics, mathematics and even molecular cuisine and arguments gradually more complicated according to the needs of the audience to whom it is addressed).

The project was first experimented with two groups of 6 each employee of a company in Milan (education financed by Regione Lombardia) and then with 15 instructors association of sailing who have used the "Know-How KIT" Chemistry Base , a chemical formulation for a cosmetic.

DETAILS ON THE PROCEDURE THAT THE SOFTWARE WILL MONITOR '

1) From the main menu is selected the type of kit (point a above).

2) The person reads the proposed text.

3) The "Know-How KIT" is presented and delivered.

4) The student will be directed to a page where answers some questions of an initial test, but is unable to get to work with the kit because he did not understand what he must do.

5) Demonstration of Study Technology - Part One

NB The procedure presented in the proof uses only a fraction of Study Technology, for the exercise of chemistry was deliberately kept a rather low level of difficulty, enough to understand what you can get when you then apply the method a subject of study complete (or instructions).

The software is addressed to individual students (or workers), requires supervision by an experienced and developed in the following exercises:

- The key words in the text are included with a first exercise called "Learn the definitions."
- Keywords are further examined and endorsed with a second exercise called "Question and Answer".
- The procedure is trained with a third exercise called "What are you doing?".
- The student is sent with the instructions, read them in and now fully understands.
- Each student uses the "Know-How KIT" and gives the required result.
- The student is subjected to a final test that confirms its preparation and then writes a final report.

6) Discussion on the Technology of study - Part Two

The software ensures that the person takes possession of some key concepts of the proposed method (at this stage we make use of the project presented last year):

- The first obstacle to learning and the idea of knowing all about it down, the personal goal and motivations that make up the lever to the study.
- The barriers to study: symptoms and their solutions.
- "Solutions to learning disorders and other teaching tools."
- How to completely understand a word or concept. How to teach you to do anything.

7) Practical application addressed to a specific subject

At this point the teacher or the instructor is able to use the teaching method proposed for improving the effectiveness of its courses (any field of teaching or education working in the industry).

The software, through a standard model, guides the teacher or instructor to prepare texts and learning paths using the "Study Technology" in order to make the data more comparable courses and therefore more effective teaching.

VI. THE POSSIBLE INTERACTION WITH THE URI JUDGE SOFTWARE

The possible interaction with the software by URI is currently under study. What we have verified by studying the paper Submitted by URI is that their software is designed to create a judge that it can meet the needs of all students. In our software last year, the new active learning tool, we suppose a process of self-improvement and integration of the software and a continuous interaction and advantageous, as well as addressed the real needs of each individual student. The purpose of participation in the WorldComp was just present our idea, tested and functional in the first phase, or the application of Study Tech L. Ron Hubbard, but also to find possible partners for further development of the software. The Judge offered by URI seems to be the ideal tool to close the circle of our application. Currently we are in contact with the authors of the paper by URI and after WorldComp 2013 will meet them to start in fact a real cooperation.

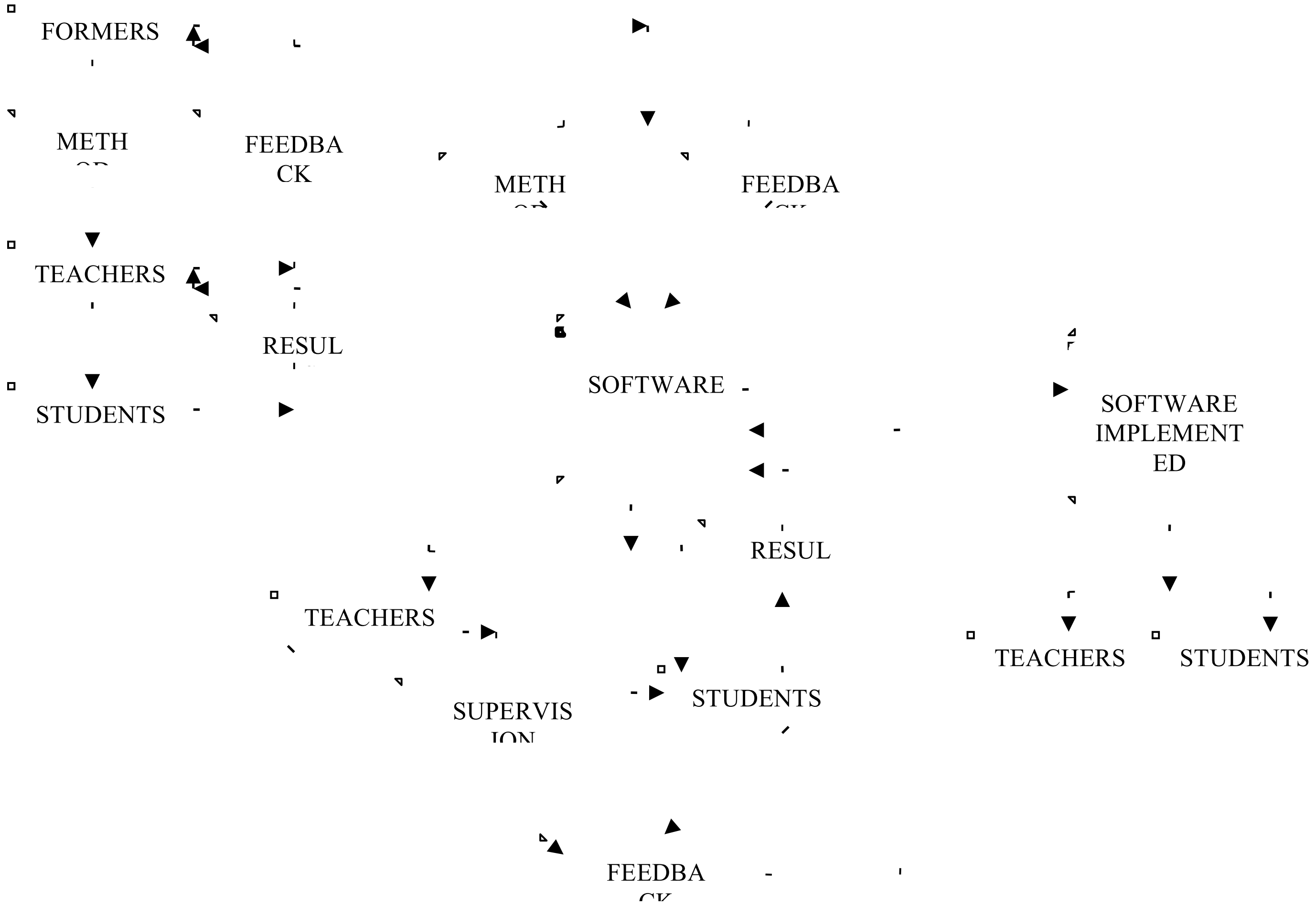


FIG.1 THE NEW ACTIVE LEARNING TOOL SOFTWARE

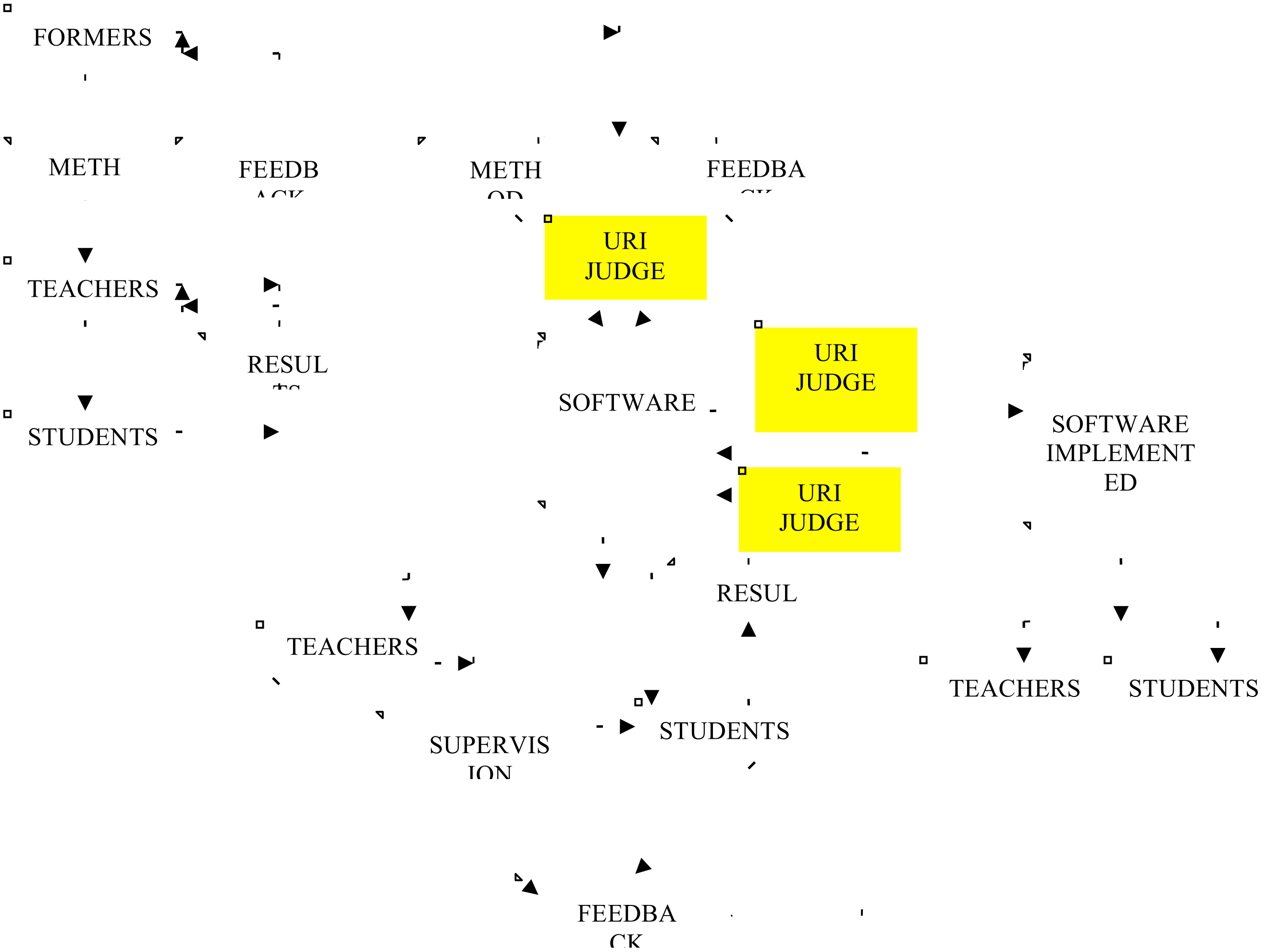


FIG.2 THE NEW ACTIVE LEARNING TOOL SOFTWARE INTERACTED WITH THE URI JUDGE SOFTWARE

COPYRIGHT

ACKNOWLEDGEMENT

Applied Scholastics Italia & Mediterraneo
Applied Scholastics International
(www.appliedscholastics.org)

TONIN Neilor A. - Computer Science and Engineering Department, Universidade Regional Integrada, erechim, RS, Brazil

BEZ, Jean Luca – Computer Science and Engineering Department, Universidade Regional Integrada, erechim, RS, Brazil

EURYDICE Network on Education System and Policies in Europe (www.eurydice.org)

For the "Know-How KIT" chemical base (zinc oxide formulation): Dr. Francesca Picinelli and Prof. Paola Perugini, Coordinator Master's Degree in Science Cosmetologic at the Department of Science of Medicine, Faculty of Pharmacy of the University of Pavia.

REFERENCES

[1]"Barriers to Study" based on the works of L. Ron Hubbard (Applied Scholastics International).

[2]TONI. Neilor A., BEZ. Jean Luca "URI Online Judge: a New Classroom Tool for Interactive Learning" FECS 2012 Vol.I pp.242-246

Cluster Analysis of Chinese International Business Students Based on Learning Styles

Da Huo
Central University of Finance and Economics
No. 39, South College Road
Beijing, China, 100081
dhuocufe@163.com

Yan Chen
Beijing University of Posts and Telecommunications
No. 10, Xitucheng Road
Beijing, China, 100876
itrade8@gmail.com

Abstract

Learning styles are important contributors to the characteristics and performance of students. Especially for the international business program which is newly developed in Chinese higher education system, it is vital to understand the learning styles of students so that courses and teachings can be well organized for this newly developed program. This study surveys students who are enrolled in the international business program, and researched their characteristics by cluster analysis based on their learning styles. This research can be helpful to educations and administrators of international business schools in their future management decisions.

Introduction

With the development of Chinese economy, a number of Chinese companies enhanced their global involvement in international market. To better serve the emerging demand of Chinese economy and Chinese enterprises, the Master of International Business (MIB) Program is officially approved with the foundation of National MIB Education Steering Committee since 2011. The operation of MIB program has become an important issue for a larger number of Chinese universities. And the understanding of Chinese students with their characteristics and learning styles is helpful to Chinese universities in designing and planning their course work.

To have a better understanding of the Chinese international business students with their learning styles, the research launched a survey to Chinese students involved in Master of International Business Program with Kolb Learning Style Inventory- 3.0 (Kolb, 1999). Cluster analysis is used to characterize the Chinese students by their different learning styles. The result of this research reveals important information that leads a way for Chinese international business students to be well trained for their future work and services to the global business area.

Literature Review

Dewey (1897) claimed that the process and goal of education are firmly related to each other. The process which enhances the students' learning is important to the improvement of higher education. Kolb (1984) defined that knowledge is created by the transformation of experiences. And the learning style refers to individual differences in learning behaviors and characteristics based on preferences of different phases in learning styles.

The learning process can be defined by two pairs of learning competences (Kolb, 1981). Abstract Conceptualization (AC) and Concrete Experience (CE) function to grasp experiences, while Active Experimentation (AE) and Reflective Observation (RO) contribute to experience transformations. Zull (2002) pointed out that the two pairs of functions in learning process are related to the function of human brains. The creation of new abstract concepts occurs in the frontal integrative cortex, and concrete experiences are formed by the sensory cortex. Active testing is functioned by the motor brain, while reflective observation involves the integrative cortex at the back. Kolb & Kolb (2005) studied the differences of preferences between abstract conceptualization (AC) and concrete experiences (CE), and also differences of preferences between active experimentation (AE) and reflective observation (RO) based on the learning styles of U.S. students. Identified by differences between AC and CE, as well as differences between AE and RO, the study characterized the students' learning style into four categories such as diverging, assimilating, accommodating, and converging.

The research of how do students learn from the course and trainings implies a great value to the Chinese universities in their MIB education systems. With the study of different competences of Chinese international business students, courses and trainings can be well developed. Understanding the different preferences of abstract concepts and concrete experiences is helpful to find out the thinking process of Chinese students in international business program. The research of active testing and reflective observation is also helpful to reveal the differences of Chinese students in their learning practices.

Research Method

This study used Kolb Learning Style Inventory- 3.0 and surveyed 61 master students involved in the international business program of China. Students are asked to grade different options of learning styles due to their preferences. 4 stands for the most preferred and 1 stands for the least. Different learning preferences such as abstract concepts, concrete experiences, active testing, and reflective observations are all included in the questions asked to the students. Cluster analysis is operated by SAS 9.2 to characterize different learning styles of Chinese MIB students. The grading of four different learning preferences is standardized and the Chinese students are characterized by four different clusters.

Result

Table 1 shows that Chinese international business students are characterized into four different learning styles. Cluster 1 is the group that has a high value in abstract concepts and active testing, but low in concrete experiences and reflective observation. Cluster 2 has a higher level of concrete experiences and reflective observation, but low in abstract conceptualization and active experimentation, which is opposite to Cluster 1. Cluster 3 is higher in both abstract conceptualization and concrete experiences and reflective observation, but only low in active experimentation. Cluster 4 is relatively higher in concrete experiences and active experimentation, but low in abstract conceptualization and reflective observation. Table 2 shows the R square of the cluster analysis and Table 3 shows the initial seed of each group of students.

Table 4 explains the cluster distances of different learning styles. It is found that most of students are characterized by Cluster 1, relatively high in abstract conceptualization and active experimentation, but low in concrete experiences and reflective observation. Both Cluster 2 and Cluster 4 are neighborly positioned to Cluster 1. Cluster 2 is opposite to the characteristics of Cluster 1, while Cluster 4 is different from Cluster 1 in learning preferences of abstract conceptualization and concrete experiences, but similar in active experimentation and reflective observation. Cluster 3 is mostly different from other three groups, and only a limited number of students are in this group.

Conclusion

The research of Chinese international business students revealed the learning styles of different students based on four different learning competences, including abstract conceptualization, concrete experiences, active experimentation and reflective observation. It is found that most Chinese students involved in Master of International Business Program are high in abstract conceptualization and active experimentation. This shows that students are good at abstract thinking, and also have a good preference of experimentation. This is helpful for students to transform what they have learned in their practices. The training for these students could be more emphasizing the concrete experiences and reflective observations. However, there are also a number of students in the opposite case. The program also needs to include the course works that improve the abstract conceptualization and active experimentations. Concerning the co-existence of both two different primary learning styles represented by Cluster 1 and Cluster 2, the international business program could be a composite of knowledge including different competences, and the team work assignments also can be a good way for both kinds of students to improve their learning competences.

Reference

Dewey, J. 1897. My pedagogic creed. The school journal. LIV(3), 77–80.

Kolb, D. A. 1981. Experiential Learning Theory and the Learning Style Inventory: A reply to Freedman and Stumpf. Academy of Management Review, 6(2): 289–296.

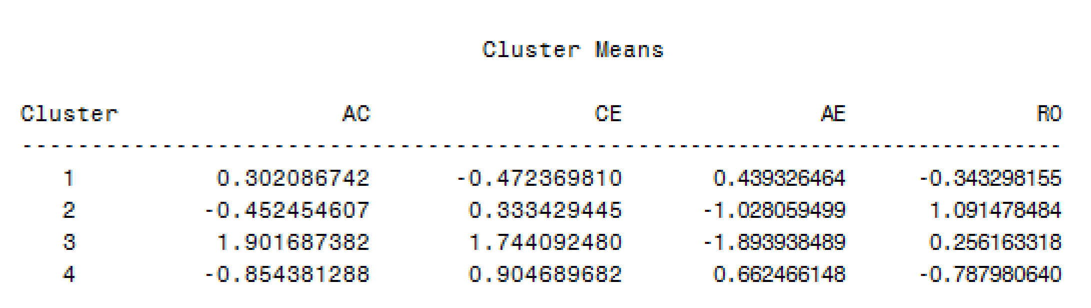
Kolb, D. A. 1984. Experiential learning: Experience as the source of learning and development. New Jersey: Prentice-Hall.

Kolb, D. A. 1999. Learning Style Inventory-Version 3: Technical specifications. TRG Hay/McBer, Kolb, Alice Y. & Kolb, David A. (2005) Training Resources Group Learning Styles and Learning Spaces: Enhancing Experiential Learning in Higher Education

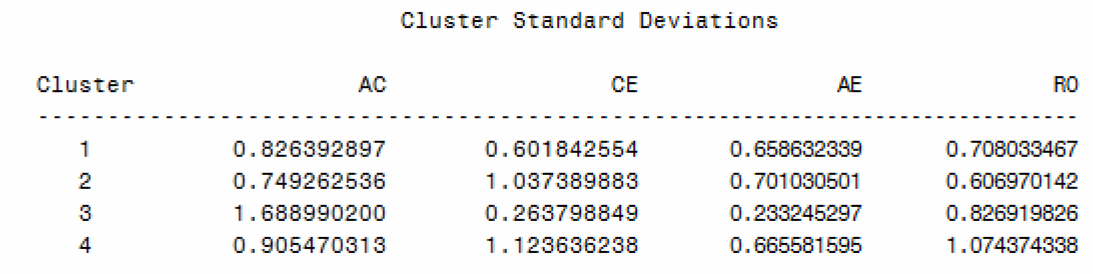
Zull, J. E. 2002. The art of changing the brain: Enriching teaching by exploring the biology of learning.

Appendix

Table 1. Descriptive Analysis of Surveyed Students

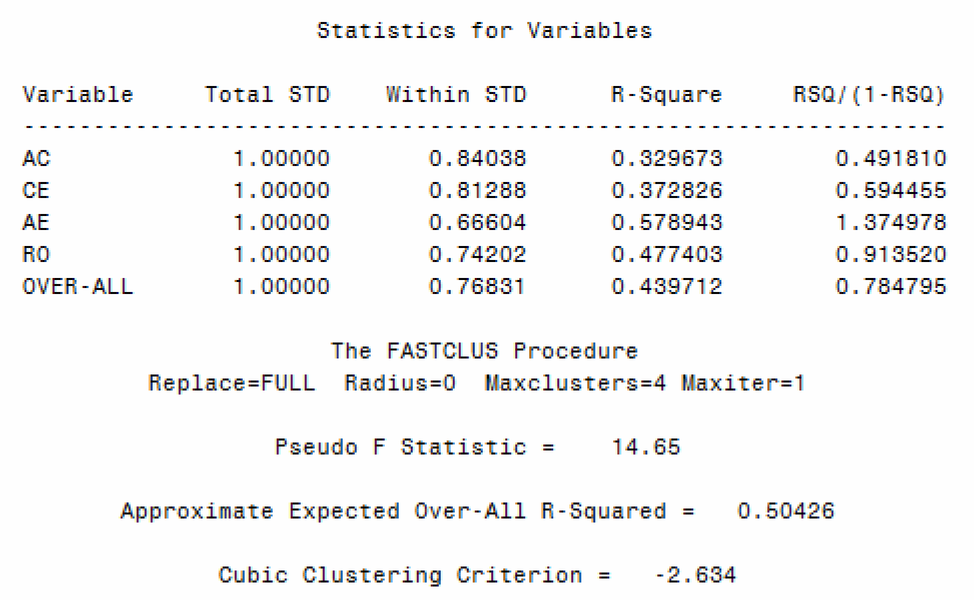


```
                              Cluster Means

Cluster                 AC                CE              AE              RO
----------------------------------------------------------------------------
   1           0.302086742      -0.472369810     0.439326464    -0.343298155
   2          -0.452454607       0.333429445    -1.028059499     1.091478484
   3           1.901687382       1.744092480    -1.893938489     0.256163318
   4          -0.854381288       0.904689682     0.662466148    -0.787980640
```



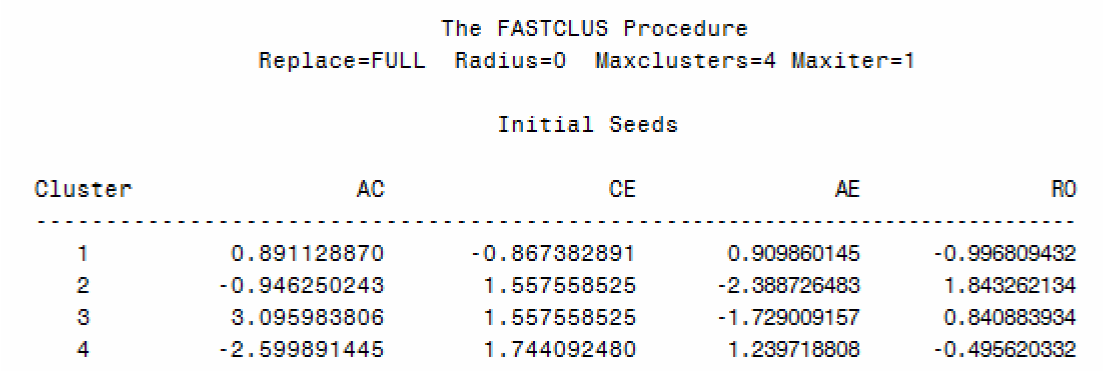
```
                        Cluster Standard Deviations

Cluster                 AC                CE              AE              RO
----------------------------------------------------------------------------
   1           0.826392897       0.601842554     0.658632339     0.708033467
   2           0.749262536       1.037389883     0.701030501     0.606970142
   3           1.688990200       0.263798849     0.233245297     0.826919826
   4           0.905470313       1.123636238     0.665581595     1.074374338
```

Table 2. Statistics of Cluster Analysis



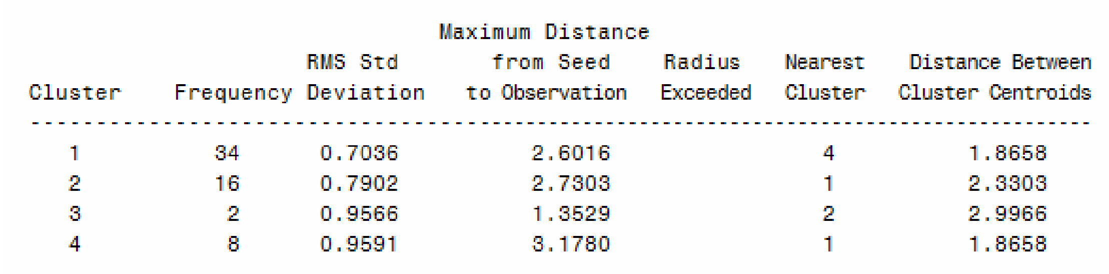
```
                   Statistics for Variables

Variable      Total STD    Within STD      R-Square     RSQ/(1-RSQ)
-------------------------------------------------------------------
AC              1.00000       0.84038      0.329673        0.491810
CE              1.00000       0.81288      0.372826        0.594455
AE              1.00000       0.66604      0.578943        1.374978
RO              1.00000       0.74202      0.477403        0.913520
OVER-ALL        1.00000       0.76831      0.439712        0.784795

                    The FASTCLUS Procedure
         Replace=FULL  Radius=0  Maxclusters=4 Maxiter=1

                  Pseudo F Statistic =     14.65

        Approximate Expected Over-All R-Squared =   0.50426

               Cubic Clustering Criterion =   -2.634
```

Table 3. Initial Seed of Learning Preferences



```
                          The FASTCLUS Procedure
           Replace=FULL  Radius=0  Maxclusters=4 Maxiter=1

                              Initial Seeds

Cluster                 AC               CE              AE              RO
----------------------------------------------------------------------------
   1           0.891128870     -0.867382891     0.909860145    -0.996809432
   2          -0.946250243      1.557558525    -2.388726483     1.843262134
   3           3.095983806      1.557558525    -1.729009157     0.840883934
   4          -2.599891445      1.744092480     1.239718808    -0.495620332
```

Table 4. Cluster Distances of Learning Preference



```
                                  Maximum Distance
                        RMS Std      from Seed     Radius    Nearest   Distance Between
Cluster      Frequency Deviation  to Observation  Exceeded   Cluster   Cluster Centroids
-----------------------------------------------------------------------------------------
   1             34     0.7036        2.6016                    4           1.8658
   2             16     0.7902        2.7303                    1           2.3303
   3              2     0.9566        1.3529                    2           2.9966
   4              8     0.9591        3.1780                    1           1.8658
```

Interactive Learning to Improve Student Success Rates in Teaching Programming: A Case Study

Said Fares[1] and Mary Fares[2]
[1]Department of Mathematics and Computer Science, Valdosta State University, Valdosta, USA
[2]Department of Astronomy, Physics, and Geosciences, Valdosta State University, Valdosta, USA

Abstract - *It is known that introductory computer programming courses are difficult and that failure rates are high [1, 2, and 3]. The aim of this project was to improve student success rates in learning to program. This paper presents a number of changes in module organization and instructional delivery system. Interactive lectures and laboratory sessions, online resources, timely feedback, and out of class help sessions and support were applied. The primary results indicate a positive evaluation of the modified instructional delivery system, overall satisfaction with the course and consequently, a higher success rate.*

Keywords: CS1; Learning Programming; Online Learning; Success Rates.

1 Introduction

Programming is the core of computer science, and therefore most computer science programs start with introductory programming courses. However, regardless of the recognized importance of programming, the results are often disappointing. Deficiencies of basic programming skills were reported by instructors of the upper-level courses. Another consequence of poor learning results was high failure rates in introductory courses. Many schools report dropout rates of 20 to 40 percent, even higher, of students on their introductory programming courses [3, 4]. Pedagogical approaches, which take advantage of learning theories and information technologies, have been proposed in the research literature to tackle the learning problems associated with introductory computer programming [5, 6, 7, 8, 10, and 11]. However, there are very few evidence-based experiences and the difficulties of learning how to program for novice students remain to be researched [6]. As blended learning becomes more and more pervasive in higher education as the most prominent delivery mechanism, expectations for learning benefits in computer programming are becoming greater. But, just providing educators with a mix of face-to-face learning and information technologies, will not have the desired effect, if the underlying blended learning model does not rely on learning theory and pedagogical principles [9].

To tackle this problem, a pilot project was developed to revise our traditional face-to-face lecturing approach in teaching CS1. This interactive model was supported by the availability of information technologies in an attempt to improve the overall student satisfaction, retention, and success rates.

2 Description of the Module

Introductory Java programming is taught to over 60 undergraduate students a semester at Valdosta State University, where more than 95% of them major in computer science or computer information systems. Up to fall 2011, we had applied the traditional method of teaching the introductory programming course. Table 1 highlights the main components and features of the traditional approach and the new interactive method of learning.

TABLE 1. Interactive vs. Traditional Method

	Components	Traditional Method	Interactive Approach
1	Face-to-face Lectures	✓	✓ With Student Engagement
2	Programming Assignments	✓	✓
3	Testing	✓	✓
4	Quizzes		✓ Weekly
5	Laboratory	Occasionally	✓ Twice a Week
6	Online Resources		✓
7	Self-Paced Practice Exercises		✓
8	Help Sessions	Occasionally Individual tutoring	✓ Weekly
9	Progress Monitoring	Midterm grades only	✓ Weekly
10	Student-Instructor interaction	Limited to class time and office hours	✓ Chatting and emailing
	Components	Traditional Method	Interactive Approach

After careful analysis of the existing teaching approach, the new module was developed with the following components.

2.1 Face-to-face Lectures

Modified regular classroom lectures were introduced to present class materials, using slides presentations, board, live developing, debugging, and running sample programs with students' active involvement and participation.

2.2 Online Delivery System

An extensive online content were developed to support student learning that were housed on WebCT and available 24/7. The online components of the module of the course included (1) A course web site, organized by Power Point Slides, Assignments, sample code, test data and executable versions of assigned programming projects, and (2) A large test bank of self-paced exercises with instant and automated feedback to students.

2.3 Laboratory

The class met in the laboratory twice a week, where each student had access to her/his computer. Additionally, the laboratory had instructor's machine with a classroom projector. The laboratory was used for instruction, practice, and testing.

2.3.1 Interactive Programming and Practice

After a brief description by the instructor of the lab programming project, students worked individually or in groups to design and implement the solution of the programming task. Individuals and teams shared their findings with the class by presenting their solutions or demonstrating their work on the class' projector. At the end of the period, sample solutions were posted on the course web site for all students to study, evaluate, and compare.

2.3.2 Quizzes

In order to encourage student learning, two types of weekly laboratory quizzes were given. The non-programming quizzes were given to encourage students keeping up with class materials, while, the programming quizzes were intended to give students enough programming practice and to enforce learning to program. During the programming quizzes, students developed programs by using an IDE and submit their work online by the end of laboratory period. To encourage students to complete the project and understand the concepts, late submissions were allowed but graded out of 50% of the total grade. More than 90% of the students were able to submit their work on time. Live coding quizzes were given with open book, online search, and open notes.

2.3.3 Help Sessions

Programming concepts and skills are related and some are prerequisite of others. Students, who do not understand early the introductory programming concepts in the semester, face a great danger of not catching up with the materials and end up dropping or failing the course. In order to help students in this critical transition, we offered students weekly laboratory help sessions. These help sessions provided help to individuals with their programming problems as well as covered common subjects that students were facing difficulties with. The help sessions presented a timely support for serious students who faced a "brick wall" in their transition period. Students' attendance of these voluntarily activities showed the effectiveness of this early intervention.

2.3.4 Progress Monitoring

Combinations of tools were applied to monitor students' performance and identify students-at-risk early enough to intervene. WebCT system tracking data of student activities, class attendance, and students' weekly progress report were used in our intervention. Post-questionnaire results showed that 95% of the students found the weekly performance report was helpful.

2.3.5 Interaction

In order to promote Student-instructor interaction, a number of strategies were applied.

2.3.5.1 Explaining

Description, discussion, and summarization of programming projects and solutions were provided to help students with practical ideas to apply to new programming applications.

2.3.5.2 Program reporting

To encourage student reflection, students were required to submit a one-page project report with each programming assignment stating the purpose of project, skills learned, problems faced, and their observations and suggestions.

2.3.5.3 Precise grading

The instructor used WebCT to grade online student programming projects with precise comments and suggestions.

2.3.5.4 Prompt chatting and emailing

In addition to personal interaction with instructor, students used WebCT chatting and emailing in further communicating with instructor and received guidance and suggestions in their class activities and programming projects.

Program reports, precise grading, and chatting and emailing provided an active form of student-instructor communication and engagement.

3 Evaluation

A detailed evaluation of the interactive approach was conducted to study the effectiveness of the new learning environment and its major components. Two survey questionnaires were employed to study student's learning.

3.1 Pre-questionnaire

A survey provided to the students after four weeks of the semester. This was designed to collect data related to the activities that were going well, issues that needed to be improved, instructor's feedback, students' overall satisfaction with the course, and students' suggestions for better learning.

3.2 Post-questionnaire

An extensive survey was given to the students one week before the end of the semester. Technical evaluation issues addressed the extent to which the in class lectures, laboratory, and online resources helped the students to learn programming. Also, pedagogical evaluation issues addressed the extent to which the interactive learning model provided support to the learning process.

The questions of the survey were grouped into three categories: (1) Overall student satisfaction with the course, (2) Usefulness of the course resources, and (3) Effectiveness of student –instructor interaction and feedback.

Table 2 shows that 89% of the students strongly agreed/agreed that the course advanced their learning to program, 81% of them strongly agreed/agreed that the course increased their interest in the computer science field, and 80% of the students strongly agreed/ agreed that the out of the class help sessions benefited them in learning.

TABLE 2. Students Satisfaction

Question	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Overall, This course has been effective in advancing my learning	49%	40%	9%	0%	2%
This course has increased my interest in the CS field	36%	45%	13%	2%	4%
I have learned a lot in this class	57%	32%	9%	0%	2%
The Help Sessions that I was able to attend have helped me	42%	38%	21%	0%	0%
The Weekly Quizzes have encouraged me to study more And benefit my learning	30%	54%	13%	2%	2%
The Pre-set due dates and times for assignments every Thursday at 11:00 PM Have helped me	57%	24%	13%	6%	0%

Figure 1 indicates that 100% of the students were satisfied with classroom discussions and programming examples, 77% of them were satisfied with programming projects, and 77% were satisfied with lectures and online resources, and only 36% of the students were satisfied with power Point presentations.

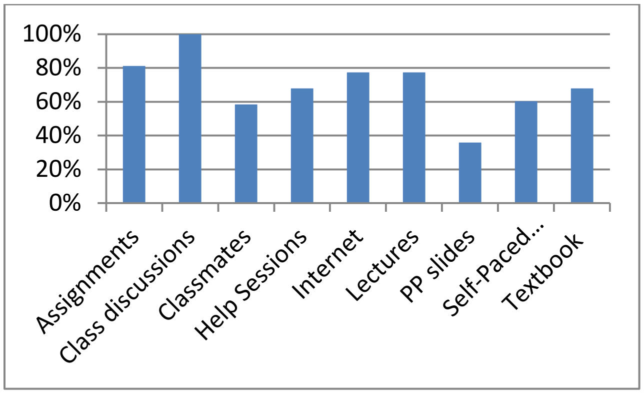


Figure 1. Usefulness of course resources

Table 3 shows students' perceptions of the effectiveness of student –instructor interaction and feedback. 90% of the students strongly agreed/agreed that prompt grading of assignments and exams benefited their learning, 96% of them strongly agreed/agreed that Blaze View (WebCT) announcements helped them stay in touch with class activities, while, 74% of them strongly agreed/agreed that the prompt email communications with the instructor helped them understood the class materials by answering their questions.

TABLE 3. Student – Instructor Interaction

Question	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Assignments are graded and returned quickly enough to benefit my learning	36%	57%	6%	2%	0%
Exams are graded and returned quickly enough to benefit my learning	30%	60%	6%	4%	0%
Announcements on Blaze View have kept me informed with what was going on in the class	59%	37%	4%	0%	0%
The prompt communications with instructor via email has helped me	22%	52%	22%	4%	0%

Table 4 shows the different grades awarded, passing/failing rates, and the changes from spring 2011 to spring 2012.

TABLE 4. Percentages of Awarded Grades and Changes

Grades	Spring 2011	Spring 2012	Change
As	14%	10%	-31%
Bs	7%	24%	245%
Cs	14%	22%	61%
Ds	21%	37%	76%
Fs	14%	2%	-88%
Ws	31%	6%	-80%
ABCs	34%	56%	61%
DWFs	66%	44%	-32%

Figure 2 and Figure 3 display the three significant improvements of the new approach over the traditional method of teaching.

1) An increase of 61% of the success rate (grades of A,B, and C)
2) A decrease of 88% of the failure rate (grades of F)
3) A decrease of 80% of the withdrawal rate (grades of W)

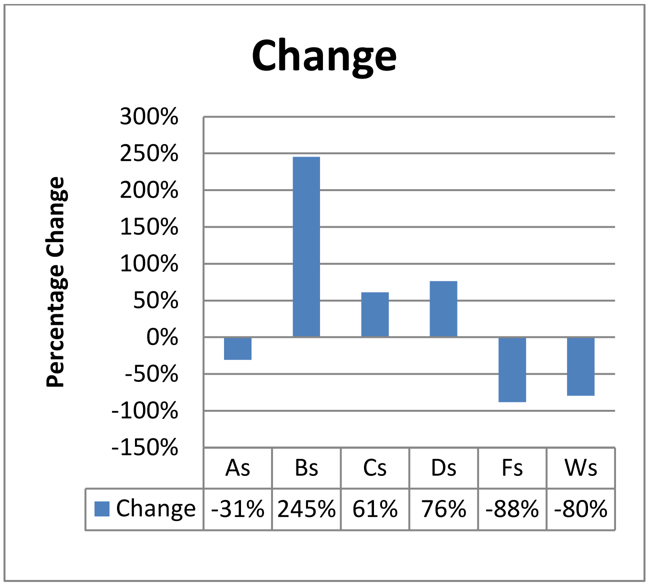


Figure 2. Percentage changes of awarded grade rates.

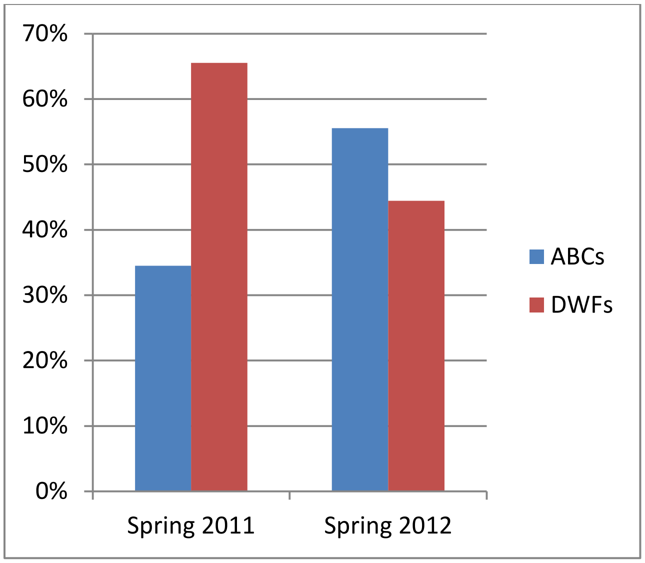


Figure 3. Percentage changes of ABC and DWF

It is also useful to point out two facts: (1) the course requirements and subjects taught in the course remained virtually unchanged from the old model to the new approach, and (2) students with the new approach scored 20% higher on an identical comprehensive final exam used in the previous model.

In addition to these issues, the students were asked whether they would recommend the course to other students. 89% of the students said that they would recommend the course to other students.

4 Conclusions

The paper has reported on the interactive learning approach to tackle the problems in teaching the introductory programming courses such as high drop out and failure rates. The preliminary results showed significant improvements in success rates and positive students' overall satisfaction. The new module is a departure of the "Swim or Sink" practice of the traditional method of teaching. Evidence from increased student class attendance and high utilization of the online resources increased student engagement and subsequently, lowered the student withdrawal rate.

As a proof-of-concept study, the data presented here is based on only two sections of CS1, which makes statistical analysis more difficult. This interactive approach will be applied to other programming, engineering, physics, and science courses to provide further validation for the interpretation drawn here.

5 References

[1] Jens Bennedsen, Michael E. Caspersen. “Failure Rates in Introductory Programming”. SIGCSE Bulletin, Vol. 39, No. 2. 2007.

[2] Bennedsen, J. and Caspersen, M.E. “An Investigation of Potential Success factors for an Introductory Model-Driven Programming Course”. Proceedings of the First International Workshop on Computing Education Research (ICER ’05). ACM. 2005.

[3] Päivi Kinnunen, Lauri Malmi. “Why Students Drop Out CS1 Course?”. ICER, Canterbury, UK, 2006.

[4] Rountree, N., Rountree, J., Robins, A. & Hannah, R. “Interacting Factors that Predict Success and Failure in a CS1 Course”. SIGCSE Bulletin, Vol. 36, No. 4. 2004.

[5] Tom Boyle, Claire Bradley, Peter Chalk, Ray Jones & Poppy Pickard. “Using Blended Learning to Improve Student Success Rates in Learning to Program”. Journal of Education Media, Vol. 28, 2003.

[6] Said Hardjerrouit. “Towards a Blended Learning Model for Teaching and Learning Computer Programming: A case Study”. Informatics in Education, Vol. 7, No. 2, 181-210, 2008.

[7] T. Grandon Gill and Carolyn F. Holton and Said Hardjerrouit. “A Self-Paced Introductory Programming Course”. Journal of Information Technology Education. Volume 5, 2006.

[8] Anthony Robins. Learning Edge Momentum: “A New Account of Outcomes in CS1”. Computer Science Education Vol. 20, No. 1. 2010.

[9] Nocols, M. “A theory of e-learning”. Educational technology and Society. Vol. 5, No. 2, 2003.

[10] Wilson, B.C. & Shrock, S. “Contributing to Success in an Introductory Computer Science Course: A Study of Twelve Factors”. ACM SIGCSE Bulletin, Vol. 33, No. 1. 2006.

[11] Woszczynski, A., Haddad, H., & Zgambo, A.: “Towards a Model of Student Success in Programming Courses”. Proceedings of the 43rd Annual Southeast Regional Conference. Vol. 1. ACM-SE 43), 2005.

The Design of Open Learner Model to Improve Interaction of Peer Assessment in Learning

Qingchun Hu[1], **Yong Huang**[2]

[1]School of Information Science & Engineering, East China University of Science and Technology

[2]Shanghai Audio-video Education Center, Shanghai Distance Education Group, Shanghai, China

Abstract - *Peer assessment has been reported as an effective collaborative learning approach. However, it is very difficult to encourage students to engage in reviewing activities. This study puts forward an open learner model approach that helps students' peer interaction in the context of online peer assessment. A framework is proposed which structures and represents the information of peer interaction including Issue, Position, Argument and Feedback. These characteristics aim to support students to recognize and target problem areas. Once targeted, on the one hand, the system can suggest strategies to help students collaborate more effectively with their peers, maximizing students' learning performance. On the other hand, students can access peers' learner models and look for feedback on their work in a group. The result will provide strategies and suggestions to encourage peer interaction in the context of online peer assessment effectively.*

Keywords: Peer Interaction, Open Learner Model

1 Introduction

There are many approaches to organize online peer assessment. However, we find it is very difficult to encourage students to engage in reviewing activities. In our previous study, we had developed and implemented a two-round online peer assessment system to support students' learning in programming course [1, 2]. Some students complained that the feedback was not helpful or even inefficacy, such as, "Good, you are right" or "Excellent!". The study of Lin et al. also reported that most students only sent simple feedback [3]. In the circumstances, learning performance is not satisfying. The goal of online peer assessment is not just for assessment by students themselves according to the rubrics, but also for enhancing students' learning through collaborative learning activities. From Vygotskian perspective [4], learning is a kind of social activity in which peer interaction plays an important role. Thus, making students have more peer interaction should be one way to enhance learning performance in the context of online peer assessment.

Students need appropriate guides to be engaged in peer interaction. Tutors can provide encouragement and guide students in time through observing students' behavior and learning performance in the classroom environment. In online learning environment, the online system could play a teacher role (as a tutor agent) and guide students' peer interaction based on learner models which record students' learning history and learning process. The study proposes a framework to design open learner models, which aims to structure and represent the information of peer interaction in the context of online peer assessment.

2 Review of related studies

Most approaches of peer assessment ask students to post comments on peers' work in a summative way in one-round, including giving a final grade and posting comments [5]. In this case, there is less interaction among peers.

In fact, peer assessment is a kind of collaborative learning, where peer interaction should play an important role in improving students' learning. Here, peer interaction means one student communicate with another around issues or topics in a group. However, putting students in a group does not guarantee them to engage in peer interaction. They may come across difficulties, such as less feedback, or even the peer missing. Thus, peer interaction should be organized properly by tutors. Especially, in a context where the teacher has less control than in a face-to-face setting, the teacher role is necessary. In online learning, the system can play as a teacher agent to provide guidance based on the students' learning.

Then, how to record students' learning by the online system? The learner model (or student model) is a kind of mechanism that records and diagnoses students' learning, including learning performance, knowledge level, misconceptions etc.[6, 7]. When the learner model is open to students, it is named as open learner model. An open learner model is one with specific provisions for the learner to have one or more views of the information in the model [8]. It has been reported that open learner model should help students improve individual reflection. Moreover, there are also studies that explore learner models used in collaborative learning, which enable students interact with the peer agent to improve students' learning [9].

Since the learner model is a kind of mechanism to record students' learning, then how to record and support the peer interaction? Peer interaction is a complex cognitive process of human being. It is difficult to obtain precisely qualitative and quantitative analyses towards the discussion activity. One reason is the discussion activity is unstructured. Therefore, it is necessary to propose a framework to construct the interaction and facilitate the computer to record the information. We find that the IBIS framework is suitable for this goal.

The Issue-Based Information System (IBIS) is proposed by Werner Kunz and Horst W. J. Rittel in order to organize the process of policy decision through discussion in CSCW environment (Computer Supported Cooperative Work)[10]. IBIS offers a framework to record information in a discussion process. It consists of a network of essentially three types of nodes: issue, position, argument. The issue is the organizational "atoms" of IBIS. The position is the response to issue(s) taken by somebody. The argument means somebody support or oppose positions.

There have been examples of IBIS type of information systems used in collaborative learning for students' online group work. For instance, Karacapilidis et al. adopted IBIS framework as an argumentation tool when students participated in solving course exercises online[11]. The study of Liu and Tsai showed an analysis of peer interaction patterns based on IBIS in learning programming course [12]. Their study showed that IBIS was a suitable way for analyzing the students' discourse in online group learning in programming course.

So, IBIS is a suitable way to structure and analyze the process of discussion by the computer. The benefits of IBIS include:

(a) Help students be aware of the process of discussion

Although IBIS is a simple structure, it helps students know the key stages in their discussion. This likes the open sentences that used in the Knowledge Forum (http://www.knowledgeforum.com/), which help scaffold the discussion. The structure of IBIS makes students realize how to engage in effective discussion.

(b) Help the information structured and saved

In generally, the process of peer interaction is very complex and unstructured. If the information is unstructured, then, the log files of peer interaction saved in the database may be disorganized. And the significant effort is required to extract communicative relationships within the group. If the information is organized according to a kind of structure, it will facilitate the analysis by the computer.

(c) Help the information reused

The following students can retrieve the issues from the database and browse the process of peer interaction. It is a good resource for students' future study.

Our study extends the IBIS, putting forward Issue-Position-Argument-Feedback (IPAF) as the framework to structure the peer interaction in the design of open learner model. The open learner model (Technology) is the vehicle for providing support to improve peer interaction (Pedagogy).

3 Issue-Position-Argument-Feedback Framework

In our study, the peer interaction in the online peer assessment is structured of issue, position, arguments and feedback (See Figure 1).

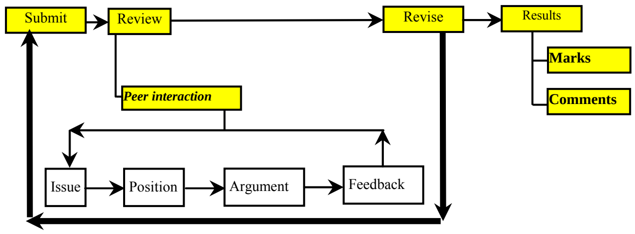


Figure 1. Peer Interaction Involved in Online Peer Assessment

3.1 Issue

Assigning students in a group with peer reviewing tasks does not guarantee that the students will engage in effective peer interaction. Perhaps, some peer groups seem to interact naturally, others struggle to maintain a balance of participation and encouragement. Thus, it is necessary that providing issues for students and the issues should be designed carefully.

Here, the issue means that a subject or problem that is discussed or argued about. In our design, the issues come from two ways. One is from the teacher, which is designed to stimulate students to consider the domain knowledge related to the assignment. Another is from students themselves. The students will have opportunities to post issues by themselves. If students post issues and this issue stimulate more interaction, then, the student will be awards. All the issues will be saved in a database for students' future learning as issues repertory.

3.2 Position

The position means a point of view on a certain question or issue. Generally, there are three kinds of positions----agree, oppose, other. And students have opportunities to show their position if they do not like select "Agree" or "Oppose".

3.3 Argument

The argument means a debate, or a statement put forth as proof or evidence. Because, the issues that students discussion are high related to their current assignments. Thus, students are encouraged to provide a statement or reason towards their assignments.

3.4 Feedback

In our design, every student has responsibility to post comments and rank on peers' arguments. It is a compulsory task. The comments and rank students get is named as "feedback". Since the issue is high related to their current assignments, and perhaps, the argument will provide a solution. Based on the feedback, students could know whose arguments are very useful for their assignments.

The rubrics for commenting on peers' arguments are based on the study of Webb [13]. It consists of five criteria for ensuring that students provide effective help to their peers. These criteria include: "help is timely; help is relevant to the student's need; the correct amount of elaboration or detail is given; the help is understood; the student has an opportunity to apply the help in solving the problem (and uses it!)". Based on the criteria, we put forward a rubric for commenting on peers' arguments. The goal is to look whether the arguments are help for students' assignments (Figure 2).

·The argument is related to the issues.	5	4	3	2	1
·The argument is help to promote the discussion.	5	4	3	2	1
·The argument is help for your assignments.	5	4	3	2	1

Figure 2. A rubric for students commenting on peers' arguments.

All above show the nodes designed in peer interaction. We will use this framework to input (recording the peers' interaction), extracting communication and determining the support provided to students.

4 A Framework of Supporting Peer Interaction by Open Learner Models

When the learner model is open to students, we need consider what information should be presented to students. That is to say, how to externalize and visualize the learner model. The methods of externalizing of learner model are varied, such as a graphical externalization of a Bayesian network [14], a fuzzy logic method [15]. Among the methods, the skill meters remain the most common form of open learner model, which can illustrate knowledge level, difficulties, misconceptions. The study of Bull and Kay suggested an open learner model which consists of four parts: "how does the open learner model fit into the overall interaction and how was it evaluated? what is open? how is it presented? Who controls access? " [16]

Based on previous studies, in our design, the open learner model consists of two parts: cognitive support and social support. It represents students' knowledge, misconceptions and interaction with the system and peers (Table 1).

4.1 Cognitive Support

The cognitive support aims to help students reflect their learning performance. It represents the domain knowledge of students themselves, including the students' current learning performance and their learning progress.

4.2 Social Support

The goal of social support is to help students reflect their interaction process. The information of students' peer interaction is described by four aspects: issue, position, argument and feedback. Here, the supporting strategies designed are named as rules.

(a) Issues

There are two kinds of issues in the design. One is from tutors. Another is from students themselves. The strategies are listed below.

▪Rules-related to Issues 1 (RI-1): show students concepts related to the issues.

Perhaps, some students do not understand the meaning of the issues. Thus, the system provides prompts about issues, including the explanation about the issues, some concepts related to the issues.

▪Rules-related to Issues 2 (RI-2): give award to students who launch an issue.

Students should have opportunities to launch an issue. And, perhaps, these issues are what they are interested in.

(b) Position

▪Rules-related to Position 1 (RP-1): provide a checklist for students to post their position.

The system will provide a checklist for students to show their positions, including agree, oppose and other.

(c) Argument

▪Rules-related to Arguments 1 (RA-1): provide an example for students on how to argument.

Perhaps some students have difficulties to launch an argument to express their thought, thus, the system will provide examples for students to browse.

(d) Feedback

As mentioned above, in order to make students have effective feedback, students are asked to post comments and rank on peers' arguments to assess whether they get benefits from students' assignments.

▪ Rules-related to Feedback 1 (RF-1): provide rubrics to students to comment on their arguments.

It is to ensure that each student's discussion is high related to their demands. We put forwards a rubric for students to comment on their arguments. The main point is to look whether the arguments is help for students assignments (Figure2).

Table 1. The Design of Peer Interaction in Open Learner Model

Open Learner Model **Peer Interaction**	What open		Why open	How open
Cognitive support	Test score Learning progress		Show students current learning performance (Know their ZPD)	Bar chart
	Errors posted by peers Peers comments Marks		Show the learning status (Help to reflect their learning)	Text
Social support	Issues	▪Rules-related to Issues 1 (RI-1): show prompts for concepts related to the issues. ▪Rules-related to Issues 2 (RI-2): show the students progress on rewards by posting issues.	Improve interaction and make reflection in the process of peer interaction (30% of the mark come from peers rating on the arguments, 70% of the mark come from the teachers)	Rating meter, Text
	Position	▪Rules-related to Position 1 (RP-1): provide checklist for students to post their position		
	Argumen t	▪Rules-related to Arguments 1 (RA-1): provide an example for students on how to argument.		
	Feedback	▪Rules-related to Feedback 1 (RF-1): provide rubrics to students to comment on their arguments.		

Based on the conceptual framework of open learner model, an example is shown below.

Assignment: “Please edit codes in C programming: show 20 Integers by Random Function between 0 to 100, and save them in an array. Print the Integer which is over 50 and sum up them. ”

Issues: Whether the Loop statement can be used in this assignment? If it can, then which kind of Loop statement is more suitable for this assignment?

The user-interface of open learner model design is showed in Figure 3.

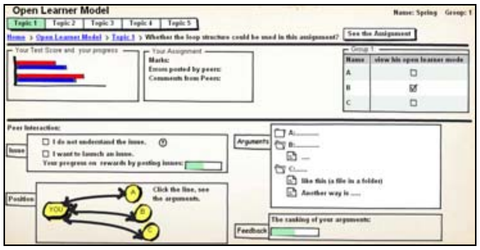

Open Learner Model
Name: Spring Group: 1
Topic 1
Topic 2
Topic 3
Topic 4
Topic 5
Home > Open Learner Model > Topic 1 > Whether the loop structure could be used in this assignment?
See the Assignment
Your Test Score and your progress
Your Assignment
Marks:
Errors posted by peers:
Comments from Peers:
Group 1
Name
view his open learner mode
A
B
C
Peer Interaction:
Issue
☐ I do not understand the issue.
☐ I want to launch an issue.
Your progress on rewards by posting issues:
Arguments
A:.........
B:.........
C:.....
like this (a file in a folder)
Another way is .....
Position
YOU
A
B
C
Click the line, see the arguments.
Feedback
The ranking of your arguments:

Figure 3. The User-Interface of Open Learner Model

5 Evaluation

The evaluation of open learner model includes the usability evaluation and the pedagogy evaluation. In our study the learner model is constructed from three aspects---contents, pedagogy and technology. Among the three aspects, the learning content is constant. Thus, the usability evaluation is to investigate whether the technology could support students to communicate with the contents through user interface. If it could, then the technology is designed appropriately.

After that, the study will investigate the pedagogy evaluation. The pedagogy evaluation means whether the Issue-Position-Argument-Feedback mechanism could support students to peer interaction. If it does not work well, then changes and modification maybe need. The details are listed in Table 2 and Table3.

Table 2. The Data Collecting Methods

Participants / Methods	Usability Evaluation	Pedagogy Evaluation
	Students, Teachers,	Students
Survey A (for teachers and students)	√	
Survey B (for students)		√
Final Testing Scores		√
The Quality of Students Programming		√
The Quality of Comments Each Student Post		√
The Quality of Arguments		√
Levels of Involvement		√

Table 3. The Pedagogy Evaluation

Independent variable / Dependent variable	With Open learner model Class A (IV1)	Without Open learner model Class B(IV2)
Survey B (for students) (V1)	√	
Final Testing Scores (V2)	√	√
The Quality of Students Programming (V3)	√	√
The Quality of Comments Each Student Post (V4)	√	√
The Quality of Arguments(V5)	√	
Levels of Involvement(V6)	√	

The pedagogy evaluation aims to investigate the effectiveness of the open learner model designed in the context of online peer assessment. There are two independent variables, IV1 and IV2 (See Table 3).

The effectiveness will be assessed by six dependent variables---the results of survey B (V1), the students’ final testing scores (V2), the quality of students’ programming (V3) and the quality of comments each student post (V4), the quality of arguments (V5) and levels of involvement (V6). The last three variables will be analyzed from the individual level and group level. The aim is to have an investigation on the effectiveness of individual learning and group learning when using open learner model.

(1) Survey of students

Each question in the questionnaires is related to evaluate the design in the Table 1.

(2) Final testing scores

Each semester, when students finish the programming course, there will be a test in ECUST, which adopts the item banks by computer automatically. Each year, the students will be tested by the same item banks. It is a standardized test. Thus, there will be a compare among the students not only in this semester but also in the past year.

(3) The quality of students’ programming

The students’ assignments will be marked by the lecture. All the participants are taught by one lecture. Thus, the lecture can compare the quality of students’ programming. This compare can be from two aspects. On the one hand, it is the quality of each student’s programming, including the marks given by lecture and the errors students commit. On the other hand, the lecture can compare the whole group level through the average marks and the average errors students commit.

(4) The quality of comments each student post

It will be measured from two aspects. One is to count the number of comments each student post. The other is to inspect the contents in the comments. The aim is to explore whether students make a serious posting, since in the above design in Table 1, the social support is designed in order to encourage students engage in a serious posting activities.

(5) The quality of arguments

The quality of arguments will be measure by the teachers based on the rubrics as mentioned above (Figure2).

(6) Levels of involvement

The level of involvement will be measure by the following questions by interview.

▪ Are students interested in the activity?

▪ Are students attending to and reacting to what peers are doing?

Levels of Involvement may vary depending on the student’s different learning performance.

6 Results and recommendations

In the usability evaluation, including 8 teachers and 54 students took part in filling the questionnaire online. Based on their comments and suggestions from questionnaire, the system was polished, revised and improved.

About 400 students have engaged in this study until this semester. Compared with the common classes, students in the experiment class shows high satisfaction with the learning through the course feedback of questionnaires, high examine pass rate through the final testing (Table 4).

Table 4. A Part of Data Collecting

	The experiment class with open learner model: Class A (IV1)		The common class without open learner model: Class B(IV2)	
	Sep. 2011	Sep. 2012	Sep. 2011	Sep. 2012
The average score of final testing	64.1	72.7	61.5	65.2
The numbers of students taking part in the competition in information literacy	15 students	27 students	Null	Null
The score of students' evaluation about the course	93.33	93.16	<90	<90

The peer interaction in online peer assessment in learning programming course makes students show more motivation taking part in the learning activities. In the experiment class, in total, the number of students is 15 and 27, who took part in the competition activities in information literacy organized by the Ministry of Education in China. However, the number of students is null in the common class. And it also found that students with high learning performance would dominate the process of peer interaction.

This study proposes a method of online peer assessment, which emphasizes the peer interaction. The following are key ideas in our design.

(1)An experiment study of online peer assessment used in programming course

In reviewing literature, the method of online peer assessment used in the big classroom in high education is few in Shanghai. Although all over the world, there are many research reported the peer assessment activities in learning, especially in leaning programming course. Compared with that, few similarity research reports in the past ten years are involved in high education in Shanghai.

(2)Reforming the assessment method of learning in programming course

This method emphasizes "Assessment for learning" rather than "Assessment of learning". It affects the students' attitudes and behaviors. The student attitudes decide the direction of learning, and further stimulate and encourage them engaging in the innovation competition activities.

Generally, the number of students in a classroom is more one hundred in Chinese high education environment, such as in information literacy course, C programming language course etc. The main characteristic of teaching in such classrooms is the number of students. So we need a method to improve students' learning in such classrooms. This study argues that improving students' peer interaction can make students learn more interestingly. For instance, the number of students, in the experiment classroom, who takes part in the competition activities of information literacy, is more than the number of students in the common class. The students show high motivation in taking part in learning activities. (See Table 4)

This study explores the open student model used in online peer assessment, to improve the peer interaction among students. In the pilot study, the method not only makes the student interest in learning programming, and further improves their learning performance. And at the same time, this inspires the students' motivation to take part in the innovation learning activities.

ACKNOWLEDGMENT

This study is supported by the Undergraduate Students Research Fund of East China University of Science and Technology (ECUST).

7 References

[1] Hu, Q. and N. Law. *The design of scaffolding in online peer assessment system for learning programming*. in *16th International Conference on Computers in Education, ICCE 2008*. Taipei, Taiwan: Center for Science and Technology for Learning.

[2] Hu, Q. and N. Law, *Designing Online Peer Assessment System in Learning Programming: an Adaptive Scaffolding Framework and Architecture*, in *World Conference on Educational Multimedia, Hypermedia and Telecommunications 2008*, J. Luca and E.R. Weippl, Editors. 2008, AACE: Vienna, Austria. p. 1739-1744.

[3] Lin, S.S.J., E.Z.F. Liu, and S.M. Yuan, *Web-based peer assessment: Feedback for students with various thinking-styles*. Journal of Computer Assisted Learning, 2001. **17**(4): p. 420-432.

[4] Vygotsky, L.S., *Mind in Society: The Development of Higher Psychological Processes*. 1978: Harvard University Press.

[5] Topping, K., *Peer assessment between students in colleges and universities*. Review of Educational Research, 1998. **68**(3): p. 249.

[6] Chrysafiadi, K. and M. Virvou, *Student modeling approaches: A literature review for the last decade*. Expert Systems with Applications, 2013. **40**(11): p. 4715-4729.

[7] Jeremić, Z., J. Jovanović, and D. Gašević, *Student modeling and assessment in intelligent tutoring of software patterns*. Expert Systems with Applications, 2012. **39**(1): p. 210-222.

[8] Bull, S., *Supporting Learning with Open Learner Models* in *4th Hellenic Conference with International Participation: Information and Communication Technologies in Education, Athens, 2004. (Keynote).*

[9] Zapata-Rivera, D. and J.E. Greer. *Exploring Various Guidance Mechanisms to Support Interaction with Inspectable Learner Models*. in *ITS 2002, LNCS 2363, pp. 442–452,Springer-Verlag Berlin Heidelberg*. 2002.

[10] Kunz, W. and H.W.J. Rittel, *Issue as elements of information systems.* Working Paper No. 131, 1970.

[11] Karacapilidis, N., et al., *Developing Higher-Order Skills with the MEDIT Web-based Learning Environment.* Educational Technology & Society, 2000. **3**(1).

[12] Liu, C.-C. and C.-C. Tsai, *An analysis of peer interaction patterns as discoursed by on-line small group problem-solving activity.* Computers & Education, 2008. **50**(3): p. 627-639.

[13] Webb, N.M., *Testing a Theoretical Model of Student Interaction and Learning in Small Groups* in *Interaction in cooperative groups : the theoretical anatomy of group learning*, R. Hertz-Lazarowitz and N. Miller, Editors. 1992, Cambridge University Press: Cambridge : p. 102-119.

[14] Millán, E., T. Loboda, and J.L. Pérez-de-la-Cruz, *Bayesian networks for student model engineering.* Computers & Education, 2010. **55**(4): p. 1663-1683.

[15] Chrysafiadi, K. and M. Virvou, *Evaluating the integration of fuzzy logic into the student model of a web-based learning environment.* Expert Systems with Applications, 2012. **39**(18): p. 13127-13134.

[16] Bull, S. and J. Kay, *Student Models that Invite the Learner In: The SMILI: Open Learner Modelling Framework.* International Journal of Artificial Intelligence in Education, 2007 **17**(2): p. 89-120.

GAMES AS A PROPOSAL FOR CHANGE IN THE CIVIL ENGINEERING LEARNING PROCESS

Authors: Rosas Sánchez, M. E., MSc,[1] and Casanova del Angel, F. Dr.[2]

[1]: Instituto Politécnico Nacional, ESIA UP ALM. Mexico. mrosass@ipn.mx

[2]: Instituto Politécnico Nacional, SEPI ESIA UP ALM. Mexico. fcasanova@ipn.mx

Abstract

This work makes an academic contribution through revision of the educational activities in the higher level of engineering schools, in order to increase performance of Civil Engineering students, particularly in the Structures subject. For this purpose, learning elements were modified, focusing on the development of skills and confidence of every student. In like manner, learning strategies different from the traditional ones were fostered, resulting in acrostics, short stories, comics, games and songs created by the students, based, mainly, on the following topics: Statics, Isostatic Structures and Resistance of Materials. With this, the learning environment of students was improved and, at the same tame, learning was supported through teamwork and deeper knowledge on the subject.

Key words: learning, teaching, skills, suitability and competence

Introduction

Among the most complex and frequent problems teachers face throughout the teaching-learning process in the civil engineering area, is to achieve that students understand the behavior of structures, since it is difficult for them to understand the corresponding fundamentals, since the relationship between mathematics and structural behavior is not immediately evident. Therefore, how could those concepts be related in order to generate significant learning? And, which is the best way to relate theory and practice in order to achieve such learning? In order to answer these starting questions, an objective of this work is to contribute to achieve learning through an innovative and different proposal, with the development of board games and other contributions for teaching. During the 2009-2011 period, in the Statics subject, the index of failed students ranged between 50% and 60%, which lead to the preparation of proposals based on the features of student population regarding factors influencing their school career. Therefore, to identify the mechanisms contributing to avoid scholar failure generating withdrawal becomes relevant.

As a consequence, to increase the students' performance and quality of the educational process became urgent in order to decrease failure and achieve performance indexes complying with scholar efficiency. In order to achieve these objectives, it is necessary to consolidate an educational offer of the best quality, and to improve academic support to students in a quantitative and a qualitative manner. For such purpose, aspects such as educational level and professionalism of teachers; the way academic work is organized; relevance and updated syllabi, material aids; and, mainly, scholar features of students must be combined.

A theory to better understand the concept of significant learning is that focusing on the teaching-learning processes, previous scientific concepts of the student in daily life, and considering every learning situation may be studied along two axes:

- learning by the student (whether by memory or significant), and
- teaching strategy (whether guided or autonomous reception/discovery).

Generation of meanings as an internalization process implies a theoretical intermediate position between associationist idea in which meanings become valuable based on the exterior (in accordance with a correspondence principle) and Piaget's theory, in accordance with which the subject significantly builds in an autonomous way. Vygotsky's development incorporates, in a clear and explicit manner, the influence of social environment **(***Baquero, 1993***)**.

For the last years, a group of teachers has been implementing new educational practices which have led us to achieve significant learning in the students in the Civil Engineering area **(***IPN, 2002***)**. This work focuses on the Structures subject, where students show higher learning difficulties, whether due to lack of previous mathematics knowledge or because the teacher is not skilled enough to teach basic fundamentals of structures.

In the Statics subject, in general, topics are presented in a traditional way, through utilization of a whiteboard and color markers. Final assessment is carried out through a written examination, focused on *Balance of Forces,* important concept since Statics involves balance. Under such scheme, at the end of the semester there is a large number of failed students, those being the ones who could not understand the fundamentals. Particularly during the 2010 January-July period, in the Statics subject, the proposal to carry out a work throughout the semester, consisting in developing the subject in a non-traditional way, was presented to the students. The students presented, in an individual manner, short stories, comics, poems, acrostics and a song on specific topics of the subject **(***Rosas, 2010***)**. In the 2010 August-December semester, a different way to work out of the classroom was presented, introducing another assessment mechanism: the development of a work as a complement for partial examinations. It included making board games such as “Goose", "One", "Bingo", "Memorama" and "snakes and ladders", in which they would use and develop the statics topics and concepts taught in class. The students decided which kind of game they would carry out in small teams of six students. With such game, they were able to interact and have the support of the teacher in order to fully understand the topics.

While they developed their application work, difficulties for progress in the assignment arose, such as distraction and lack of knowledge. In order to overcome obstacles to create games, a different way to address difficulties in the structures area was developed. That gave a sense of peculiarity, significance and distinctiveness to the group and, as they progressed, the members showed more capability to work with peers in order to understand fundamental topics of the subject. Doubts regarding balance of bodies were cleared during the development of games. It was a different way to learn the conceptual part of this kind of structures topic.

The following semester, trying to innovate, it was decided to use Information and Communication Technologies (ICT), since such make learning and the development of skills easier, and increase efficiency in the classroom based on their usage and easiness to use the Internet and informatics. Based on the use of ICT, we modified learning elements, focusing on the development of the students' skills and confidence. Based on the above, in order to better support this proposal, to use new didactical strategies became necessary. Teachers interested in educational innovation were considered to have better possibilities of success focusing on improvement of their competencies in order to achieve significant learning.

We believe that teaching begins with the implementation of certain characteristics inherent to the education. Nowadays, education means: learn to learn, learn to know, learn to do, and learn to understand others. Therefore, we found in these strategies a helpful tool, since:

- They are innovative and creative, providing access to new ways of knowledge;
- They have more influence and benefits, in a higher proportion, for the educational area, rendering it more accessible and dynamic; and
- They are communication means potentializing learning.

The educational work throughout these semesters allowed to foster learning strategies different from the traditional ones, which gave as a result that students adapted traditional board games with topics based on the subjects of Statics, Isostatic Structures and Resistance of Materials, achieving the improvement of educational environment of students and, at the same time, supporting learning through teamwork while the student acquired deeper knowledge on the subject.

Application

Let us see below a typical and traditional example of the Balance Unit taught in the Statics subject. The resulting force of the three tensions of cables shown in the Figure is vertical and $T_1 = T_2 = T$, specify T_3

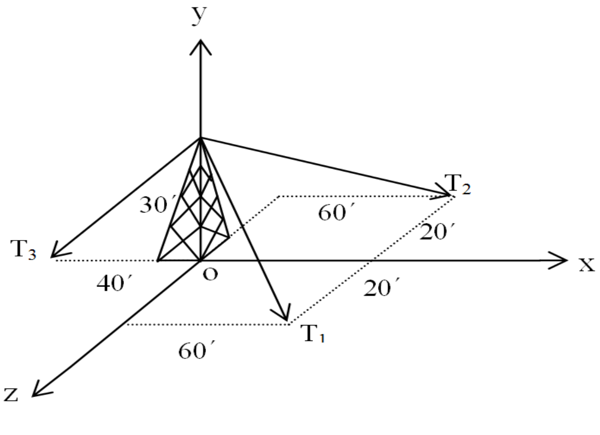


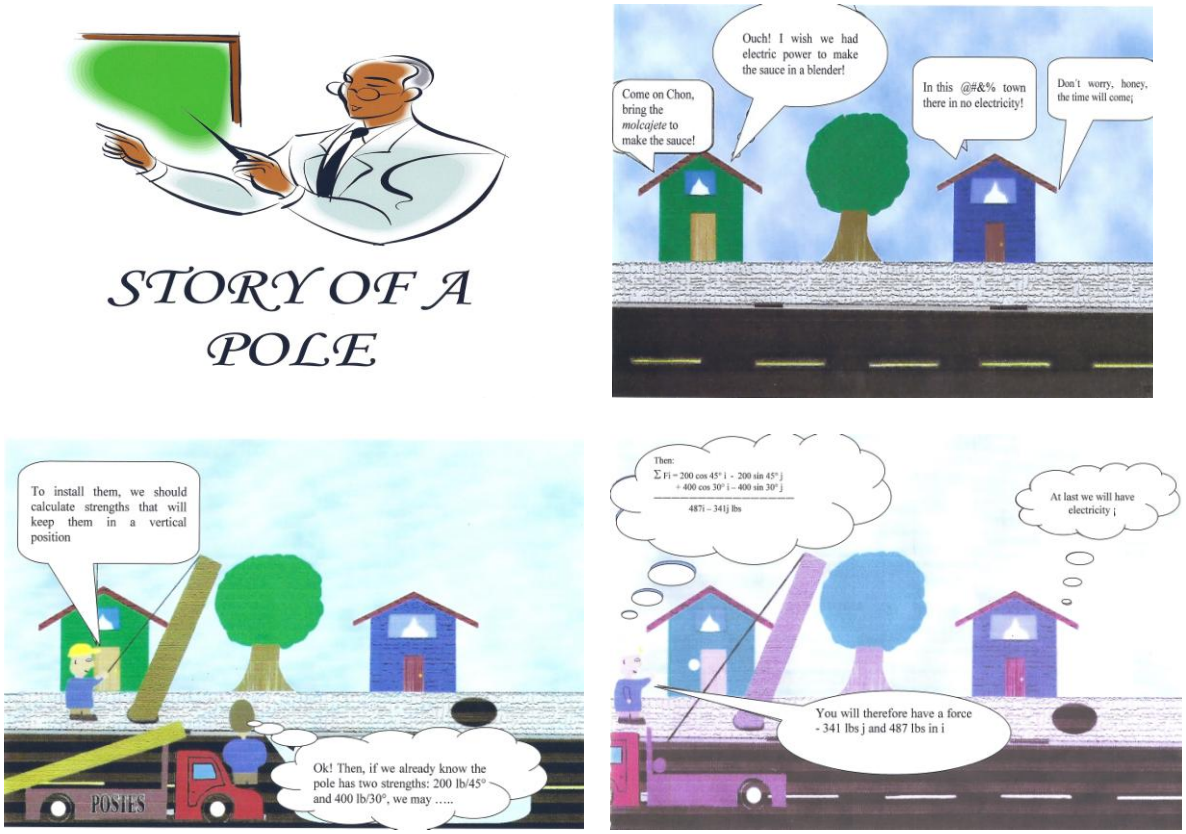
Figure 1. Tower in vertical position fixed with three cables.

Required vectorial equations in order to solve such system of concurring forces in space are: $\Sigma F_x = 0$; $\Sigma F_y = 0$; $\Sigma F_z = 0$. Using these equations, we will solve the system of equations, obtaining: $F_R = \sum F\boldsymbol{i} = T_1 + T_2 + T_3$. Since we know that: $T_1 = T_2 = T$. We shall determine T_3 using the following equations:

$$F_R = \sum F_{\boldsymbol{i}}$$
$$F_R = T\,(60i\text{-}30j\text{+}20k)/70 + T(60i\text{-}30j\text{-}20k)/70 + T_3(\text{-}40i\text{-}30j)/50$$
$$F_R = T(0.86i - 0.43j + 0.28k) + T(0.86i - 0.43j + 0.28k) + T_3(-0.80i - 0.60j)$$
$$F_R = 0.86Ti + 0.86Ti - 0.80T_3i - 0.43Tj - 0.43Tj - 0.60T_3j + 0.28Tk + 0.28Tk$$
$$F_R = 1.72Ti - 0.86Tj - 0.80T_3i - 0.60T_3j$$
$$F_R = 1.72Ti - 0.80T_3i - 0.86Tj - 0.60T_3j$$

Finding the value of unknown T value from the *x* axis: $1.72\ Ti = 0.80\ T_3i$. Therefore, $T_3 = 1.72\ Ti/0.80i = 2.15\ T$. Substituting such value in the *y* axis, we have a strength of: $F_R = -2.3\ T$

As we see, the procedure is somewhat difficult for the student with scarce knowledge of algebra. One of the work proposals for third, fourth and fifth semesters of the Bachelor Degree in Civil Engineering was to assess and/or make various materials based on structures topics, depending on their skills. At the end of the courses, various materials were shown, among them, the one shown in Figure 2, where the student represented the topic of balance in the following comic:


STORY OF A POLE
Come on Chon, bring the *molcajete* to make the sauce!
Ouch! I wish we had electric power to make the sauce in a blender!
In this @#&% town there in no electricity!
Don´t worry, honey, the time will come¡
To install them, we should calculate strengths that will keep them in a vertical position
Ok! Then, if we already know the pole has two strengths: 200 lb/45° and 400 lb/30°, we may …..
POSTES
Then:
Σ Fi = 200 cos 45° i - 200 sin 45° j
+ 400 cos 30° i – 400 sin 30° j
487i – 341j lbs
At last we will have electricity ¡
You will therefore have a force - 341 lbs j and 487 lbs in i

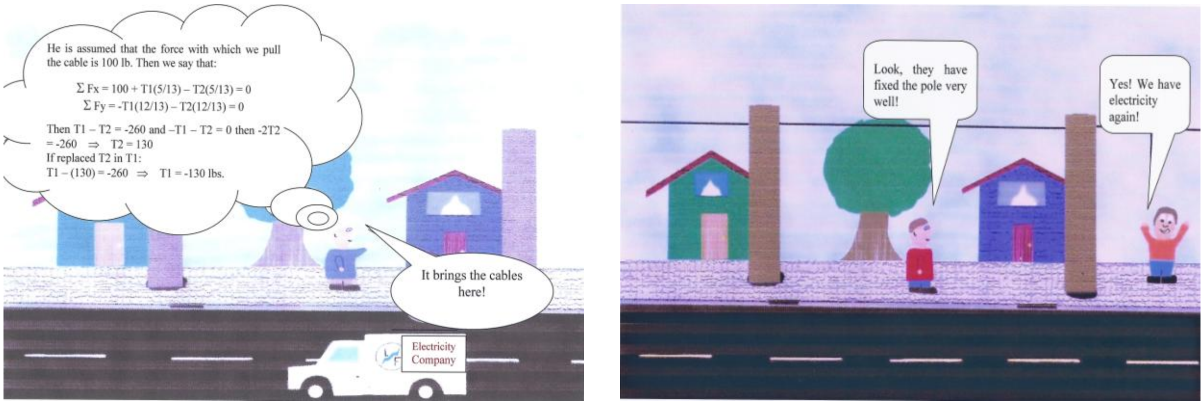

He is assumed that the force with which we pull the cable is 100 lb. Then we say that:
$\Sigma Fx = 100 + T1(5/13) – T2(5/13) = 0$
$\Sigma Fy = -T1(12/13) – T2(12/13) = 0$
Then $T1 – T2 = -260$ and $–T1 – T2 = 0$ then $-2T2 = -260 \Rightarrow T2 = 130$
If replaced T2 in T1:
$T1 – (130) = -260 \Rightarrow T1 = -130$ lbs.
It brings the cables here!
Electricity Company
Look, they have fixed the pole very well!
Yes! We have electricity again!

Figure 2. Representation of balance in a comic.

Conclusions

Among results obtained throughout the development of the educational research herein shown, the following must be underlined:

- Most participants considered that the innovation process was motivating, thus demonstrating that impact of innovations in the teaching-learning process are relevant and support educational processes.
- This new activity enriched learning by students, since teamwork and new skills for understanding of fundamentals in the subject were achieved.
- With the new educational models implemented in the Mexican education system; particularly at the Instituto Politécnico Nacional, the preparation of new strategies contributing to the achievement of institutional goals for higher education becomes necessary.
- The teacher is promoter and enabler of learning, since he/she develops new methods, techniques and activities in order to promote building knowledge among students, as well as linking teaching contents with their personal and social experiences; and
- Since the teacher is who has renewed his/her practice and has suffered the transformation process, he/she can account for such and pass it to other teachers presenting educational and training proposals focused on transformation of educational practice.

Higher education institutions should consider this type of innovative projects. They should focus more on innovative aspects, widen training of teachers with plans and syllabi allowing them to provide the students with required knowledge and competencies, to promote critical and independent thought, as well as the capacity to learn throughout life. Such criteria shall stimulate innovation and significant learning.

Acknowledgments

This document was developed with part of the time devoted to the IPN-SIP 20120585 research project.

References

Baquero, Ricardo. (1993). *Aprendizaje pedagógico. Módulo de trabajo PTFD. Ministerio de Educación.* Argentina.

Fuentes Agustí, Marta. (2001). Naufragar en Internet. Ed. Virtual Educa. México

Gimeno, S. J. (1999). *Diseño de la enseñanza. El papel de los profesores*. Curso de inducción a la docencia en el IPN. México.

Instituto Politécnico Nacional. (2002). *Un nuevo modelo educativo para el IPN*. México.

Programa de Desarrollo Institucional. (2001). IPN. México

Rosas Sánchez, M. E. (2010). Contribución académica a la licenciatura en ingeniería civil. XVIII Congreso Nacional de Ingeniería Estructural. León, Guanajuato. México.

Villa, A and Poblete, M. (2007). *Aprendizaje basado en competencias, una propuesta para la evaluación de las competencias genéricas.* Bilbao. Ediciones Mensajero/ICE Universidad de Deusto.

Accessibility evaluation of Chats and Forums in e-learning environments

R. Calvo[1], B. Iglesias[1], A. Gil[1] and A.Iglesias[1]
[1]Computer Science, Universidad Carlos III de Madrid, Leganés, Madrid, Spain

Abstract *- Collaborative learning is useful for students in their learning process. Nowadays, most e-learning systems include Computer Supported Collaborative Learning (CSCL) tools like chats and forums; however, are they accessible for everybody? This paper presents a heuristic evaluation of accessibility of two CSCL tools (chat and forum) in four web-based, open-source Learning Content Management Systems (LCMS): Moodle, ATutor, dotLRN and Claroline. The evaluation results show that the CSCL tools evaluated present accessibility barriers which are a handicap for many students who want to use the LCMSs Moreover, some recommendations are offered in order to improve the accessibility of the evaluated tools. Considering these recommendations in the development of the evaluated tools, all students could participate actively in the collaborative tasks proposed by teachers.*

Keywords: Accessibility; Computer Supported Collaborative Learning Tools; Chats; Forums; Learning Content Management Systems.

1 Introduction

Nowadays, many educational centers combine their traditional learning based on face-to-face classes and electronic learning (e-learning) systems based on web sites [1]. These e-learning systems are created with Learning Content Management Systems (LCMS) or Virtual Learning Environments (VLE) [2]. It is software which facilitates the creation and management of courses. Among all their functionalities, LCMSs offer typically tools to share materials, to assess or to collaborate. With regard to the collaborative tools, which are usually named Computer Supported Collaborative Learning (CSCL) tools, there are many tools which help users in the collaborative process. Some of the most important CSCL tools are chats and forums, which are really useful for students. However, some students cannot access to these CSCL environments because they have accessibility barriers. As a result, these LCMSs do not accomplish with some educational laws.

The main goal of this research work is to answer the question: are the chat and forum of the most used LCMSs (Moodle, ATutor, dotLRN and Claroline) accessible for everybody? To achieve it, a heuristic accessibility evaluation is carried out from the point of view of World Wide Web Consortium (W3C) guidelines. Then, considering the obtained results, some recommendations are specified to improve the accessibility of the evaluated tools.

FECS'12, July 22–25, 2013, Las Vegas, Nevada, USA

2 State of art

2.1 Accessibility: laws and guidelines

Everybody has the right to access to the Information Technologies (IT) in spite of their disabilities, age, technical environment or circumstances. Thus, there are different accessibility standards, guidelines and laws which try to normalize or regulate access to ITs and learning environments.

From the point of view of IT laws, United States of America (USA) has created a law to protect the rights of people with disabilities, Section 508 [3] and Europe developed the Recommendation 2006/952/EC on the Protection of Minors and Human Dignity in Audiovisual and Information Services [4].

On the other hand, there are laws and recommendations which enshrine the rights of every student, nevertheless of their abilities and functional diversity. For instance, the United Nations Educational, Scientific and Cultural Organization (UNESCO) conventions, recommendations and declarations [5] and the Convention on the Rights of People with Disabilities [6], in particular article 24 on education, consider that discrimination in education is a violation of the human rights. Moreover, United Kingdom has specified the law Disability Discrimination Act (DDA) [7]for education and USA has created the law Individuals with Disabilities Education Act (IDEA)[8]

With regard to the accessibility standards and guidelines that LCMSs must accomplish, W3C provides some accessibility guidelines like Authoring Tool Accessibility Guidelines (ATAG)[9] and Web Content Accessibility Guidelines (WCAG)[10], which has been converted in a ISO standard the ISO/IEC 40500:2012 Information technology -- W3C Web Content Accessibility Guidelines (WCAG) 2.0[11]

2.2 Computer supported collaborative learning tools and accessibility barriers

CSCL tools provide teachers and students benefits such as: share information and knowledge; facilitate the communication between them or allow them to participate in their learning process in an effective way [12]. Chat and Forum are two of the most useful CSCL tools in e-learning systems [13]. These tools allow students to exchange information and to communicate with other students or teachers easily.

Currently, an active participation in these CSCL tools is really useful for students and this participation is even taken into account for the final marks by some teachers [14]. However, some students cannot access to these tools because they present accessibility barriers. Previous studies have detected that the basic functions of LCMSs usually present accessibility barriers [15].

Specifically in CSCL tools, preliminary evaluations detected the main accessibility problems that forums and chats have [16]. These problems are explained in detail in this research work.

With regard to Chats, some specific accessibility barriers have been detected in previous research works. For instance, screen reader users have accessibility problems if the website is auto-refreshing continuously and it is not tagged properly because it causes the screen reader restarts [17]. Moreover, if the user is not informed about new opened Windows or new created buttons the user can be disoriented [18]. Besides, if one of the emitters is not able to write quickly, he could not be able to follow the conversation [19].

On the other hand, Forums present also specific accessibility barriers because these tools used to have a What You See Is What You Get (WYSIWYG) feature which is not accessible for some users [20]. Furthermore, this tool allows users to create content and, if the user is not an accessibility expert, s/he could generate inaccessible content. For instance, s/he could use tables for layout or colors without a minimum contrast [21].

However, none of these previous evaluations carried out heuristic accessibility evaluations of these tools from the point of view of W3C guidelines. Thus, the main goal of this research is to detect the main accessibility barriers that people have to face when use these tools in four of the most used CSCL environments.

3 Evaluation

The details of the heuristic evaluation of accessibility carried out in this study are described in next subsections.

3.1 Evaluation objective and environment

The main objective of the paper is to evaluate the accessibility of a synchronous and an asynchronous tool because they have different ways of interaction. Previous studies have demonstrated the usefulness of the chat and forum as synchronous and asynchronous CSCL tools respectively. Thus, this research is focused on the evaluation of the chat and forum tools in the selected e-learning systems from the point of view of accessibility. Moreover, as the study reveals accessibility barriers, some recommendations are specified to correct them.

With regard to the LCMSs selected, four of the most used web-based, open-source LCMS in the world have been chosen for the comparative study: Atutor 2.0.3[1], Claroline 1.10.6 [2], dotLrn 2.4[3] and Moodle 2.0.5[4]. Furthermore, all of them take into account accessibility in their development.

The accessibility evaluation includes two different perspectives: one evaluates the accessibility of the interface in all CSCL tools; and the other evaluates the accessibility of the CSCL tool as an authoring tool.

Thus, to achieve it the WCAG 2.0 (Web Content Accessibility Guidelines)[10] and the ATAG 2.0 (Authoring Tool Accessibility Guidelines)[9] are considered. These guidelines are divided into different priority levels, from A to AAA. For this evaluation, the priority level selected is the AA conformance level because it is the minimum accessibility level required by the law; thus, the tool should accomplish with the A and AA guidelines.

3.2 Method for evaluation

A heuristic evaluation has been carried out by three different accessibility experts. According to the methodology and recommendations of W3C [22], the interface evaluation of the CSCL tools was conducted automatically, semi-automatically and manually from the perspective of WCAG 2.0. In this case, the automatic tools used were TAW[5] and Hera[6] and the semi-automatic tool was WAVE[7].

On the other hand, the accessibility evaluation of the CSCL tools from the point of view of an authoring tool was carried out manually and with the help of semi-automatic tools according to the ATAG 2.0 draft guidelines. Considering that, to the knowledge of the authors of this article, no automatic tools exist currently for this kind of evaluations but the semi-automatic tool WAVE was used.

3.3 Evaluation results

This section presents the main results obtained in the heuristic evaluation. These results indicate that accessibility barriers are presented in the CSCL tools of each LCMS evaluated. Actually, the analysis of the findings shows that none of the collaborative tools accomplish even the A priority level of WCAG or ATAG guidelines.

The obtained results are summarized in Table 1. The errors have been categorized in general errors (E1 to E14) depending on the nature of the error. For instance, the category E1 groups the WCAG 2.0 and ATAG 2.0 guidelines which are related with non-textual content. Thus, it considers that non-textual content in the system should need alternative information in order to be accessible. Table 1 shows the Code of error (name of the category) and the WCAG 2.0 and ATAG 2.0 guidelines which are related to this category. For instance, guidelines 1.1.1, 1.2.1, 1.2.2, 1.2.3, 1.2.5 and 4.1.2 of WCAG 2.0 and guidelines A.1.1.1, A.1.2.1, A.1.2.2, A.2.1.1 and A.2.1.2 of ATAG 2.0.

On the other hand, Table 2 and Table 3, which are showed in the Annex 1, detail the errors found in the CSCL tools and how they have been categorized in this paper (code and description).

According to the results, the most accessible chat tools are the chats of Moodle and ATutor, because they accomplish more guidelines of A priority level than the others tools. Moreover, they try to solve one of the specific accessibility problems of chats, the auto-refresh.

On the other hand, the most accessible forum is the forum in DotLRN because it is the tool which fulfills more accessibility guidelines of A priority level. However, all of the tools present accessibility problems and none of them help the author to create accessible content.

A complete list of accessibility barriers presented in each collaborative tool for each LCMS can be found at the website http://labda.inf.uc3m.es/Evaluations [8].Next, the main accessibility barriers found are explained for each CSCL tool: chat and forum.

[1] Atutor. http://atutor.ca/atutor/ (May 2013)

[2] Claroline. http://www.claroline.net/?lang=en (May 2013)

[3] dotLrn. http://www.dotlrn.org/index.html (May 2013)

[4] Moodle. https://moodle.org/ (May 2013)

[5] TAW. http://www.tawdis.net/ (May 2013)

[6] Hera http://www.sidar.org/hera/ (May 2013)

[7] Wave. http://wave.webaim.org/ (May 2013)

[8] Password: EVALUATIONS

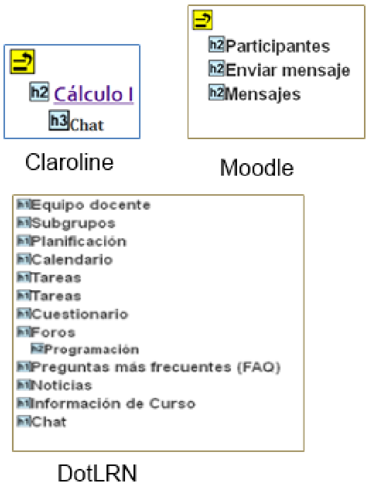
Table 1. Relationship between errors categories and WCAG and ATAG guidelines

Code	WCAG 2.0	ATAG 2.0
E1_ NonTextualContent	1.1.1;1.2.1;1.2.2; 1.2.3;1.2.5; 4.1.2	A.1.1.1;A.1.2.1; A.1.2.2;A.2.1.1; A.2.1.2
E2_ Information	1.3.1;1.3.2 ;1.4.3 2.4.2;2.4.4;2.4.5 2.4.6;2.4.7;3.1.1 3.1.2;3.2.2;3.2.3 3.3.2	A.1.1.1;A.1.2.1 A.1.2.2
E3_ Sensorial	1.3.3;1.4.1	A.1.1.1;A.1.2.1 A.1.2.2
E4_ Focus	2.4.3;2.4.7	A.1.1.1;A.1.2.1 A.1.2.2
E5_ Personalization	1.4.2;2.2.1;2.2.2	A.1.1.1;A.1.2.1 A.1.2.2;A.3.2.2 A.3.3.1;A.3.6.2
E6_ Keyboard	2.1.1;2.1.2;2.4.1 2.4.3; 3.2.2	A.1.1.1;A.1.2.1 A.1.2.2;A.3.1.1 A.3.1.2;A.3.1.3
E7_ ControlErrors	3.3.1;3.3.3	A.1.1.1;A.1.2.1 A.1.2.2
E8_ StandardErrors	4.1.1	A.1.1.1;A.1.2.1 A.1.2.2
E9_ EditingViews	--	A.1.1.1;A.1.2.1 A.1.2.2,A.2.2.1 A.2.2.2,A.3.4.1 A.3.5.1, A.3.7.1
E10_ ImproveSession	--	A.3.2.1, A.3.2.2
E11_ Documentation	--	A.4.2.1, A.4.2.2; B.2.4.1
E12_Genaration Accessible Content	--	B.1.1.1;B.1.1.2 B.1.2.2,B.1.2.1 B.1.2.4
E13_ Produce Accessible Content	--	B.2.1.1;B.2.2.1 B.2.2.2;B.2.3.2 B.2.3.3;B.2.4.1 B.2.4.2;B.3.1.1 B.3.1.2
E14_ AccessibilityFeatures	--	B.4.1.1;B.4.1.3 B.4.1.4;B.4.1.5 B.4.2.1;B.4.2.2

3.3.1 Chat

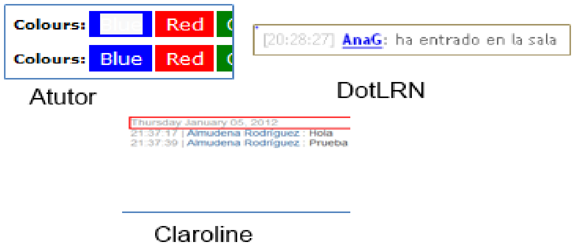
With regard to the WCAG 2.0 guidelines, some errors can be highlighted in chats. All the non-textual content must contain alternative information; however, none of the tools provide a description for all images (E1). Moreover, the information is not well-structured or visualized (E2). For instance, the headers and the website title are not used properly in each chat tools evaluated. The Figure 1 DotLRN uses headers in an improper way because it specifies the left menu and not the page structure. Moreover, Moodle and Claroline do not follow a proper logic order and Atutor does not use headings in the chat but it is not showed because the result is null.

Figure 1. Headings



Besides, the contrast ratio is not the minimum in the chats except in Moodle (E3) as it is shown in the Figure 2 . Thus, the user is not able to distinguish the information showed in the chat.

Figure 2. Contrast Ratio in Chats

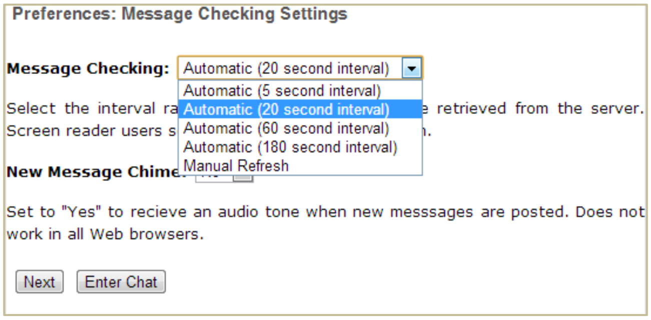


[20:28:27] AnaG: ha entrado en la sala

Thursday January 05, 2012
21:37:17 | Almudena Rodriguez : Hola
21:37:30 | Almudena Rodriguez : Prueba

The focus is not visible in all the evaluated chats except of Moodle and the focus order is correct only in Claroline (E4). Related to the error category E5, all chats except the chat of Moodle and Atutor do not allow adjusting the timing. Thus, the time of the conversation cannot be stopped, paused or adjusted. For instance, as it is shown in the Figure 3, the Atutor chat allows the user to specify the time interval to refresh the chat and if the user prefers to refresh the chat manually. Moreover, the user can specify the sound of the new messages but he cannot disable the sound of new connected users, for example.

Figure 3. Atutor Chat Preferences



Preferences: Message Checking Settings

Message Checking: Automatic (20 second interval)

Select the interval ra… …e retrieved from the server.
Screen reader users s… …).

New Message Chime…

Set to "Yes" to recieve an audio tone when new messsages are posted. Does not work in all Web browsers.

Automatic (5 second interval)
Automatic (20 second interval)
Automatic (60 second interval)
Automatic (180 second interval)
Manual Refresh
Next
Enter Chat

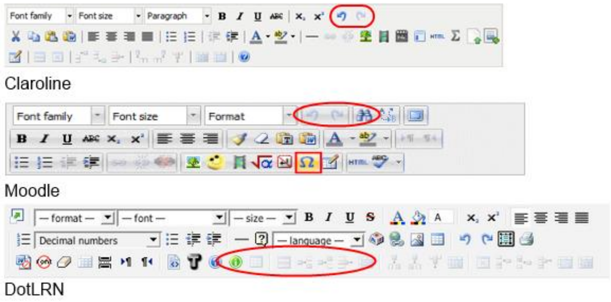
Other important error is that the user is not able to control any chat with the keyboard exclusively (E6). With regard to error E7, Claroline and Atutor do not check all the errors that the user can commit. For instance, the user could send blank messages. Finally, all the tools have webpages which have not been developed using web standards as HTML or CSS (E8).

From the point of view of ATAG guidelines it is important to emphasize that this module produces fewer content than forums. Thus, there are fewer accessibility guidelines which are not fulfilled. The first guideline of the ATAG makes a reference to the accomplishment of the WCAG; so, as it has been explained before, there are many WCAG guidelines (errors from E1 to E8) which are not fulfilled. Moreover, when the authoring tool generates content, this content is not accessible (E12) as it is shown in the figures, Figure 1 and Figure 2; Furthermore, the author cannot check the accessibility of the content created by himself automatically or semi-automatically (E13) because there is not functionalities for it. Finally, the documentation of the e-learning tools does not provide accessibility information to inform the user about the accessibility functionalities and the accessibility problems that users can face when they use chats (E11, E14).

3.3.2 Forum

Regarding to the content, according to WCAG 2.0, a general error that exists in all the forums is that alternative text is not provided for the images and likewise subtitles or audio-descriptions files are not provided for the uploaded videos (E1). Moreover, there are not elements to make the navigation easier within the application and some pages do not have an appropriate title (E2). Also, Claroline, Moodle and DotLRN use colors to represent the elements that are enabled or disabled (E3). The Figure 4 shows the WYSIWYG Editors of Claroline, Moodle and DotLRN which use colors to represent the enabled or disabled buttons. Besides, there are elements which are not accessible and manageable through keyboard and the focus is not visible (E4, E6). For instance, the WYSIWIG editors used in the LCMSs are not accessible by keyboard (see Figure 4).

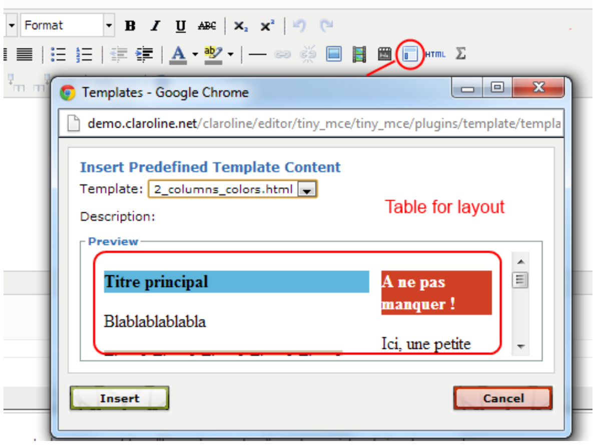
Figure 4. WYSIWYG Editors


Claroline
Moodle
DotLRN

Finally, ATutor does not allow to personalize the audio (E5), Claroline does not control the errors produced by the users (E7) and all the tools evaluated have bugs in the code and style sheet (E8).

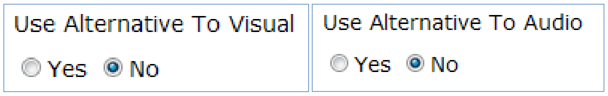
Considering the ATAG 2.0 guidelines, all the tools have accessibility problems related to the WCAG (errors from E1 to E8). Thus, they do not accomplish the first guideline of the ATAG. Claroline and Atutor do not show the status messages and the text presentation in a programmatic way and all tools, except Moodle, do not allow searching through the content (E9). Moreover, ATutor cannot preview the content properly (E9) and Claroline does not auto-save the content (E10). The tools evaluated do not provide a mechanism to verify the accessibility of that content and the generated content is not accessible (E12). Furthermore, the tools do not suggest and serve as a guide to the author, in the same way it must warn about accessibility errors when needed (E13). For instance, Claroline allows users to include templates in the generated content. However, this content is not accessible because it includes a table for layout as it is shown in the Figure 5 and the template does not specify its accessibility level.

Figure 5. Insert Predefined Template Content


Templates - Google Chrome
demo.claroline.net/claroline/editor/tiny_mce/tiny_mce/plugins/template/templa
Insert Predefined Template Content
Template: 2_columns_colors.html
Description:
Preview
Titre principal
A ne pas manquer !
Blablablablabla
Ici, une petite
Table for layout

Moreover, ATutor provides accessibility features; however, these features are not activated by default, see error E14.

Figure 6. Accessibility Features Are Not Activated by Default


Use Alternative To Visual ○ Yes ● No
Use Alternative To Audio ○ Yes ● No

Finally, none of the tools provide clear and complete instructions about the use of the tool, including accessibility examples and documentation related to the accessibility (E11, E14).

4 Recommendations

Considering the evaluation results obtained from this research work, a set of recommendations has been elaborated in order to improve the accessibility of the chat and forum tools for the four LCMS. A summary of them is list next:

- **Provide textual information:** The chat and the forum should provide textual information for the non-textual content showed in the interface. For instance, every image should have alternative content, every video uploaded should include subtitles and audio-description and every text input should have a label associated.
- **Keyboard:** The CSCL tools should be controlled completely through keyboard. For instance, users who cannot use the mouse would not have any handicap to use it.
- **Skip content:** The CSCL tools should include mechanisms to skip content or use shortcuts.
- **Avoid errors:** The tools should help the author to avoid errors like sending blank messages or creating inaccessibility content. It is important to remark that authors could not be accessibility experts and even expert users can commit errors.
- **Web standards:** All the webpages and style sheets should be created without code errors and according to web standards like HTML, CSS, etc.
- **Check accessibility:** The tools should inform the authors about the accessibility errors and how to solve them.

- **Accessibility documentation:** The tools should provide documentation related to accessibility features and how to create accessible content as well as complete documentation about the entire tool.

Specifically for the chat tool, due to its synchronous character, it is really important for the users to be able to stop, control and adjust the time of auto-refreshing the sentences. Thus, users could be able to follow the conversation without any problem.

Finally, the forum tool should check the accessibility of the content generated by the authors. Thus, the tool should control the accessibility automatically and inform the authors when there were accessibility errors in their content and how to solve them. Moreover, the tool should allow searching through the content and if the tool allows previews of the content, the previewed content should be showed properly.

5 Conclusion

The study presented in this paper lay out the accessibility barriers that students and teachers usually face when using chat and forum as CSCL tools in e-learning systems and offers a set of general and specific recommendations to solve these barriers.

This comparative study is based on a heuristic evaluation of four of the most used LCMSs worldwide: Moodle, Atutor, Claroline and dotLRN. The research concludes that every CSCL tool in each LCMS present accessibility barriers. The most accessible chat tools are the ones of Atutor and Moodle, which also incorporate ways to control the auto-refresh of the website. On the other hand, the most accessible forum tool is the DotLRN forum because it accomplishes more guidelines of A priority level,

Currently we are working in evaluating the accessibility of other collaborative tools in e-learning systems and moreover we are preparing a user evaluation of all these tools.

6 Acknowledgments

This research work has been supported by the Regional Government of Madrid under the Research Network MA2VICMR (S2009/TIC-1542) and by the Spanish Ministry of Economy under the project MULTIMEDICA (TIN2010-20644-C03-01).

7 References

[1] Clark, R. and Mayer, R. 2011. *E-Learning and the Science of Instruction*. John Willey and Sons.

[2] Pizzutilo, S., Tangorra, F., and De Carolis, B. 2005. An e-learning environment based on open-source software. In *Proceedings of the 4th WSEAS international Conference on Telecommunications and informatics* (Prague, Czech Republic, March 13-15, 2005) pp. 1-6.

[3] Sección 508 United States Laws, Overview of the Rehabilitation Act of 1973 (Sections 504 and 508). (1998). http://www.section508.gov/index.cfm?&FuseAction=Content&ID=12

[4] Recommendation 2006/952/EC of the European Parliament and of the Council on the protection of minors and human dignity . (2006) http://europa.eu/legislation_summaries/audiovisual_and_media/l24030a_en.htm

[5] UNESCO. Convention against Discrimination in Education (1960) http://portal.unesco.org/en/ev.php-URL_ID=12949&URL_DO=DO_TOPIC&URL_SECTION=201.html

[6] Convention on the Rights of Persons with Disabilities. (2006) http://www.un.org/disabilities/convention/conventionfull.shtml

[7] Disability Discrimination Act (DDA) of 1995. (1995) http://www.direct.gov.uk/en/DisabledPeople/RightsAndObligations/DisabilityRights/DG_4001068

[8] Individuals with Disabilities Education Act (IDEA) of 2004. (2004) http://idea.ed.gov/

[9] W3C. Authoring Tool Accessibility Guidelines. 2012. http://www.w3.org/TR/ATAG20/

[10] W3C. Web Content Accessibility Guidelines. 2008. http://www.w3.org/TR/WCAG/

[11] ISO/IEC 40500:2012 Information technology -- W3C Web Content Accessibility Guidelines (WCAG) 2.0. http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=58625

[12] Soller, A. (2001) Supporting Social Interaction in an Intelligent Collaborative Learning System International *Journal of Artificial Intelligence in Education*, 12, 40–62

[13] Curran, K. (2002) A Web-based collaboration teaching environment. *IEEE Multimedia, 9, 3. (Jul-Sept 2002) 72-76.* DOI= 10.1109/MMUL.2002.1022860

[14] Shih, C. and Gamon, J. (2002) Relationships among learning strategies, patterns, styles and achievement in web-based courses. *Journal of Agricultural Education, 43, 4.* 1-11. DOI= 10.5032/jae.2002.04001

[15] Iglesias, A., Moreno, L., Martínez, P. and Calvo, R. (2011) Evaluating the accessibility of three open-source learning content management systems: a comparative study. *Computer Applications in Engineering Education*. (June 2011) DOI= 10.1002/cae.20557

[16] Calvo, R., Gil, A., Iglesias, B. and Iglesias, A. (2013) Are Chats and Forums accessible in e-learning systems? A heuristic evaluation comparing four Learning Content Management Systems. *In proceedings of the* ITiCSE 2013, Canterbury, UK, July 1-3, 2013. ACM. (In press)

[17] Lazar, J., Allen, A. , Kleinman, J. And Malarkey C. (2007) What Frustrates Screen Reader Users on the Web:A Study of 100 Blind Users. *Internation Journal of Human-Computer Interaction.* 22, 3, 247–269.

[18] Harrison, L. (2002). Access to online learning: the role of the courseware authoring tool developer. Library Hi Tech. 20, 4. 433-440 DOI= 10.1108/07378830210452631

[19] Guenaga, M., Burguer, D. and Oliver, J. Computers Helping People with Special Needs. In *proceedings 9th International Conference, ICCHP 2004*, Paris, France, July 7-9, 2004 Lecture Notes in Computer Science, 2004, Volume 3118/2004, 626

[20] Moreno, L., Martínez, P. and Ruiz, B.(2008) Guiding accessibility issues in the design of websites. In *Proceedings of the 26th annual ACM international conference on Design of communication, SIGDOC 2008*, Lisbon, Portugal, Sept. 22-24, 2008. ACM, DOI= 10.1145/1456536.1456550

[21] Taras, C., Siemoneit, O., Weisser, N., Rotard, M. and Ertl, T. (2008) Improving the Accessibility of Wikis. *In Proceedings of the 11th International Conference on Computer Helping*

People with Special Needs, ICCHP 2008, Linz, Austria, July 9-11, 2008. Springer. DOI= 10.1007/978-3-540-70540-6_62

[22] W3C (2005), WAI, Conformance Evaluation of Websites for Accessibility. http://www.w3.org/WAI/eval/conformance.html

8 Annex 1

The Annex shows the errors found in the evaluation and they are summarized in Table 2 and Table 3. These tables are divided into four columns: Code, Description, Important Errors and Tools. The first column shows the code of the error. These codes are the groups in which the errors of the column *Important Errors* are grouped. To carry out this aggrupation, it has been considered the experience of the authors according to the nature of the errors. The column *Description* specifies more information about the error shown in the column *Code*. Finally, the column *Tools* specifies which are the tools that present this error.

Table 2. Error description and code of the Accessibility Problems Found in Chats and Forums. Errors E1-E7

Code	Description	Important Errors	Tools*
E1_ NonTextual Content	The non-textual content does not have alternative information	The images do not have alternative content or do not ask you to provide them. (A)	All
		Videos or audio cannot be uploaded with alternative content. (A)	MF| DF| CF |AF
		There are not labels for input text. (A)	All
E2_ Information	The information is not well structured, visualized or represented.	Headers are not used properly. (A)	All (Ex. AF)
		The title is not descriptive. (A)	All (Ex. CC)
		The links are not descriptive and there are link icons without identification purpose. (A)	CF | AF | AC
		Multiple ways of navigation (AA)	AC
		Coherent navigation (AA)	CF | CC
		Website language (A |AA)	All (Ex. AF| AC)
		Meaningful Sequence and visible focus. (A|AA)	CF| CC| DF | DC
		The tool uses pop-ups (A)	MF| MC| CF| CC
		The text contrast between text and background is not minimum (AA)	MF| MC|DF
E3_ Sensorial	There is information which is related to colors, shapes, etc.	There are icons which are not specified with text. (A)	All
		The tool uses color to specify information. (A)	CF | DF| MF | MC
E4_ Focus	The focus cannot be seen	Visible focus (AA)	CF | DF | DC
		Focus order (A)	AF | AC | CF
E5_ Personalization	The user cannot personalize the content or features	The user cannot control the audio. (A)	AF | AC
		The user cannot personalize the autorefresh (A)	DC | CC
		The time session is not adjustable. (ATAG) (A)	CF | CC
		Visual information cannot be stopped (ATAG) (A)	AF | AC
		The author settings are not preserved (ATAG) (A |AA)	AF | AC
E6_ Keyboard	The user cannot access to all the information through keyboard	There are elements which cannot be selected or there are keyboard traps. (A)	All (Ex. DF| CC)
		The web cannot be navigated sequentially by keyboard (A)	MF| MC| AC
		There are no mechanisms to skip content. (A)	CF | CC
		There are no shortcuts or they are overlapped (ATAG) (AA)	AC | CF | CC
E7_ ControlErrors	The tool does not inform the user about errors	The user can send information without text and the tool does not alert about it. (A |AA)	CF | CC | AC
		The user cannot cancel the action. (A)	DC

*Values: Moodle Forum (*MF*); Moodle Chat (*MC); dotLRN Forum (DF); dotRN Chat (DC);* ATutor Forum *(AF)*; ATutor Chat *(AC);* Claroline Forum *(CF)*; or Claroline Chat *(CC); All* if all tools present this error; or *All (Ex: XY)* which means that all the evaluated tools have the error except the tools specified in the parenthesis.

Table 3. Error description and code of the Accessibility Problems Found in Chats and Forums. Errors E8-E14

Code	Description	Important Errors	Tools*
E8_ StandardErrors	The tool does not accomplish standards and guidelines	The HTML and CSS code contain errors (A)	All (Ex. MF | MC)
E9_ EditingViews	The editing view is not accessible	The editing view does not show the status messages or the text presentation in a programmatically way. (A)	CF| CC| AF| AC
		The user cannot navigate through content structures. (AA)	All (Ex. CC)
		Text search through the content (AA)	MF| DF| AC| CF
		The content cannot be previewed properly. (A)	AF
E10_ ImproveSession	The tool does not autosave the information.	The tool does not autosave the information. (A)	CF |CC
E11_ Documentation	Accessibility documentation	The documentation is not complete and does not include information related to accessibility (A|AA)	All
E12_ Generation Accessible Content	Fully automatic processes must produce accessible content	The generated content during session is not accessible. (A)	All (Ex. CC)
		The content generated after session is not accessible (A)	MF|MC|DF|DC
		Transformations and copies do not preserve the accessibility (A)	CF | AF |AC
E13_ ProduceAccessibleContent	Authors must be supported in producing accessible content	The author has restrictions which do not allow him to create accessible content (A)	AF | CF
		Authors are not guided to produce accessible content (A)	CF| CC| AF| AC
		The tool does not check the accessibility (A)	All
		The tool provide default alternative text which does not identify the element (A)	AF
		Templates are not accessible and do not specify its accessibility level. (A)	All (Ex. CC)
E14_ AccessibilityFeatures	Authoring tools must promote and integrate their accessibility features	The accessibility features are not activated by default (A)	AF| AC
		If the user deactivates an accessibility function the tool does not inform about the problem. (AA)	AF| AC
		The tool does not inform about accessibility of each format and does not check it. (AA)	All
		There is not documentation related to accessibility and how to implement it (A)	All

*Values: Moodle Forum (*MF*); Moodle Chat (*MC);* dotLRN Forum *(DF);* dotRN Chat *(DC);* ATutor Forum *(AF)*; ATutor Chat *(AC);* Claroline Forum *(CF)*; or Claroline Chat *(CC); All* if all tools present this error; or *All (Ex: XY)* which means that all the evaluated tools have the error except the tools specified in the parenthesis.

802.11 Wireless Networks: Incorporating Hands-On Learning Experience into the Undergraduate Classroom

Mira Yun, Charlie Wiseman, Leonidas Deligiannidis

Wentworth Institute of Technology
Department of Computer Science and Networking
550 Huntington Avenue
Boston, MA 02115, USA
{yunm, wisemanc, deligiannidisl}@wit.edu

Abstract

Wireless networking and communication systems are already fundamental in typical day-to-day use of the Internet. This usage is continuing to increase as newer wireless technologies such as 4G LTE, WiMAX, and 802.11s gain traction. As such, it is vital that current students learn both the theoretical concepts and the practical details of wireless networking. This paper describes a course that gives students that opportunity. Moreover, this course is targeted at undergraduates in computer science and computer networking whereas most wireless courses are only taught at the graduate level and often in computer or electrical engineering departments. Details are included on teaching methods for concepts in wireless technology through example lab assignments. Although the course will continue to evolve as new wireless standards emerge, we believe that the course provides a solid basis for teaching the theory and practice of wireless communication systems.

Keywords – IEEE 802.11 Wireless Networks, undergraduate computer science and networking education, hands-on experience.

1. INTRODUCTION

Wireless and mobile Internet-enabled devices such as laptops, smartphones, and tablets are becoming increasingly popular and affordable. With their promise of “anywhere, anytime” access to the Internet, wireless technology has become a crucial component in communication and networking systems and technology.The advantages of wireless and mobile technologies are more evident than ever, with various wireless technologies including 4G Long Term Evolution (LTE), WiFi, Bluetooth, Near Field Communication (NFC), Global Positioning System (GPS) and 802.16 Worldwide Interoperability for Microwave Access (WiMAX) being continually developed and deployed [1]–[3]. With its growing popularity, wireless technology supports various emerging applications in the fields of medicine, on-line education, military operations, industrial management, home automation and many others [4]-[7].

In response to the increasing demand for wireless networking skills in almost all industries, it is crucial to teach and provide our students with the knowledge to understand the state of the art in wireless technologies and applications, to help identify the current trends and challenges, and eventually to excel in their careers. Despite the popularity of the topic and increasing industry demand, courses or laboratories on wireless and mobile networks are not frequently offered at the undergraduate level; this is mainly due to the fact that the topic is relatively new and that it requires greater research. Unlike graduate students, teaching a wireless course to undergraduate students is a difficult task because the study requires the students to have strong knowledge in both technical implementation and theoretical concepts. The nature of the subject, which tends to be dry and heavily based on theory, makes the materials covered in class difficult to digest for many undergraduate students. Thus, introducing hands-on learning activities into the course study can dramatically improve the quality of the classroom experience by allowing students to engage with the topic in a practical manner [8]-[14]. As one of the leading undergraduate polytechnic institutes, the Computer Science and Networking Department at Wentworth Institute of Technology (WIT), Boston, sees the importance of active participation and has developed a hands-on wireless networks course for undergraduate students [15].

In this paper, we will provide an overview of this hands-on 802.11 wireless networks course for undergraduate juniors and seniors with specific examples of assignments and projects.

2. TEACHING WIRELESS NETWORKS

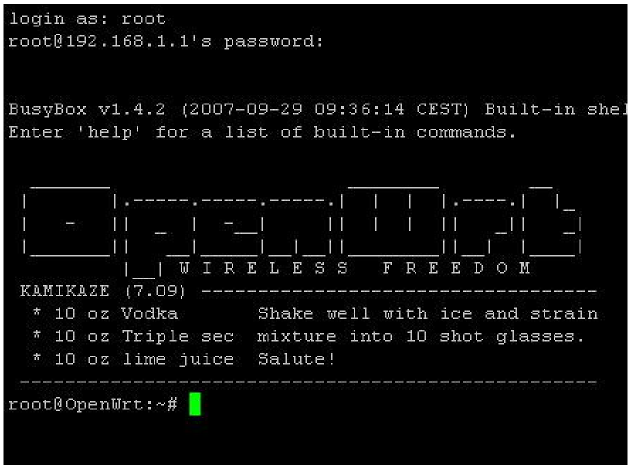
Teaching engineering and computer networking courses is a demanding task and continuous research has been made to develop effective teaching methods [8]-[14]. Although many diverse methods have been proposed, they all agree that incorporating hands-on learning experiences into these engineering courses significantly enhances student learning about engineering and networking [16][17]. However, it is difficult to provide practical demonstrations or hands on learning materials for engineering students at the undergraduate level. In response to this problem, Sarkar and Craig [8] presented several interesting Wi-Fi project ideas including infrared remote control, Wi-Fi Antenna, and Ad Hoc Networks to provide the students of wireless communication and networking with a hands-on learning experience. Guzelgoz and Arslan [9] presented eight fundamental wireless communication experiments including waveforms, modulation, synchronization, and channel impact in wireless communications for electrical engineering education. In extension of these efforts, several wireless laboratories for education and research have been developed to enhance student learning about wireless communications and networking technology. Most of these wireless laboratories focus on providing low cost and commercially available systems and devices to students. Linn [10] used a Xilinx Spartan-3 A Starter Kit board [18]. Chenard, Zilic, and Prokic [11] used the microprocessor system kit McGumps [19] [20] to setup a wireless and mobile embedded laboratory. However, these wireless laboratories were designed for engineering students and did not cover the same concepts and material required in computer science and networking. For example, they focused on the physical transmission of wireless signals and waveforms rather than communication protocols and security concerns.

Furthermore, as new wireless technologies and networks continue to be developed and updated, building a new experimental environment for each system and network is expensive and difficult. Thus, Wang and Jiang [12] used Network Simulator (NS)-3, a free software simulation platform, as an effective means of wireless local area network teaching. Instead of relying solely on existing free software such as NS-2[21], OPNET[22], and QualNet (formerly GloMoSim) [23], Sanguino et al. [13] developed a new educational wireless network simulator, WiFiSiM, devoted to the study and evaluation of wireless networks. Momeni and Kharrazi [14] presented a combined approach with both software simulations and physical network topologies.

It is interesting to note however that none of these previous endeavors are targeted at undergraduate computer networking students. In this paper, several Wi-Fi experiments for computer networking students are introduced. These experiments are designed with commercially available Linksys WRT54G series routers [24], which cost less than $50 USD, and other various software tools including aircrack-ng [25], InSSIDer [26], and Wireshark [27].

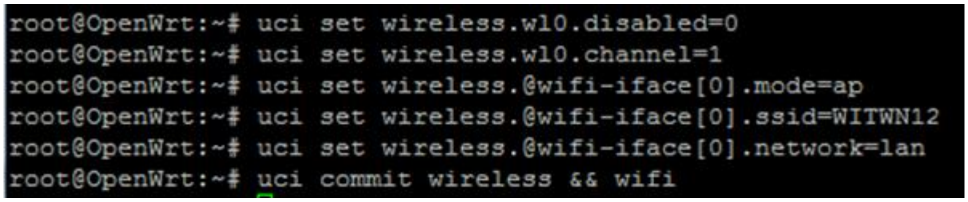
3. 802.11 WIRELESS EXPERIMENTS

The 802.11 Wireless Networks course is a four-credit course with 3 hours lecture and 2 hours lab weekly designed for junior or senior level undergraduate students. Theoretical aspects including the fundamental principles, architecture, and standards of modern WiFi communication systems are discussed during the lecture session. The principles of wireless communications are covered including basic terms and concepts, modulation, spread spectrum, multiplexing, antenna, Orthogonal Frequency Division Multiplexing (OFDM), and Code Division Multiple Access (CDMA). Based on these fundamentals, IEEE 802.11 a/b/g/n wireless standards are discussed in detail. This course also addresses the physical layer medium access control (MAC) protocols and their security protocols including Wired Equivalent Privacy (WEP), Wi-Fi Protected Access (WPA), and 802.11i (WPA2). Various experiments from a single wireless access point (AP) to an advanced wireless distribution system (WDS) are conducted in order to strengthen student comprehension of the theoretical material. The major experiments are described in the following sections.



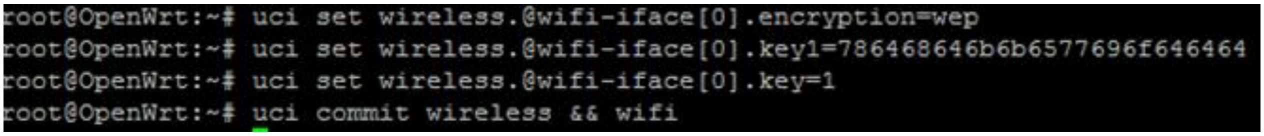
```
login as: root
root@192.168.1.1's password:


BusyBox v1.4.2 (2007-09-29 09:36:14 CEST) Built-in shel
Enter 'help' for a list of built-in commands.

  _______                     ________        __
 |       |.-----.-----.-----.|  |  |  |.----.|  |_
 |   -   ||  _  |  -__|     ||  |  |  ||   _||   _|
 |_______||   __|_____|__|__||________||__|  |____|
          |__| W I R E L E S S   F R E E D O M
 KAMIKAZE (7.09) -----------------------------------
  * 10 oz Vodka       Shake well with ice and strain
  * 10 oz Triple sec  mixture into 10 shot glasses.
  * 10 oz lime juice  Salute!
 ---------------------------------------------------
root@OpenWrt:~#
```

Figure 1. OpenWRT Login Page



```
root@OpenWrt:~# uci set wireless.wl0.disabled=0
root@OpenWrt:~# uci set wireless.wl0.channel=1
root@OpenWrt:~# uci set wireless.@wifi-iface[0].mode=ap
root@OpenWrt:~# uci set wireless.@wifi-iface[0].ssid=WITWN12
root@OpenWrt:~# uci set wireless.@wifi-iface[0].network=lan
root@OpenWrt:~# uci commit wireless && wifi
```

Figure 2. UCI Commands for Wireless Network



```
root@OpenWrt:~# uci set wireless.@wifi-iface[0].encryption=wep
root@OpenWrt:~# uci set wireless.@wifi-iface[0].key1=786468646b6b6577696f646464
root@OpenWrt:~# uci set wireless.@wifi-iface[0].key=1
root@OpenWrt:~# uci commit wireless && wifi
```

Figure 3. UCI Commands for WEP Encryption Setup

3.1. Basic Wireless Network Setup

Many wireless routers actually run the Linux operating system with a custom web interface that allows users to do configuration with a minimum of technical know-how. OpenWRT [28] is essentially an open source Linux distribution for embedded devices, designed to fit into a small memory foot print. It was originally developed specifically for the Linksys WRT54G series routers; however, many other devices are supported today. Most of these compatible devices use the same Broadcom processor, but some other processors are also used. The idea behind OpenWRT was to open up functionality that was not originally available and also to provide a Linux frame work for customizing your device to do far more than it was originally designed to do. In order to explore various functionalities of off-the-shelf wireless APs, the course uses Linksys WRT54G series routers running the third party firmware from OpenWRT.

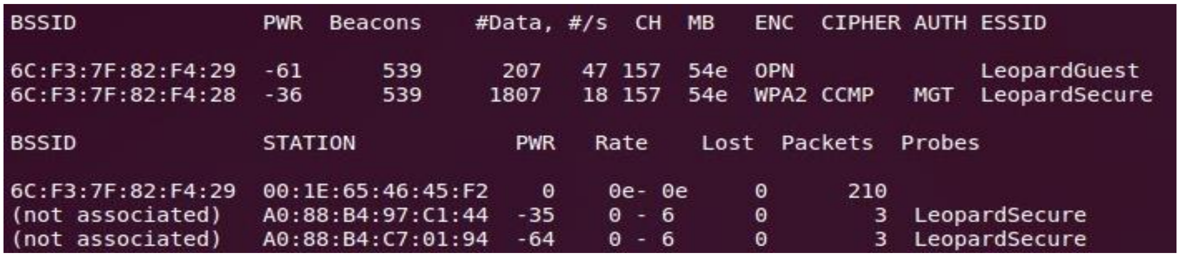
After installing OpenWRT on the WRT54G router, it is possible to connect to it via Secure Shell (SSH) or the web. Figure 1 shows logging on to the router with the root account and password through the SSH command `ssh root@192.168.1.1`.

Students first learn how to set up a wireless network by configuring wireless parameters on OpenWRT. Wireless specific (Layers 1 and 2) configurations are stored in `/etc/config/wireless`. Layer 3 parameters are stored in `/etc/config/network`. Students can manually modify these two configuration files using the vi editor [29]. Because there is no syntax checking inside vi, however, the Unified Configuration Interface (UCI) command [30] is used to set various parameters inside these two files directly. Figure 2 shows how to enable Wi-Fi networking, place the AP on a particular channel, and set the customized service set identification (SSID, i.e. the public name of a wireless network) by using uci commands.

Next, the security schemes are set up through additional command line configuration. Figure 3 shows how to configure WEP encryption with UCI commands. As per the standard, up to four WEP keys can be configured. The procedure is well documented in [31].

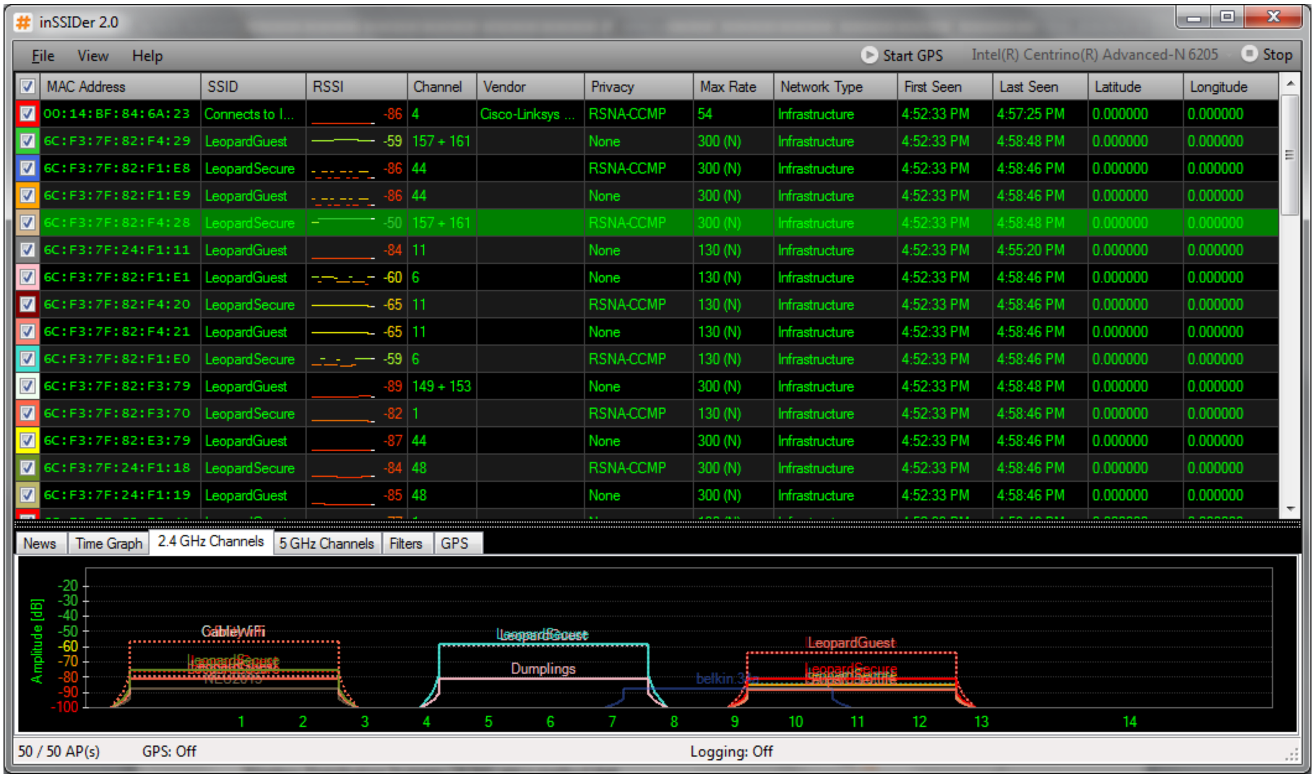
After all configuration is completed, students are asked to check the `/etc/config/wireless` file to verify that all changes have taken place. There are several methods and tools to get more detailed wireless information. Through one of the wireless scanning tools, students are able to verify that their wireless network is set up and functioning correctly. Figure 4 uses airodump-ng [25], a Linux based wireless packet capture tool. Figure 5 shows the scanning result of InSSIDer [26], a Windows based WiFi scanning software suite.

Finally, students move to the more advanced wireless network parameters such as antenna adjustment, rudimentary penetration testing, MAC filter setup, etc [28].



```
BSSID              PWR  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

6C:F3:7F:82:F4:29  -61      539      207   47 157  54e  OPN              LeopardGuest
6C:F3:7F:82:F4:28  -36      539     1807   18 157  54e  WPA2 CCMP   MGT  LeopardSecure

BSSID              STATION            PWR   Rate    Lost  Packets  Probes

6C:F3:7F:82:F4:29  00:1E:65:46:45:F2    0    0e- 0e    0      210
(not associated)   A0:88:B4:97:C1:44  -35    0 - 6     0        3  LeopardSecure
(not associated)   A0:88:B4:C7:01:94  -64    0 - 6     0        3  LeopardSecure
```

Figure 4. Linux Method: airodump-ng



☑	MAC Address	SSID	RSSI	Channel	Vendor	Privacy	Max Rate	Network Type	First Seen	Last Seen	Latitude	Longitude
☑	00:14:BF:84:6A:23	Connects to I...	-86	4	Cisco-Linksys ...	RSNA-CCMP	54	Infrastructure	4:52:33 PM	4:57:25 PM	0.000000	0.000000
☑	6C:F3:7F:82:F4:29	LeopardGuest	-59	157 + 161		None	300 (N)	Infrastructure	4:52:33 PM	4:58:48 PM	0.000000	0.000000
☑	6C:F3:7F:82:F1:E8	LeopardSecure	-86	44		RSNA-CCMP	300 (N)	Infrastructure	4:52:33 PM	4:58:46 PM	0.000000	0.000000
☑	6C:F3:7F:82:F1:E9	LeopardGuest	-86	44		None	300 (N)	Infrastructure	4:52:33 PM	4:58:46 PM	0.000000	0.000000
☑	6C:F3:7F:82:F4:28	LeopardSecure	-50	157 + 161		RSNA-CCMP	300 (N)	Infrastructure	4:52:33 PM	4:58:48 PM	0.000000	0.000000
☑	6C:F3:7F:24:F1:11	LeopardGuest	-84	11		None	130 (N)	Infrastructure	4:52:33 PM	4:55:20 PM	0.000000	0.000000
☑	6C:F3:7F:82:F1:E1	LeopardGuest	-60	6		None	130 (N)	Infrastructure	4:52:33 PM	4:58:46 PM	0.000000	0.000000
☑	6C:F3:7F:82:F4:20	LeopardSecure	-65	11		RSNA-CCMP	130 (N)	Infrastructure	4:52:33 PM	4:58:46 PM	0.000000	0.000000
☑	6C:F3:7F:82:F4:21	LeopardGuest	-65	11		None	130 (N)	Infrastructure	4:52:33 PM	4:58:46 PM	0.000000	0.000000
☑	6C:F3:7F:82:F1:E0	LeopardSecure	-59	6		RSNA-CCMP	130 (N)	Infrastructure	4:52:33 PM	4:58:46 PM	0.000000	0.000000
☑	6C:F3:7F:82:F3:79	LeopardGuest	-89	149 + 153		None	300 (N)	Infrastructure	4:52:33 PM	4:58:48 PM	0.000000	0.000000
☑	6C:F3:7F:82:F3:70	LeopardSecure	-82	1		RSNA-CCMP	130 (N)	Infrastructure	4:52:33 PM	4:58:46 PM	0.000000	0.000000
☑	6C:F3:7F:82:E3:79	LeopardGuest	-87	44		None	300 (N)	Infrastructure	4:52:33 PM	4:58:46 PM	0.000000	0.000000
☑	6C:F3:7F:24:F1:18	LeopardSecure	-84	48		RSNA-CCMP	300 (N)	Infrastructure	4:52:33 PM	4:58:46 PM	0.000000	0.000000
☑	6C:F3:7F:24:F1:19	LeopardGuest	-85	48		None	300 (N)	Infrastructure	4:52:33 PM	4:58:46 PM	0.000000	0.000000

Figure 5. Windows Method: InSSIDer

3.2. Wireless Distribution System

Wireless Distribution Systems (WDS) are one way to enable the wireless interconnection of APs (or routers) in an IEEE 802.11 network. It allows a wireless network to be expanded using multiple APs without the need for a wired backbone to link them together, as it has been traditionally required. In WDS, an AP (or router) can be either a main, relay, or remote AP as shown in Figure 6. A main AP is typically connected to the Internet. A relay AP relays data from remote APs, wireless clients or other relay APs to either a main or another relay AP. A remote AP accepts connections from wireless clients and passes them on to the relay or main APs.

WDS can be used to provide two modes of wireless AP-to-AP connectivity:

1) *Bridge mode*: wireless bridging in which WDS APs communicate only with each other and do not allow wireless clients or stations to access them.
2) *Repeater mode*: wireless repeating in which APs communicate with each other and with wireless stations. Repeater mode is more advanced than bridge mode.

All APs in a WDS must be configured to use the same radio channel, and share WEP keys or WPA keys if they are used. They can be configured to use different SSIDs. WDS also requires that every AP be configured to forward to others in the system.

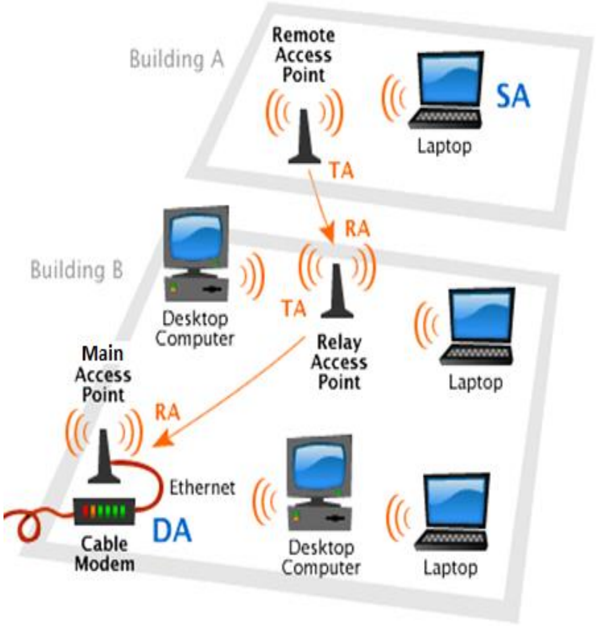
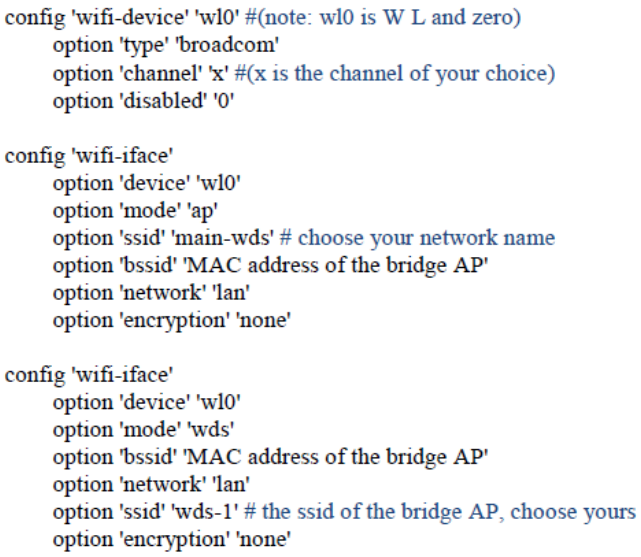


Figure 6. Wireless Distribution System



```
config 'wifi-device' 'wl0' #(note: wl0 is W L and zero)
        option 'type' 'broadcom'
        option 'channel' 'x' #(x is the channel of your choice)
        option 'disabled' '0'

config 'wifi-iface'
        option 'device' 'wl0'
        option 'mode' 'ap'
        option 'ssid' 'main-wds' # choose your network name
        option 'bssid' 'MAC address of the bridge AP'
        option 'network' 'lan'
        option 'encryption' 'none'

config 'wifi-iface'
        option 'device' 'wl0'
        option 'mode' 'wds'
        option 'bssid' 'MAC address of the bridge AP'
        option 'network' 'lan'
        option 'ssid' 'wds-1' # the ssid of the bridge AP, choose yours
        option 'encryption' 'none'
```

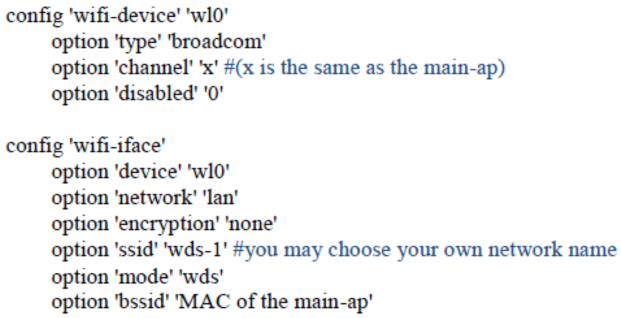
Figure 7. Main AP Wireless Configuration

3.4.1 Setup Bridge Mode

In this section, we illustrate a detailed example of how to set up a WDS in bridge mode with two or more APs. From two APs, we choose one AP as the main AP and name it as “`main-ap`”. The other AP will be a wireless bridge in the WDS mode and is named as “`wds-1`”. Of course students may name their APs in their preferred manner, perhaps including their username, so that they can be distinguished easily in the classroom.

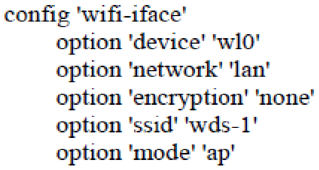
Here are the steps:

1) Open a new terminal and SSH to 192.168.1.1. Log on to each AP and get the MAC address (a.k.a, basic service set identification (BSSID)) of each router using the command: `ifconfig wl0.`
2) **Configure the Main AP:** Wireless specific configurations are stored in `/etc/config/wireless`. Backup the existing wireless configuration file and then modify the wireless file as shown in Figure 7.
3) **Configure the Bridge AP:** Change the IP address of the bridging AP (`wds-1`) to 192.168.1.2 (it can be any IP that is not 192.168.1.1). Go the `/etc/config` directory and edit the file named `network` to update the IP address (192.168.1.2).
4) Go to the `/etc/init.d` directory and disable the domain name system (DNS) and firewall services through the `rm dnsmasq` and `rm firewall` commands.
5) Go to the `/etc/config` directory and edit the file named `dhcp`. Remove all the lines that match these two entries: `config dhcp lan` and `config dhcp wan`.



```
config 'wifi-device' 'wl0'
        option 'type' 'broadcom'
        option 'channel' 'x' #(x is the same as the main-ap)
        option 'disabled' '0'

config 'wifi-iface'
        option 'device' 'wl0'
        option 'network' 'lan'
        option 'encryption' 'none'
        option 'ssid' 'wds-1' #you may choose your own network name
        option 'mode' 'wds'
        option 'bssid' 'MAC of the main-ap'
```

Figure 8. Bridge AP Wireless Configurations



```
config 'wifi-iface'
        option 'device' 'wl0'
        option 'network' 'lan'
        option 'encryption' 'none'
        option 'ssid' 'wds-1'
        option 'mode' 'ap'
```

Figure 9. Repeater AP Wireless Configurations

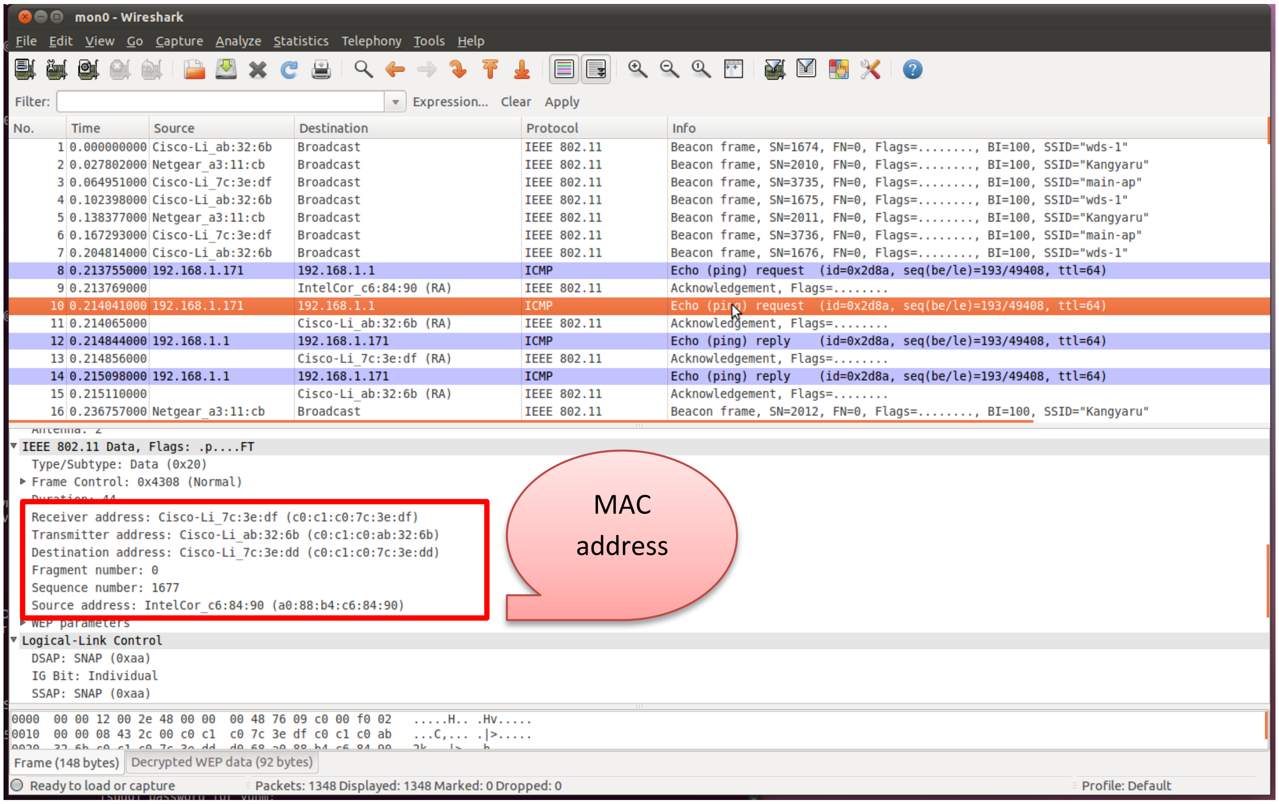
6) Backup the `/etc/config/wireless` file and then modify the wireless file as shown in Figure 8.

After rebooting each router, students need to verify their setup. If students connect a laptop (say laptop-a) to `main-ap` wirelessly and another laptop (say laptop-b) to the bridge AP using a network cable (the laptop needs to use DHCP instead of static IP), then laptop-b should obtain an IP address even though the bridge AP does not offer dynamic host configuration protocol (DHCP) services.

3.4.2 Setup Repeater Mode

In order to setup repeater mode, the bridge AP should be tuned into a relay AP (repeater mode). As shown in Figure 9, an extra `wifi-iface` configuration should be appended into the bridge AP wireless file in `/etc/config/wireless`. Nothing needs to be changed on `main-ap`.

After rebooting the router, students need to verify their setup. If students connect a laptop (say laptop-a) to `main-ap` wirelessly and another laptop (say laptop-b) to the relay AP (`wds-1`) wirelessly (the laptop needs to use DHCP instead of static IP), then laptop-b should obtain an IP address. If students ping laptop-a from laptop-b, students can capture the ping request and reply frame pair (between the two laptops) by using a packet capturing tool such as Wireshark [30]. Figure 10 shows an example of a ping request frame to check how four MAC addresses (receiver, transmitter, destination, and source) are set in the WDS mode.


mon0 - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter:
Expression... Clear Apply
MAC address
Frame (148 bytes)
Decrypted WEP data (92 bytes)
Profile: Default

```
No. Time         Source             Destination             Protocol     Info
  1 0.000000000 Cisco-Li_ab:32:6b  Broadcast               IEEE 802.11  Beacon frame, SN=1674, FN=0, Flags=........, BI=100, SSID="wds-1"
  2 0.027802000 Netgear_a3:11:cb   Broadcast               IEEE 802.11  Beacon frame, SN=2010, FN=0, Flags=........, BI=100, SSID="Kangyaru"
  3 0.064951000 Cisco-Li_7c:3e:df  Broadcast               IEEE 802.11  Beacon frame, SN=3735, FN=0, Flags=........, BI=100, SSID="main-ap"
  4 0.102398000 Cisco-Li_ab:32:6b  Broadcast               IEEE 802.11  Beacon frame, SN=1675, FN=0, Flags=........, BI=100, SSID="wds-1"
  5 0.138377000 Netgear_a3:11:cb   Broadcast               IEEE 802.11  Beacon frame, SN=2011, FN=0, Flags=........, BI=100, SSID="Kangyaru"
  6 0.167293000 Cisco-Li_7c:3e:df  Broadcast               IEEE 802.11  Beacon frame, SN=3736, FN=0, Flags=........, BI=100, SSID="main-ap"
  7 0.204814000 Cisco-Li_ab:32:6b  Broadcast               IEEE 802.11  Beacon frame, SN=1676, FN=0, Flags=........, BI=100, SSID="wds-1"
  8 0.213755000 192.168.1.171      192.168.1.1             ICMP         Echo (ping) request  (id=0x2d8a, seq(be/le)=193/49408, ttl=64)
  9 0.213769000                    IntelCor_c6:84:90 (RA)  IEEE 802.11  Acknowledgement, Flags=........
 10 0.214041000 192.168.1.171      192.168.1.1             ICMP         Echo (ping) request  (id=0x2d8a, seq(be/le)=193/49408, ttl=64)
 11 0.214065000                    Cisco-Li_ab:32:6b (RA)  IEEE 802.11  Acknowledgement, Flags=........
 12 0.214844000 192.168.1.1        192.168.1.171           ICMP         Echo (ping) reply    (id=0x2d8a, seq(be/le)=193/49408, ttl=64)
 13 0.214856000                    Cisco-Li_7c:3e:df (RA)  IEEE 802.11  Acknowledgement, Flags=........
 14 0.215098000 192.168.1.1        192.168.1.171           ICMP         Echo (ping) reply    (id=0x2d8a, seq(be/le)=193/49408, ttl=64)
 15 0.215110000                    Cisco-Li_ab:32:6b (RA)  IEEE 802.11  Acknowledgement, Flags=........
 16 0.236757000 Netgear_a3:11:cb   Broadcast               IEEE 802.11  Beacon frame, SN=2012, FN=0, Flags=........, BI=100, SSID="Kangyaru"
```

```
  Antenna: 2
▼ IEEE 802.11 Data, Flags: .p....FT
    Type/Subtype: Data (0x20)
  ▶ Frame Control: 0x4308 (Normal)
    Duration: 44
    Receiver address: Cisco-Li_7c:3e:df (c0:c1:c0:7c:3e:df)
    Transmitter address: Cisco-Li_ab:32:6b (c0:c1:c0:ab:32:6b)
    Destination address: Cisco-Li_7c:3e:dd (c0:c1:c0:7c:3e:dd)
    Fragment number: 0
    Sequence number: 1677
    Source address: IntelCor_c6:84:90 (a0:88:b4:c6:84:90)
  ▶ WEP parameters
▼ Logical-Link Control
    DSAP: SNAP (0xaa)
    IG Bit: Individual
    SSAP: SNAP (0xaa)

0000  00 00 12 00 2e 48 00 00  00 48 76 09 c0 00 f0 02   .....H.. .Hv.....
0010  00 00 08 43 2c 00 c0 c1  c0 7c 3e df c0 c1 c0 ab   ...C,... .|>.....

Ready to load or capture    Packets: 1348 Displayed: 1348 Marked: 0 Dropped: 0
```

Figure 10. Wireshark Example: Ping Request Frame

4. CONCLUSION AND FUTURE WORK

The novelty of this work lies in developing a course targeted for undergraduates in computer science and computer networking. Most wireless courses are only taught at the graduate level and often in computer or electrical engineering departments. Through the combined hardware and software experiments, students become engaged in the topic and learn the material from a practical perspective. Course evaluation based on student feedback shows a high level of enthusiasm and engagement among the students in the wireless course. Most importantly, the students learned about current state of the art IEEE 802.11 wireless technologies in depth due to the strong hands-on learning nature of the course. In addition, this course addressed the physical layer specification and the medium access control protocols as well as their advanced applications such as WDS. In the future, the course will include wireless security experiments. Current wireless security protocols including WEP, WPA, and 802.11i can be circumvented due to a flaw that exists in the authentication procedure [32]-[34]. Providing students with an opportunity to see how these standards can be cracked and broken will enable them to build practical experience with current security issues and become more engaged in the topic of wireless security.

REFERENCES

[1] Yi, L.; Kai Miao; Liu, A., "A comparative study of WiMAX and LTE as the next generation mobile enterprise network," Advanced Communication Technology (ICACT), 2011 13th International Conference on, vol., no., pp.654,658, 13-16 Feb. 2011

[2] Michahelles, F.; Thiesse, Frederic; Schmidt, A.; Williams, J.R., "Pervasive RFID and Near Field Communication Technology," Pervasive Computing, IEEE , vol.6, no.3, pp.94,96, c3, July-Sept. 2007

[3] Want, R., "Near field communication," Pervasive Computing, IEEE , vol.10, no.3, pp.4,7, July-September 2011

[4] Tang Yu-liang; Luo Yu; Huang Lian-fen; Guo Jian; Lei Ying, "Wireless sensor network for on-line structural health monitoring," Computer Science & Education (ICCSE), 2012 7th International Conference on , vol., no., pp.386,389, 14-17 July 2012

[5] Yuksekkaya, B.; Kayalar, A.A.; Tosun, M.B.; Ozcan, M.K.; Alkar, A.Z., "A GSM, internet and speech controlled wireless interactive home automation system," Consumer Electronics, IEEE Transactions on , vol.52, no.3, pp.837,843, Aug. 2006

[6] Danielle Bragg, Mira Yun, Haya Bragg, and Hyeong-Ah Choi, "Intelligent Transmission of Patient Sensor Data in Wireless Hospital Networks", Proc. AMIA(American Medical Informatics Association) 2012 symposium, Chicago, Nov 2012

[7] Mira Yun, Danielle Bragg, Amrinder Arora, and Hyeong-Ah Choi, "Battle Event Detection using Sensor Networks and Distributed Query Processing", Proc. INFOCOM Workshops 2011, IEEE, CPNS 2011, Shanghai, China, April 2011

[8] Sarkar, N.I.; Craig, T.M., "Teaching wireless communication and networking fundamentals using Wi-Fi projects," Education, IEEE Transactions on , vol.49, no.1, pp.98,104, Feb. 2006

[9] Güzelgöz, S.; Arslan, H., "A Wireless Communications Systems Laboratory Course," Education, IEEE Transactions on , vol.53, no.4, pp.532,541, Nov. 2010

[10] Linn, Y., "An Ultra Low Cost Wireless Communications Laboratory for Education and Research," Education, IEEE Transactions on , vol.55, no.2, pp.169,179, May 2012

[11] Chenard, J-S; Zilic, Z.; Prokic, M., "A Laboratory Setup and Teaching Methodology for Wireless and Mobile Embedded Systems," Education, IEEE Transactions on , vol.51, no.3, pp.378,384, Aug. 2008

[12] Anbao Wang; WenRong Jiang, "Teaching Wireless Local Area Network Course Based on NS-3," Computer Network and Multimedia Technology, 2009. CNMT 2009. International Symposium on , vol., no., pp.1,4, 18-20 Jan. 2009

[13] Mateo Sanguino, T. J.; Serrano Lopez, C.; Marquez Hernandez, F. A., "WiFiSiM: An Educational Tool for the Study and Design of Wireless Networks," Education, IEEE Transactions on , vol.PP, no.99, pp.1,1, 2012

[14] Momeni, B.; Kharrazi, M., "Improving a Computer Networks Course Using the Partov Simulation Engine," Education, IEEE Transactions on , vol.55, no.3, pp.436,443, Aug. 2012

[15] Leonidas Deligiannidis, Charlie Wiseman, Mira Yun, and Tom Goulding, "Network Security Course: A Demonstration of Project-Based Learning", In Proc. of the 2012 International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS'12), pp.28-34, July 16-19 2012, Las Vegas NV, USA.

[16] M.F. Young, "Instructional design for situated learning," Educ. Technol., vol 41, pp. 43-58, 1993

[17] J.R. Anderson, L. M. Reder, and H.A. Simon, "Situated learning and education," in Educ. Res., 1996, vol. 25, pp. 5-11.

[18] Spartan 3 A Starter Kit HW-SPAR3A-SK-UNI-G. Xilinx Inc., San Jose, CA, May 2009 [Online]. Avaiable: http://www.xilinx.com

[19] J.-S. Chenard, U. Khalid, M. Prokic, R. Zhang, K.-L. Lim, A. Chattopadhyay, and Z. Zilic, "Expandable and robust laboratory for microprocessor systems," in Proc. IEEE Int. Conf. Microelectronic Systems Education, Anaheim, CA, Jun. 2005, pp. 65–66.

[20] Z. Zilic, J.-S. Chenard, and M. Prokic, "A laboratory for wireless and mobile embedded systems," in Proc. IEEE Int. Conf. Microelectronic Systems Education, San Diego, CA, Jun. 2007, pp. 103–104.

[21] T. Issariyakul and E. Hossain, Introduction to Network Simulator NS2. New York: Springer, 2008.

[22] G. Flores Lucio, M. Paredes-Farrera, E. Jammeh, M. Fleury, and M. J. Reed, "OPNET modeler and Ns-2: Comparing the accuracy of network simulators for packet-level analysis using a network testbed," in Proc. 3rd WEAS ICOSMO, 2003, vol. 2, pp. 700–707.

[23] X. Zeng, R. Bagrodia, and M. Gerla, "GloMoSim: A library for parallel simulation of large-scale wireless networks," inProc. 12th PADS, 1998, pp. 154–161.

[24] Cisco-Linksys WRT54G Wireless-G Router, Aug 2044, http://www.linuxjournal.com/article/7322

[25] Aircrack-ng, http://www.aircrack-ng.org/

[26] InSSIDer, http://www.metageek.net/products/inssider/

[27] Wireshark, http://www.wireshark.org/

[28] OpenWRT, https://openwrt.org/

[29] The Open Group (1997). "vi — screen-oriented (visual) display editor", Version 2, http://pubs.opengroup.org/onlinepubs/007908799/xcu/vi.html

[30] UCI, http://wiki.openwrt.org/doc/techref/uci

[31] OpenWRT Basic Configuration, http://wiki.openwrt.org/doc/howto/basic.config

[32] Boland, H.; Mousavi, H.; "Security issues of the IEEE 802.11b wireless LAN," Electrical and Computer Engineering, 2004. Canadian Conference on , vol.1, no., pp. 333- 336 Vol.1, 2-5 May 2004.

[33] Hal Berghel and Jacob Uecker, "WiFi attack vectors". Commun. ACM 48, August 2005

[34] Scott R. Fluhrer, Itsik Mantin, and Adi Shamir, "Weaknesses in the Key Scheduling Algorithm of RC4", In Revised Papers from the 8th Annual International Workshop on Selected Areas in Cryptography (SAC '01), Serge Vaudenay and Amr M. Youssef (Eds.). Springer-Verlag, London, UK, 1-24.

Blogging
and Online Collaborative Discovery Learning

Making a case for a successful group-tracking technique

Zeenath Reza Khan[1]
[1]Faculty of Computer Science and Engineering, University of Wollongong in Dubai, Dubai UAE

Abstract - *With the surge of digital native students entering classrooms, instructors worldwide have been trying hard to incorporate ICTs into their pedagogy in order to enhance student learning environment and prepare better graduates for the employment market. Among various shifts in paradigms, online collaborative discovery learning (OCDL) has become a coin phrase that instructors want to include in their many attempts at responding to the needs of the digital native students. With the increasing popularity of OCDL, researchers have studied issues related to OCDL such as instructor lack of understanding of OCDL and how it can or should be incorporated into pedagogy; studies have also highlighted issues with actual assessment of collaborative projects. However, few studies have actually studied or even recognized the issue of tracking these collaborative efforts of students, monitoring their progress, reducing conflicts and enhancing students' overall collaborative experience.*

This papers identifies three barriers to creating a successful OCDL environment in terms and proposes an adaptation and implementation of a blended learning tool, blogging, to help both instructors and students track their groups.

Keywords: e-learning, weblogs, social networking, education, blended learning

1 Introduction

Discovery learning allows students to work independently and become active participants in their learning process ([1];[2];[3];[4];[5];[6]) while collaborative learning (OL) is a strategy used by instructors to increase efficacy and longevity of learning through cooperative strategies between students and their instructors [7]. One such strategy implemented has been to use small student groups for assessment completion [8]. Collaborative discovery learning allows participants to strategize and execute their chosen strategies together while online collaborative discovery learning (OCDL) is an environment that allows students to work together in cyber space and find solutions to problems they have been given independently [9].

However, where instructors have marveled at how fast students learn and retain information when working in groups, be it in a face-to-face classroom setting or online, they have also voiced concerns over issues such as actually managing group work, ensuring fair grading of all members of the group, tracking amount of contribution by each member, and maintaining honesty and harmony within groups.

This paper records the experience of the author in successful adaptation and implementation of weblogs to track student group-work, enhance student learning and increase group homogeneity thus increasing student participation in a computer applications course at the University of Wollongong in Dubai.

2 Digital Natives and blended learning

Information communication technology (ICT) is defined as any and every communication and computer software or hardware technology [10]

Due to the rapid growth of ICT [Khan, 2012] towards the end of 20th century and early 21st century, studies suggest that students in classrooms today represent a new generation who have grown up with new technology ([11]; [12]; [13]), and fundamentally process information and think differently than their predecessors. This generation is sometimes called the digital natives [14]. Digital natives are said to interact, learn, and communicate differently because they have never known a world without ICTs such as computers, Web, games, smart phones, Internet among others [15].

Studies suggest that instructors have been introducing new systems and tools to their teaching environments in order to meet the expectations of the digital natives ([16];[17]), giving rise to the concept of blended learning.

Blended learning is now a common place in higher education, allowing instructors to combine face-to-face classroom setting and ICTs. These ICTs include computers, the Internet, phones, wikis (technology that allows website to be collaboratively constructed and edited), weblogs (blog is a website that contain dated entries; can be individual or collective entries that can be bookmarked, referenced or commented on by others), and even learning management systems (web-based tools that instructors use to plan and deliver courses) and allow students to engage in various pedagogical fits that suit their learning styles in real-time and sequentially over the Internet [18].

3 CDL and higher education

CDL is defined as a process of learning that is generated by student groups who are interdependent [8] and who work towards solving a given problem themselves [19].

According to some research, CDL helps students develop their skills and knowledge, ultimately shaping a learning community [20].

Studies have shown that there are other benefits of collaborative studies in higher education. One such study suggests that the CDL helps develop critical thinking in students, helps better understand concepts, allows for sharing of information, and even helps develop communication [21].

A shift in educational thinking from teacher-centered learning to student-centered learning environments has also encouraged the use of CDL by instructors ([6];[22]). Research has shown that CDL enhances learner’s cognitive processes [14]. Other studies propose that social interactions particularly among students with similar cognitive capabilities aid in enhancing cognitive skills in students [6] further rationalizing the implementation of CDL in higher education.

Another reason educators try to incorporate CDL is in response to “employability agenda” [23]. Research suggests that the connection between higher education and the industry is the gradate employability [23]. Industry expects higher education institutions to instill qualities in their graduates that the corporate world seeks. In this case, although the term “group” may not reflect the term “team”, research suggests that CDL environments perhaps help develop communication and teamwork skills that may make graduates effective team players at work [19], and thus preferred would-be employees.

4 OCDL and issues in higher education

With blended learning incorporated more into higher education institutes, and CDL gaining popularity among instructors, studies suggest that CDL has evolved to become OCDL where instructors use, among other tools, online technologies such as wikis, weblogs, and even social networking sites such as Facebook to enhance learning management systems, change the learning process from unidirectional to facilitate self-governed, problem-based and collaborative activities [24].

Kozma and Anderson [33] have defined the impact of ICT in HE as a bridge between classrooms and the real world, providing tools to enhance learning, allowing students and teachers more flexibility and opportunity for communication and collaboration.

Some studies suggest that specific ICTs such as weblogs have been used in higher education successfully to enhance OCDL ([25];[26];[27];28]). Similarly, studies suggest that wikis have been effective in OCDL in higher education [29];[30];[31];[32]).

However, literature focuses mostly on ICTs as OCDL tools. Many studies have suggested wikis or blogs as tools to enhancing student OCDL [19], or studies propose types of assessments that can use OCDL ([30];[31];[32]). Some studies have even proposed motivations for using OCDL tools such as weblogs in allowing sharing of ideas, recording ideas and making new contacts [34].

Problems proposed by studies highlight instructors’ lack of clear understanding of ICTs and OCDL ([19];[35];[36]) or studies focus on problem of assessing actual work submitted by student groups and so on [37].

Little or no research has been uncovered that discusses the issue of monitoring group-work outside classrooms, whether face-to-face or online. Violet and Mansfield [38] suggest that students may be inclined to divide the work among them and simply assemble everything and submit as one work. Other studies suggest that CDL or OCDL cannot be guided by instructors [39] and therefore makes both instructors and students apprehensive.

Therefore, this paper presents a case of successful implementation of weblogs as tool to tracking student contribution in group-work and reducing group conflicts, ultimately enhancing student learning and improving student group performance.

5 Background: Case Study

University of Wollongong in Dubai (UOWD) is an off-shore campus of University of Wollongong, Australia (UOWA). It is also accredited by the Ministry of Higher Education (MOE) in United Arab Emirates (UAE) where it is located and adheres to standards set both by UOWA and MOE in terms of subject learning objectives, content and assessment.

As part of the MOE accreditation, the university has to offer a variety of freshmen courses such as University Life, Philosophy, Islamic Culture and so on. One such subject is Computer Application which teaches students basic computer applications and software to aid them in whatever degree program they decide to pursue.

Part of the learning objective of the subject expects students to complete a semester-long group project on the evolution of ICT in a particular industry and its impact on that industry eg. Airline or Medicine. For this assessment, students have five modules to complete:

Module 1: a contract they sign as a group indicating their roles in the group,

Module 2: introduction and background

Module 3: current impact of ICT on the chosen industry with a case study

Module 4: future of ICT in the industry

Module 5: a campaign ad created using one of the software taught in the class to promote the ICT used in the case.

The students are also expected to prepare a 10-minute presentation based on the report they submit. This assessment weighs 30% of the subject's internal grades.

6 Statement of problem

The subject has been on offer since 2004 and the major problem faced by the students and instructor for the subject has been the group dynamics and complaints regarding the group project.

Based on student evaluations conducted by the Office of Planning and Performance between 2008 - 2010, the identified complaints from student feedbacks include [40]:

1. Instructor has difficulty tracking and monitoring group's progress unless there is a face-to-face meeting. With large class sizes (usually ranging from 150 – 350), it is almost impossible to do so with each and every group, every semester.
2. Some students become unreachable and show up only for the final report submission or presentation. Other group members feel compelled to then help them out. This leads to unfair grading of student contribution.
3. Group conflicts arise as members meet and track decisions verbally or in absentia which are then opposed by other members

Due to these identified issues, the experience of the group project has been described by students as "overwhelming", "not useful", "need more intervention from tutor" [40]. The grades for the projects are regularly "disputed" by students or disgruntled group members [40] and the overall grades of the group project have been consistently low at an average of 65% - 78% [41].

The author decided to implement weblogs in spring semester of 2012 in order to try to monitor the groups and test if the four issues identified above could be addressed and the statement of the problem proposed was: ***Can innovative OCDL tools enhance group work experience among students at UOWD and increase overall group performance?***

7 Method

The author followed the following steps to include weblogs and track the groups:

Step 1: Divided the two hours allocated for computer lab to Lab Hour and Project Hour. Lab hour used to learn new software and practice lab exercise sheets. Project Hour to come together as a group and work on the group project.

Step 2. Chose an easy, free-to-use, website that allows users to develop a free weblog and maintain it.

Step 3. Included a one lab tutorial on how to set up a weblog

Step 4: Included weblog as part of the assessment. Blogging was graded at 5% of the internal grade (making up the group assessment mark of 30%)

In Week 2 of a 13-week semester, students were expected to set up a weblog account in the chosen site after they chose the group members. From the third week onwards, students were encouraged to start posting entries in their respective blogs after each group meeting they had. These entries were then monitored and graded by tutors in the following week's project hour.

Two semesters were chosen to test the method: Spring 2012 and Spring 2013. At the end of each semester, author gathered student feedback using a paper-based questionnaire which was developed to maintain absolute anonymity and conducted by a third-party research assistant to remove any bias towards author as instructor of course. The first section of the survey collected anonymous demographic data. The second section collected data on the frequency of usage of the system, locations used and such. The third section used a five-point Likert Scale to record data on student attitude towards blogging, its usage and effectiveness in managing groups. The last section ended with student recommendations in qualitative format.

8 Results and Discussion

The weblogs allowed the author and lab tutors track and monitor the groups, their progress and contributions on a weekly basis. As the frequency of monitoring increased, the average performance of the group increased as illustrated in Figure 1 below.

As the Figure 1 shows, the average group project mark obtained by groups in each semester ranged from the 60s- to 70s prior to the introduction of blogs.

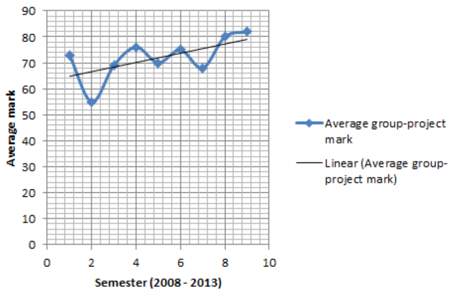


Figure 1: Average group project mark per semester

The two semesters in question, Spring 2012 and Spring 2013 showed a clear spike in the average grade of the class which was well above 80% for the class.

In regards to the issue of ***Instructor difficulty tracking and monitoring group's progress***, on an average, each student posted at least one entry to their respective blogs each week for 11 weeks, and read and added comments to group members' blogs as well with a total of over 3000 entries over the two semesters. However, as these posts were checked during project hour within the lab setting and in front of the group members, the author and tutors had enough time to read and grade the blog entries, clarify any doubts, monitor progress of the groups and question groups on specific issues mentioned in their entries within a time frame that reduced the administrative load of the instructor/tutor.

In regards to the issue of ***Unfair grading of student contribution***, the weekly reporting of meetings, student attendance in the meetings and individual contributions were now recorded and viewed by the instructor/tutor and eliminated the issue of tracking student contribution and unfair grading.

In regards the last issue of ***Group conflicts***, the entries posted every week clearly demonstrated what each student thought was discussed and decided in each meeting. Students viewed each other's entries and corrected any misunderstandings during the project hour.

Furthermore, the student feedbacks on their experience of using the blog revealed the following results:

For the question: How often did you access your blog, over 50% of the students responded that they accessed their blogs once a week and sometimes twice a week. By Spring 2013, over 30% of the students were accessing their blogs daily.

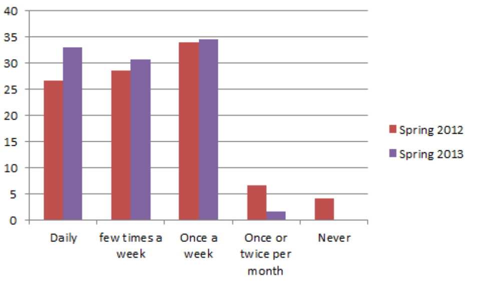


Figure 2: 'How often did you access your blog?'

For the question: From where did you access your blog, students responded a variety of access points, from their own home to university facilities to mobile devices etc as illustrated in the figure 3 below:

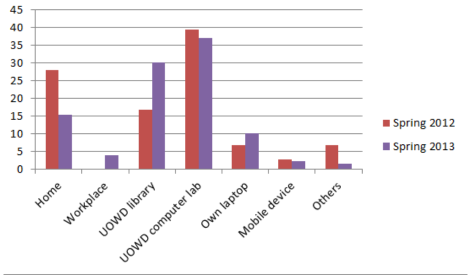


Figure 3: 'From where did you access your blog?'

For the questions on Getting Started, the students responded they were "reasonably comfortable" with blogging as illustrated in Figure 4 below:

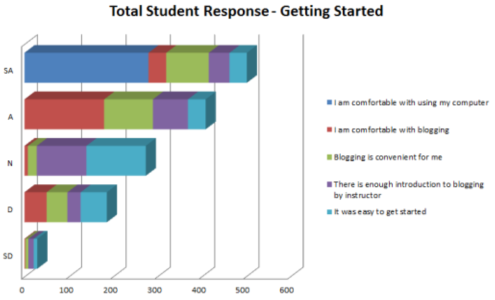


Figure 4: 'Getting Started'

Finally, to the questions pertaining to blogging and groups, majority of the students' responses varied between agree and strongly agree when they were asked if blogging helped reduce conflicts, increased communication between group members, increased access to group decisions and tracking the group's progress as illustrated in Figure 5 below:

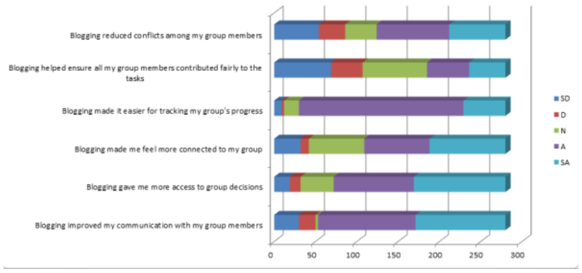


Figure 5: Total student response: Blogging and Groups

9 Conclusion

Understanding learning needs of the digital native students in order to enhance their learning experience has become the driving force behind major shifts in teaching paradigms in the last decade. The process is continuous, ever-growing, ever-evolving [42]. Instructors are trying their utmost to enhance the digital native students' learning through student centered learning styles such as OCDL that incorporates ICT and collaborative discovery learning in order to capture student attention, enhance group environment and ultimately make their graduates more employable.

Literature has suggested many positive effects of OCDL, allowing instructors to achieve substantial results in areas of understanding ICTs, their use in CDL, how, why and when CDLs should be used, how the CDL or OCDL assessments should be graded and used to enhance student learning experience.

This paper proposed some new issues in implementing OCDL and suggested a possible solution to these issues. Based on the group marks and student feedback, tt is believed this study successfully demonstrated the adaptation and implementation of blogging as blended learning tool to track and monitor student groups, increase students' over-all experience working in groups, reduce group conflicts, eliminate unfair grading of student contributions in group work and increase student group performance.

Hence it is concluded that, online collaborative discovery learning, if structured and aided by blended learning and tracking tools such as blogs, can enhance students' overall experience working in a collaborative environment and enhance their overall group performance.

References

[1] Bruner, J. S. (1961). The act of discovery. Harvard Educational Review, 31, 21-32.

[2] Bruner, J. (1967). On Knowing: Essays for the Left Hand. Boston: Harvard University Press.

[3] Dewey, J. (1938). Logic: the theory of inquiry. New York: Holt and Co.

[4] Dewey, J. (1997). Democracy and education. New York: Simon and Schuster. (Original work published 1916)

[5] Piaget, J. (1954). Construction of reality in the child. New York: Basic Books

[6] Vygotsky, L.S. (1978). Mind in Society: The Development of Higher Psychological Processes. Harvard University Press.

[7] Banerjee, R. K. (2012). The origins of collaborative learning. Bright Hub Education. [Online] Available URL: http://www.brighthubeducation.com/teaching-

methods-tips/69716-origins-and-brief-history-of-collaborative-learning-theories/

[8] Johnson, D.W., & Johnson, R.T. (1989). Cooperation and competition: Theory and research. Edina, MN: Interaction Book Company.

[9] Borthick, A. F. and Jones, D. R. (2000). The motivation for collaborative discovery learning online and its application in an information systems assurance course. Issues in Accounting Education. Vol 15. No. 2. Pp 181 – 210.

[10] Stevenson, D. (1997). The Independent ICT in Schools Commission Information and Communications Technology in UK Schools: An Independent Inquiry. Impact noted in Kelly, J. (2000). What the Web is Doing for Schools. Financial Times. September 13 2000. [Online] Available URL: http://specials.ft.com/lifeonthenet/FT3NXTH03DC.html

[11] Khan, Z. R. (2012) Technology boost among University students. Computer Technology Application Journal. Vol 3. No. 4 David Publishing Company. USA

[12] Prensky, M. (2001). Digital Natives, Digital Immigrants. On the Horison. MCB University Press. Vol. 9. No. 5.

[13] Palfrey, J., & Gasser, U. (2008). Born digital: Understanding the first generation of digital natives. New York: Basic Books.

[14] Tapscott, D. (2008). Grown up digital: How the Net generation is changing your world. New York: McGraw-Hill.

[15] Ras, E. and Rech, J. (2009). Using Wikis to support the net generation in improvising knowledge acquisition in capstone projects. The Journal of Systems and Software. Vol. 82. Pp 553-562

[16] Goff, H. (2007). Too much technology in the classroom? BBC News. [Online] Available URL: http://news.bbc.co.uk/1/hi/education/6241517.stm

[17] Schumacher, D. (2009). Medium versus Method: A learner-centered approach to blended learning. DHU TESOL Review. Pp159 – 179. [Online] Available URL: http://www.tesolreview.org/down/2-9.%20Dennis%20Schumacher.pdf

[18] Colis, B., and Moonen, J. (2001). Flexible learning in a digital world: Experiences and expectations. London: Kogan-Page

[19] Witney, D. & Smallbone, T. (2011): Wiki work: can using wikis enhance student collaboration for group assignment tasks?, Innovations in Education and Teaching International, 48:1, 101-110

[20] Brown, A.L. (1997). Transforming schools into communities of thinking and learning about serious matters. American Psychologist, 52(4), 399–413.

[21] McConnell, D. (2005). Examining the dynamics of networked e-learning groups and communities. Studies in Higher Education, 30(1), 25–42.

[22] Piaget, J. (1928). The judgment and reasoning in children. London: Routledge and Kegan.

[23] Higher Education Funding Council for England (HEFCE). (2002). Performance indicators in higher education. Working Paper 52. Bristol: Higher Education Funding Council for England.

[24] Dalsgaard, C. (2006) Social Software: E-learning beyond learning management systems. European Journal of

[25] Open, Distance and E-Learning, 2006/II. [Online]. Available URL: http://www.eurodl.org/materials/contrib/2006/Christian_Dalsgaard.htm

[26] Brooks, K., Nichols, C. and Priebe, S. (2004) Remediation, Genre, and Motivation: Key Concepts for Teaching with Weblogs. Into the Blogosphere. Rhetoric, Community, and Culture of Weblogs, [Online]. Available URL: http://blog.lib.umn.edu/blogosphere/remediation_genre.html

[27] Churchill, D. (2009) Educational applications of Web 2.0: Using blogs to support teaching and learning. British Journal of Educational Technology, 40(1), pp. 179-183.

[28] Du, H.S. and Wagner, C. (2005) Learning with Weblogs: An Empirical Investigation. In System Sciences, 2005.Proceedings of the 38th Annual Hawaii International Conference on System Sciences. [Online]. Available URL: http://wagnernet.com/tiki/tiki-index.php?page=Constructivist+Learning+with+Weblogs

[29] Safran, C. (2008) Blogging in higher education programming lectures: an empirical study. Proceedings of the 12th international conference on Entertainment and media in the ubiquitous era, Tampere, Finland: ACM, pp. 131-135. [Online]. Available URL: http://portal.acm.org/citation.cfm?id=1457199.1457228

[30] Browne, T., Hewitt, R., Jenkins, M., & Walker, R. (2008). Survey of technology enhanced learning for higher education in the UK. UCISA (Universities and Colleges Information Systems Association). [Online] Available URL: http://www.ucisa.ac.uk/en/groups/ssg/surveys.aspx

[31] Doolan, M.A. (2006, June). Effective strategies for building a learning community online using a Wiki. Proceedings 1st Annual Blended Learning Conference University of Hertfordshire, 51–63.

[32] Parker, K.R., & Chao, J.T. (2007). Wiki as a teaching tool. Interdisciplinary Journal of Knowledge and Learning Objects, 3, 57–72.

[33] Kozma, R., Anderson, R.E. (2002). Qualitative Case Studies of Innovative Pedagogical Practices Using ICT. Journal of Computer Assisted Learning. Iss. 18. pg. 387-394.

[34] Schmidt, J. and Mayer, F. (2006) Wer nutzt Weblogs für kollaborative Lern- und Wissensprozesse? Ergebnisseder Befragung 'Wie

ich blogge?!' 2005, Bamberg: Universität Bamberg.

[35] Safran, C. (2008) Blogging in higher education programming lectures: an empirical study. Proceedings of the12th international conference on Entertainment and media in the ubiquitous era, Tampere, Finland: ACM, pp. 131-135. [Online], http://portal.acm.org/citation.cfm?id=1457199.1457228 [30 Jan 2009].

[36] Laurillard, D. (2000). Rethinking university teaching: A framework for the effective use of educational technology. London: Routledge

[37] Kirschner, P., & Kreijns, K. (2004). The sociability of computer-mediated collaborative learning environments: Pitfalls of social interaction and how to avoid them. In R. Bromme, F. Hesse, & H. Spada (Eds.), Barriers and bias in computer-mediated knowledge communication–and how they may be overcome (pp. 169–192). Dordrecht, NL: Kluwer.

[38] Violet, S., & Mansfield, C. (2006). Group work at university: Significance of personal goals in the strategies of students with positive and negative appraisals. Higher Education Research & Development, 25(4), 341–356.

[39] Andergassen, M., Behringer, R., Finlay, J., Gorra, A., and Moore, D. (209). "Weblogs in Higher Education –why do Students (not) Blog?" Electronic Journal of e-Learning Volume 7 Issue 3 pp203 -215. [Online] Available URL: www.ejel.org

[40] Office of Planning and Performance (2008- 2010). Subject Evaluation for Computer Application. Confidential semester evaluation summaries available at the Office of Planning and Performance Archives. UOWD.

[41] Subject Report (2008 – 2011). Reports available in subject folders based on final grades (including internal grades and final exam marks) of students produced for Faculty assessment meetings every semester at Faculty of Computer Science and Engineering. UOWD.

[42] Khan, Z. R. (2011). Learning Management Systems and Guided Discovery: innovative learning tools to teaching computer-application to business students. IASTED International Conference on Technology for Education. December 14 – 16 2011. Dallas. USA

A Learning Model of Simple Computing Machine Architecture

Tai-Chi Lee and Hector Zimmermann-Ayala
Department of Computer Science and Information Systems
Saginaw Valley State University
University Center, MI 48710

Abstract

This work emerged from an independent study project, which involved building a simple machine with few registers and limited memory that could execute a program in the machine language of binary code. The purpose of this project is to learn the basic concepts and the fundamental logic design of a processor, the basic machine architecture is outlined. In order to fully understand how an instruction set gets executed, first the instruction format must be well defined and with implemented simplicity where we only limit our instruction types to the minimum to reduce size of the instruction set, which makes it easier to follow when analyzing and synthesizing the functionality of various components in a processor. For example, the arithmetic operations consist only of Addition, Subtraction, Multiplication, and Division. For data movement we have the Load and Store, and for logical operations only Compare is used. For Arithmetic operations only subtraction and addition are implemented. Multiplication and division can be achieved by repeated additions and subtractions respectively. Program flow control uses only the Jump. Furthermore, for memory management, we added a one-byte Cache. This project provides students with an easy learning experience, which is needed for their study at the introductory level of the computer architecture course, while getting a better understanding of how a processor works.

1. Introduction

Over the past five decades the computer industry has experienced enormous changes in the architectural design. Increases in device speed and reliability, as well as reduction in hardware cost and physical size have greatly enhanced computer performance. While major components such as central processing units, memories, input and output units which perform basic computations remain unchanged, issues of using cache, pre-fetch techniques, number of cores in a single chip, and choice of Reduced Instruction Set Computer (RISC) or Complex Instruction Set Computer (CISC) CPUs [1, 3] are debatable. Also, according to studies in the field, CISC CPUs do not seem to have as big an advantage as some thought they would have. So one variant of RISC, Small Instruction Set Computer (SISC) [6], being implemented in this project is introduced for the purpose of study as it seems to have less overhead for a performance advantage over CISC Processors.

2. Constructions and Designs of Basic Components

To orient the student researchers to our SISC CPU [7] for performance and simulation reasons for an Embedded System the following CPU Simulator was developed:

1) Registers

This SISC CPU Emulator was designed with four byte size registers R0, R1, R2, R3, and R0 is the Accumulator with the ability to have some operations in internal Level 1 Cache Memory with the R1 register.

2) Main Memory

This SISC CPU Emulator operates with an external memory of 256 bytes as it is designed for embedded systems.

3) Cache Memory

This SISC CPU Emulator pre-fetches a byte in Level 2 Cache Memory to increase performance in case the processor requires it as data.

4) Buses

This SISC CPU Emulator uses the data and memory buses in its design.

5) Other peripherals

Any peripherals that would be added to the system can have interfaces through the Memory Addresses available to the CPU.

3. Instruction Set

A Small Instruction Set is provided with support for the Memory, Cache Memory, and Buses, as well as the ALU (Arithmetic Logic Unit).

3.1. Instruction Set Description

This SISC CPU has the following Instruction Set:

Mnemonic	Instruction	Length	Description.
ADD01	00000001	1 Byte	Add registers R0 and R1 and stores the result on R0 (The Accumulator). This operation also affects the Zero Flag (Whether the result is Zero) and the Overflow Flag (Whether the operation exceeded a Byte or not).
ADD	00000010	2 Bytes	Add the contents of the Accumulator (Register R0) and the data byte previously Cached, storing the result on the Accumulator (Register R0). This operation also affects the Zero Flag (whether the result is Zero) and the Overflow Flag (whether the operation exceeded a Byte or not).
ADD1	00000011	2 Bytes	Similar to ADD, but for Register 1 (R1).
ADD2	00000100	2 Bytes	Similar to ADD, but for Register 2 (R2).
ADD3	00000101	2 Bytes	Similar to ADD, but for Register 3 (R3).
SUB01	00000110	1 Byte	Subtract R1 from R0 and store the result into the Accumulator (R0). The Overflow Flag is affected to reflect whether the result exceeded one Byte or not and the Zero Flag is also affected to reflect whether the result is Zero or not.
SUB	00000111	2 Bytes	Subtracts the Cached Data from the Accumulator and places the result in the Accumulator (R0.) The Overflow Flag is affected to reflect whether the result exceeded one Byte or not and the Zero Flag is also affected to reflect whether the result is Zero or not.
SUB1	00001000	2 Bytes	Similar to Sub, but for R1.
SUB2	00001001	2 Bytes	Similar to Sub, but for R2.
SUB3	00001010	2 Bytes	Similar to Sub, but for R3.
LOAD	00001011	2 Bytes	Loads the Memory Contents with Memory Address in Cache into the Accumulator.
LOAD1	00001100	2 Bytes	Similar to LOAD, but for R1.
LOAD2	00001101	2 Bytes	Similar to LOAD, but for R2.
LOAD3	00001110	2 Bytes	Similar to LOAD, but for R3.
SEC	00001111	1 Byte	Sets the Carry Flag.
CLC	00010000	1 Byte	Clears the Carry Flag.
SEI	00010001	1 Byte	Sets the Interrupt Flag.
CLI	00010010	1 Byte	Clears the Interrupt Flag.
JMP	00010011	2 Bytes	Unconditional Jump to the address pre-fetched on the Cache.
JNZ	00010100	2 Bytes	Conditional Jump to the address pre-fetched on the Cache. Jumps if the Zero Flag is NOT set (Jump if Not Zero).
JIZ	00010101	2 Bytes	Conditional Jump to the address pre-fetched on the Cache. Jumps if the Zero Flag IS set (Jump If Zero).
JCS	00010110	2 Bytes	Conditional Jump to the address pre-fetched on the Cache. Jumps if the Carry Flag IS set (Jump if Carry Set).
JCC	00010111	2 Bytes	Conditional Jump to the address pre-fetched on the Cache. Jumps if the Carry Flag is Clear (Jump if Carry Clear).
STORE	00011000	2 Bytes	Stores the Accumulator (R0) contents into the Memory Address pre-fetched in Cache.
STORE1	00011001	2 Bytes	Similar to STORE, but for R1.

STORE2	00011010	2 Bytes	Similar to STORE, but for R2.
STORE3	00011011	2 Bytes	Similar to STORE, but for R3.
RET	11111111	1 Byte	Return / End.

3.2 Configuration of Components

The learning model presented in this paper consists of four major components including Processing Unit , Input Unit, Output Unit, and Secondary Storage (external memory), where Processing Unit is composed of Central Processing Unit (CPU) and Main Memory (internal memory). And, CPU is further divided into Arithmetic/Logic Unit (ALU) and Control Unit or Master Control Unit (CU/MCU). Figure 1 shows the configuration of the components.

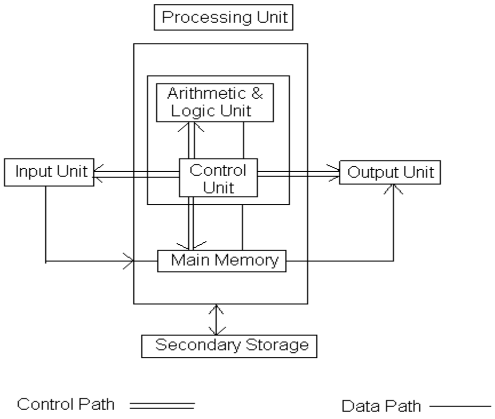


Figure 1

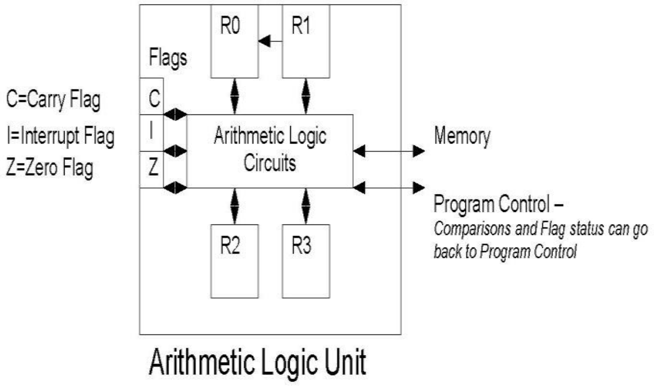


Figure 2

The following describes the role of each component.

1) ALU contains a number of registers, which performs all the arithmetic and logical operations.

2) CU or MCU is central controller, which commands the sequence of operations for all the components.

3) Main Memory is the primary storage for storing programs or data that are currently being used by the computer. It comes in two forms of memory, Random Access Memory (RAM) and Read Only Memory (ROM) [7].

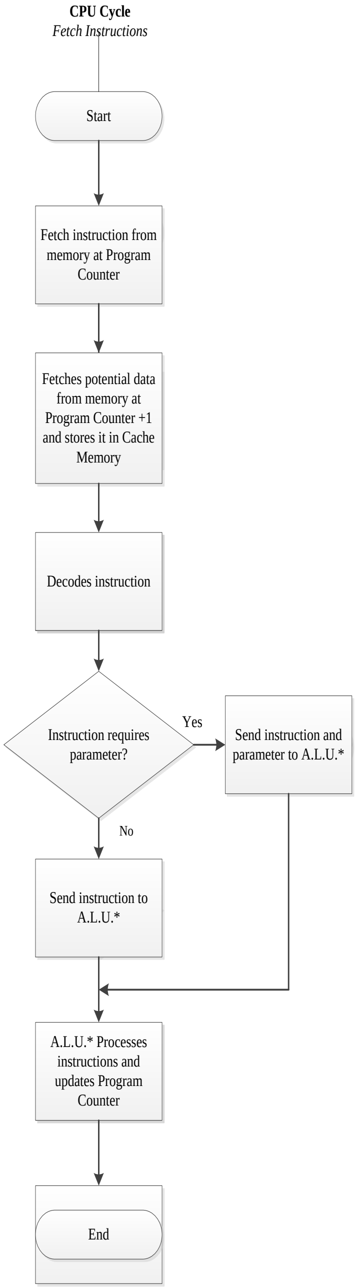
4) Input Unit is a device that allows users to enter data into the main memory. Examples are keyboard, mouse, etc…

5) Output Unit is a device that allows users to display the results from processing. Examples are printer, monitors, etc…

6) Secondary Storage is for storing and retrieving data and program. Examples are hard disk drive, CD, USB, tape, etc.…

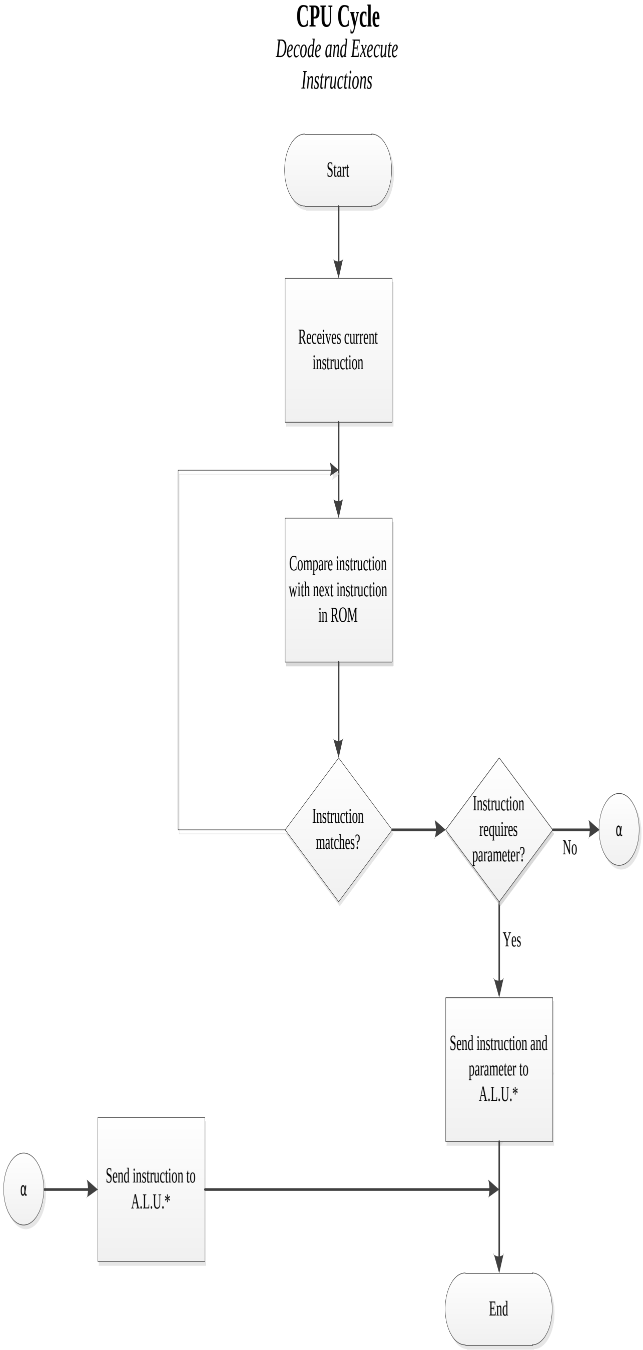
7) Bus is a path by which the electronic signal (control path) or data (data path) travels from one place to another.

In our designed model CPU will have a list of the custom instruction set it supports. When the CPU is started, it will read this list under the PC Master Control. The PC Master Control then loads the program through the Control Instruction LOADPRG, and executes a Memory Dump on screen. Afterwards, it transfers control to the PARSE instruction, which then executes a Register Dump on Screen and reads the first two bytes into Cache [7]. The Master Control scans for instructions to match the required operation and, if found, will determine whether it needs the Cached data byte or not. If it does, it will execute the instruction and send the Cached data byte as a parameter. It will then update the Program Counter accordingly and execute the instruction. After executing the instruction, it will then output a disassembly of the instruction on Screen. At the end of the run, it will perform a last Memory Dump on Screen. The following Figures 2 and 3 illustrate the flows of executions followed by a test run with results shown in Section 3.3.


CPU Cycle
Fetch Instructions
Start
Fetch instruction from memory at Program Counter
Fetches potential data from memory at Program Counter +1 and stores it in Cache Memory
Decodes instruction
Instruction requires parameter?
Yes
Send instruction and parameter to A.L.U.*
No
Send instruction to A.L.U.*
A.L.U.* Processes instructions and updates Program Counter
End

*Arithmetic Logic Unit

Figure 3


CPU Cycle
Decode and Execute Instructions
Start
Receives current instruction
Compare instruction with next instruction in ROM
Instruction matches?
Instruction requires parameter?
No
α
Yes
Send instruction and parameter to A.L.U.*
α
Send instruction to A.L.U.*
End

*Arithmetic Logic Unit

Figure 4

3.3 Test Runs and Results

The following sample program was written to test out the SISC CPU. It is expressed in decimal number system and interpreted in binary, as well as disassembled by the program for a better appreciation. Also, it provides memory dumps after loading and at the end of execution.

```
Input program file: 11 21 12 22 13 23 14 24 1 25 22 4 15 26 22 16 15 23 25 255 10 10 5 5 24 23 255

Program Loaded Ok

*** Memory dump begin ***
  1  11  21    12  22 13 23
  7  14  24     1  25 22   4
  13 15  26    22  16 15 23
  19 25 255   10  10   5   5
  25 24   23 255  0     0   0
*** End of memory dump ***

Running Program...

R0:0 R1:0 R2:0 R3:0          ←Initial contents in R0, R1, R2, and R3 are all 0.
1 LOAD 21                    ← Memory Location 21 contains 10, which gets loaded  into R0
R0:10 R1:0 R2:0 R3:0
3 LOAD1 22                   ← Memory Location 22 also has 10, which will be loaded into R1
R0:10 R1:10 R2:0 R3:0
5 LOAD2 23                   ← Memory Location 23 has 5, which will be loaded into R2
R0:10 R1:10 R2:5 R3:0
7 LOAD3 24                   ← Memory Location 24 has 5, which will be loaded into R3
R0:10 R1:10 R2:5 R3:5
9 ADD01                      ← This will add R0 and R1 and will store the result into R0
R0:20 R1:10 R2:5 R3:5
10 STORE1 22                 ←This will store the contents of R1 into Memory Location 22
R0:20 R1:10 R2:5 R3:5
12 ADD2 15                   ← This will add 15 to R2 and store it into R2
R0:20 R1:10 R2:20 R3:5
14 STORE2 22                 ← This will store the contents of R2 into Memory Location  22
R0:20 R1:10 R2:20 R3:5
16 CLC                       ←This clears the Carry Flag
R0:20 R1:10 R2:20 R3:5
17 SEC                       ← This sets the Carry Flag
R0:20 R1:10 R2:20 R3:5
18 JCC 25                    ← If Carry Flag was clear it would jump to a STORE, which proves that the conditional
20 RET                         jump JCC is operating fine on opposite condition.

*** Memory dump begin ***

  1  11   21   12  22  13  23
  7  14   24    1  25  22    4
  13 15   26   22  16  15  23
  19 25 255   10  20    5    5
  25 24   23 255    0    0    0
  *** End of memory dump ***
```

Similarly, by changing the source program to have test runs on the remaining instruction types not contained in the above test program, we found that they all work correctly. This assures that our designed model works as they are designed for. Note that due to page limitation, the remaining test run results are not shown, but they are available upon request.

4. Conclusion

In this simple model of machine architecture, we show the basic components of a computer and the constructions of each element. For learning purposes, we keep it simple enough for beginners to follow easily. Once they have the concepts and a good understanding of the basic operations, they can carry on to expand the functionalities necessary to handle the more complex computations. Furthermore, they can also customize the design to meet the needs of individual applications. If design couples with Field Programmable Gate Array (FPGA) co-design techniques [4, 5], it may become a viable means by which to perform complex calculations quicker than general purpose processors. Our approach and design make use of the fundamental concepts of digital logic. If custom-built instructions are used, we expect the customized system to out scale similar single- purpose computing platforms due to the use of a smaller instruction set and embedded processors. The model presented here is intended to be used for an introductory course of computer architecture. More complex features such as shared memory multiprocessors [6], parallel and pipelining processing [2, 3] can be added when the level of students advances.

5. Acknowledgement

We are grateful for the support of the SVSU professional development grant for the work of this paper and the student author would like to thank Dr. Lee for providing the resources and encouragements. Also, our gratitude goes to Ann Garcia for her efforts in preparing this manuscript.

References

[1] John P. Hayes, *Computer Architecture and Organization*, 1988, McGraw-Hill, Inc.

[2] Kai Hwang and Faye A. Briggs, *Computer Architecture and Parallel Processing,* 1984, McGraw-Hill, Inc.

[3] Richard Y. Kain, *Advanced Computer Architecture: A Systems Design Approach*, 1996, Prentice- Hall, Inc

[4] Tai-Chi Lee and Patrick Robinson, *A FPGA-Based Designed for an Image Compressor*, International Journal of Pure and Applied Math, Academic Publications, Volume 33 No.1 2006, pp 63-67.

[5] Tai-Chi Lee, Mark White, Michael Gubody, Allison Nicol, Christpher Plachta, Jeremy Strawn, and Cori Thompson, *Building An FPGA-Based Computing Platform,* The Proceedings of The 2012 International Conference on Frontiers in Education: Computer Science & Computer Engineering, pp 522-527, July 16-19, 2012, Las Vegas, NV.

[6] Julia Lobur and Linda Null, *The Essentials of Computer Organization and Architecture,* 3rd Edition, 2012, Jones & Bartlett Learning, LLC.

[7] William Stallings, *Computer Organization and Architecture: Designing for Performance,* 9th edition, 2013, Person Education., Publishing as Prentice Hall.

Learning computer programming with a COAC#.

Marcos A. Jakeline
Computer Science Department
Tecnológico de Monterrey (ITESM)
Monterrey, N.L. México

Abstract - *The purpose of this work is to present a visual tool to support first-year programming courses. This interactive tool could be used by professors in class, or by students anytime, anywhere. It was formally developed and tested, and we found that a key to make this tool successful was its user interface; therefore special attention was set on the development of a good user interface. This tool has successfully used in the last three terms, starting with my own programming courses and now being used for at least other three lectures. The first experimental results show that it's a very effective tool, allowing student to improve their programming course grade.*

Keywords: computer languages, C# programming, C++ programming, programming fundamentals, CS1.

1. Introduction.

With the rapid evolution of technology in recent years, education has been transformed from mainly a teaching discipline to a technology enabled learning environment [1]. The old scenario where the teacher used to give his/her lecture and students passively listened, trying to understand the teacher's explanation, is not enough anymore. The goal is active exploration, construction, and learning rather than the passivity of lecture attendance and textbook reading [2].

Along with technology, the strategies, techniques, and tools related with teaching have evolved through time. Especially, teaching programming languages has always been a challenge, due to the abstract nature of those languages and developing of the student's logical thinking.

The challenge involved in teaching programming languages is an issue and concern of many professors who belong to the Computer Science field. As a matter of fact, according to a survey conducted among 350 professors from different universities in the U.S, where they were asked which one is the hardest topic for beginner students of Programming Fundamentals (CS1), their answers were according to the next four categories [3]:

1. Problem solving and design,
2. Control structures, parameters and recursion,
3. Object-oriented: inheritance, polymorphism, methods, classes and
4. Students' lack of maturity and good studying habits.

If we focus on the second category, teaching control structures such as loops, or loops with arrays and matrices, it is definitely not an easy task. In fact, when developing programming skills a determinant factor is the mastering of control structures, specifically those of repetition. [4].

On the other hand, the use of simulators to teach Computer Architecture and Organization is a common practice. In this regard, a lot of research has been done about the use of simulators to teach basic organization and computer architecture concepts. Nevertheless, very little research is available on the development and use of tools to teach how to build basic programs [5].

2. Programming Fundamentals.

One of the most powerful concepts in any programming language is repetition or loops. This is one of the elements that allow a program to do much more than just performing a simple calculus [6].

Before proceeding and to provide the reader with some context, a brief review on the concept of loops will be offered:

The word ***while*** indicates the repetition of instructions between braces. The condition that is shown between parentheses immediately after the word *while* controls the loop [7]. The format would be the following:

```
while (condition)
{
    instructions
}
```

Many of those components from the *while* statement are also included in the **for** loop, grouped in the instruction's heading:

```
for(int i = 1; i < n; i++)
```

Between parentheses in the *for* statement, there are three elements divided by semicolons:

- an initial instruction: will be performed only once, before the loop begins.

- a condition: is evaluated before running the loop anytime.
- a final instruction: to be run just before the end of each loop.

When the instructions *while* or *for* are used, the evaluation will always take place at the beginning of the repetition. The **do-while** loop is an alternative structure, in which the evaluation takes places at the end of each repetition [7].

```
do {
    instructions
}while (condition);
```

It is important to point out that the use of these control structures represent certain "difficulties" for students who, for the first time, are learning a programming language. For CS1 courses, simple control structures are still a problem even with an object- oriented language [3].

Some common errors in programming are:

- When writing loops, particular attention must be pay to specifying the condition. It is a very common error to make a loop finish one repetition too early, or else repeat once to many times. This is sometimes called an "off by one" error [7].
- Unless you have a mechanism for escaping from a loop, it is important to make sure it doesn't run forever. This is the case in which the execution can never end [6].
- A common error is to confuse the length of an array or string with the range of valid indices. This is the case when the program fails with an "IndexOutOfRangeException" message [7].

3. Developing a tool called COAC#

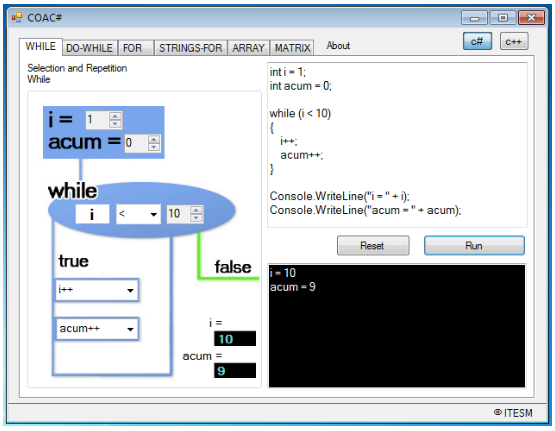
Programming involves several dynamic concepts that many times are taught through static means (projected presentations, verbal explanations, diagrams, blackboard drawings, texts, and so on). For some students this is a problem, as they fail to understand program dynamics through this type of materials [8].

Furthermore, for a beginner student it can be overwhelming to learn the right syntax, *and* at the same time the logic or functioning of control structures. For this reason, and with the purpose of helping students through their learning process of programming languages, in the fall of 2011 work was begun towards the creation of an application prototype that could simulate the execution of a code's fragment that could be manipulated by the user.

In the beginning this application was developed for C# language -where COAC# gets the name from- and later, the C++ version was added.

The application consists in simulating the execution of a code fragment, dividing the screen in three parts (see fig.1):

- The code written in programming language, whether it is C# or C++.
- The results of the code execution, displayed in a textbox that simulates a console.
- The animated flow diagram chart, which makes it easy for the user to choose or set certain values for the variables and certain conditions for the loop.



```
int i = 1;
int acum = 0;

while (i < 10)
{
   i++;
   acum++;
}

Console.WriteLine("i = " + i);
Console.WriteLine("acum = " + acum);
```

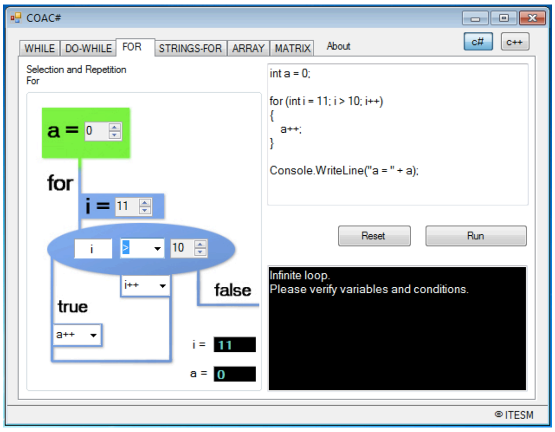
```
i = 10
acum = 9
```

Fig. 1 Example of the Application COAC#

4. COAC# advantages.

Some advantages of this application are:

- Better visualization of the repetition statutes, by using an animated simulation showing the execution flow inside the loop.
- Better understanding of what happens in memory, by providing a simulation that shows how the variables values are changing.
- Possibility of trying different data for the variables and conditions for the loops, so that the user is able to analyze and visualize "what would happened if..."
- Language choice, whether it is C++ or C#.
- Stand-alone tool, not requiring Internet connection.
- Immediate feedback of errors such as infinite loops, index out of bounds, etc. (see fig. 2)



```
int a = 0;

for (int i = 11; i > 10; i++)
{
   a++;
}

Console.WriteLine("a = " + a);
```

```
Infinite loop.
Please verify variables and conditions.
```

Fig. 2 Error description for an infinite loop.

5. Experience evaluation.

Any professor who teaches a beginner's course of programming languages will understand that explaining the concept of loops or working with strings and arrays, is not an easy task. It could be that some will try drawing, or some others would prefer daily life analogies, but at the end, they all try to explain this topic in the best possible way, for the students' better understanding.

In the end, the exact purpose of applications like COAC#, is to be used by the teacher in the classroom, when giving the lecture; or by the student, once he is studying on his own.

Along with the COAC# application, a practice was designed (see Appendix A). The objective of this practice is to guide the student in the use of the application, so that he will think on the different cases that are stated and achieve a broader comprehension of the analyzed concepts.

6. COAC# by the numbers

During Fall-2012 term, COAC# was used in the classroom, specifically in 3 sessions where loops were explained. Next, students were asked to complete a guided practice (see Appendix A). Sample students were from two different CS1 sections with the same teacher, summing up to 57 students in total. Later, in a written test, a problem related to loops with strings was included.

Results of the performance analysis of students who used the COAC# tool versus those who did not, are the following:

From those 57 students, only 22 handed in the assignment and 35 did not. Those students, who *did* turn in the assignment, achieved an average of 18.43 points out of 25 in the test's loop problem. As for the students who did not write the guided practice, they obtained an average of 17.18 points out of 25.

Besides, it is known that 4 students who *did not* hand in the practice, and 4 students who *did*, already had basic programming knowledge before taking the course. Setting these aside, the average was recalculated, and it was noted that those students who *did* turn in the practice achieved a higher number of points (17.58) than those who *did not* solve the guided practice with COAC# (15.72).

	Performance
No COAC#	62.88
COAC#	70.32

Table 1. Performance results

It should be considered that, involved in this analysis, there are some factors that cannot be controlled, such as the fact that some students who turned in the guided practice could have been dishonest and may have copied the results from their classmates in order to obtain the assignment's grade. Or, those who *did not* hand in the assignment, not because they did not do it at all, but only because they did not completely finish.

According to data, the correlation coefficient is showing that whether the guided practice was done or not, does not result in a significant impact on the test's grade. Nevertheless, looking at the averages, some improvement is clearly identified among students who used the COAC# application. From a qualitative approach, it could be said that students seemed to like the new tool and that it does help them understand the new concepts, since students were seen using the application in class even when not asked to do so.

During the Fall-2012 term, the application was presented in class and students appeared to be very interested and eager to use it. When a second test's grades were given to the students, a girl who obtained 68/100 in the previous test, was happy to achieve a 95/100 grade and told her classmates: "...I really found the COAC# very useful since I couldn't join my study team and had to do it on my own.”

5. Conclusions and future work.

It would be hard to declare the COAC# application can guarantee itself the students' full understanding of the topic, but it could definitely be recognized as an innovative tool, easy to use, accessible for students and teachers. It also does help understand certain topics that are very hard to explain sometimes and a little too abstract to be understood immediately.

The proper and creative use of IT and communication in the teaching and learning process can become a key factor in any course. The use of new tools such as COAC# can make a big difference in the learning process because it offers the possibility of being used not only by the professor in the classroom, but also by the student, anywhere, anytime.

The last application's updates already includes modules about Arrays and Matrices, in both C# and C++ languages.

References.

[1] Yadin, A. (2011). Reducing the dropout rate in an introductory programming course. *ACM Inroads, 2* (4).

[2] D. A. Norman and J. C. Spohrer (1996), Learner-centered education. *Communications of the ACM, 39* (4).

[3] Dale, N. (2006). Most difficult topics in CS1: results of an online survey of educators. *SIGCSE Bulletin, 38* (2).

[4] Milne, I., and Rowe, G. (2002). Difficulties in Learning and Teaching Programming – Views of Students and Tutors. *Education and Information Technologies, 7.*

[5] Stone, J. (2006). Using a machine language simulator to teach CS1 concepts. *SIGCSE Bulletin, 38* (4).

[6] Overland, B. (2011). *C++ Without Fear: A Beginner's Guide That Makes You Feel Smart*. Second Edition. Prentice Hall.

[7] Bell & Parr (2010). *C# for Students*. Revised edition. Pearson Education.

[8] Gomes, A., Mendes, A.J. (2007). Learning to Program – Difficulties and Solutions. Proceedings of the International Conference on Engineering Education. Coimbra, Portugal.

Appendix A: (Example of an Assignment using COAC#)

Programming Fundamentals

Guided Practice using COAC# application

Topics: While, DO-While, For, Strings

TECNOLÓGICO DE MONTERREY®

Name: ____Student id: ________________

WHILE

1. Run the application **COAC#** using WHILE and notice the results with each of the following inputs. Write the results and if applicable, any conclusions.

a. Initialize acum = -5; i = 0; Loop condition: i < 10

b. Initialize acum = 0; i = 9; Loop condition: i < 10

c. Initialize acum = 0; i = 10; Loop condition: i < 10

d. Initialize acum = 0; i = 5; Loop condition: i > 10

e. Initialize acum = 0; i = 15; Loop condition: i > 10 increment: i++

f. Initialize acum = 0; i = 15; Loop condition: i > 10 decrement: i--

DO-WHILE

2. Run the application **COAC#** using *DO-WHILE* and notice the results with each of the following inputs. Write the results and if applicable, any conclusions.

a. Initializing acum = 0; i = 0; While: i < 0
Notice the loop runs at least once, even if the condition is not met.

b. Initializing acum = 0; i = 0; While: i < 2; ¿what happens?

c. Initializing acum = 0; i = 3; While: i >= 2; con i++; ¿what happens? ¿why?

d. Initializing acum = 0; i = 3; While: i >= 2; con i--; ¿what happens? ¿why?

FOR

3. Run the application **COAC#** using FOR and notice the results with each of the following inputs. Write the results and if applicable, any conclusions.

a. Initializing a = 0; i = 0; While: i < 0; i++;
Notice that if the condition is not met, the loop is NEVER run.

b. Initializing a = 0; i = 0; While: i < 5; i++; ¿How many times is the loop executed?

c. Initializing a = 0; i = 0; While: i <= 5; i++; ¿How many times is the loop executed?

d. Initializing a = 0; i = 10; While: i >= 5; i++; ¿what happens? ¿why?

e. Initializing a = 0; i = 10; While: i >= 5; i--; ¿what happens? ¿why?

STRINGS

4. Run the application **COAC#** using STRINGS and notice the results with each of the following inputs. Write the results and if applicable, any conclusions.

Once inside COAC#, write your name under the "name" variable and then:

a. Try the for loop starting with int i = 0; while i <= name.Length-1; i++ ¿what happens?

b. Try the for loop starting with int i = 0; while i <= name.Length; i++ ¿what happens?

c. Try the for loop starting with int i = name.Length; while i >= 0; i-- ¿what happens?

d. Try the for loop starting with int i = name.Length-1; while i >= 0; i-- ¿what happens?

e. Try the for loop starting with int i = 0; while i < name.Length; i+=2 ¿what happens?

Implementing a Virtual Teaching and Learning Environment: a model proposed by UNNOBA (National University of the North West of the Province of Buenos Aires, Argentina)

C. Russo[1], **M. Sarobe**[1], **H. Ramón**[1], **N. Alonso**[1], **L. Esnaola**[1], **P. Lencina**[1] **and T. Ahmad**[1]
[1] Institute of Research and Technology Transfer (ITT), UNNOBA, Junín, Buenos Aires, Argentina

Abstract - *This document discusses the design, creation and coordination of a virtual teaching and learning environment and the related information systems. At UNNOBA, we advocate for the design and implementation of a model to achieve the potential of IT tools, the geographical and time reach of the Internet and the management systems thus seeking to improve the quality of current teaching and learning methods. This model should consider the continuous improvement of management systems and encourage and facilitate the use of virtual classrooms and the training of human resources to fully benefit from the tools and services available on the environment.*

Keywords: virtual environment, teaching and learning, information systems, administrative process, Moodle.

1 Introduction

The past decade has been witness to unprecedented technological developments: new devices, multiple networks, a noticeable increase in the information available and "hyper-connected" people. The fact that habits, relationships, society, our world and, particularly, the ways of learning have changed cannot be denied.

"*…Technology issues may not be urgent, but we believe they are a need, because they are there: a cell phone held by a teenager who is sending text messages at a speed that leaves adults astonished; a computer at a school in the middle of nowhere; a two-year old child pushing a button on a toy and waiting for an answer.*" [1]

Education cannot be unaware of this fact and, as a result, in the past few decades, ICTs have become a part of the teaching and learning process in what is now known as: e-Learning. [2]

"*…Only a comprehensive e-learning proposal can satisfy the needs of the players involved. Organizing an educational activity, covering different levels and areas of knowledge, and doing it in a continuous manner during the academic year and over a period of time, guaranteeing certain levels of success to the participants, customizing learning to the specific demands of groups or institutions is a real complex task that needs to be undertaken by highly specialized professionals, especially if the activity is carried out in different modalities (face-to-face, on-line, and blended-learning)*". [3]

As stated by Javier García Manzanedo: "*…we are witness to a new social model characterized by rapid changes and transformations that technology development can bring about and education and training should be adapted to this new situation. Following Elliot Massie, it's about bringing learning to people and not, as we have done till now, bringing people to learning*". [4]

The purpose is still the same, people want to learn and they place themselves in the hands of those who can teach them, it's the means that changed.

We currently have new means that give us multiple benefits but, at the same time, present us with new challenges. One of those challenges is how to coordinate them and guarantee that the people who use them, in this case, students, heads of departments, administrative staff etc., can fully benefit from them. With this in mind, we have proposed a comprehensive model for UNNOBA, with the aim of providing teaching and learning experiences with quality guidance.

2 Aspects of the Model

A series of aspects that the model seeks to address with concrete solutions were identified. These aspects are the basis of what we consider should be part of a virtual teaching and learning environment for UNNOBA:

- Administrative aspect: A virtual teaching and learning environment is a system that integrates different solutions but its main objective is to provide adequate support to the teaching and learning activities of the main players: teachers and students alike. They need strong guidance to be provided by people who are highly trained and supported by well-defined processes.

- Educational aspect: Teachers suggest methodologies and work plans but on a dynamic teaching and learning environment, certain information is needed in the available tools. Once again, strong guidance by specialized staff is not only desirable but necessary.

- Technical aspect: Any virtual teaching and learning environment needs to provide technical assistance and guidance to deal with any issue that may compromise the usability of the system. The guidance provided by specialized human resources is desirable and necessary.

- Functional aspect: Necessary tasks such as the management of virtual classrooms: their creation, enrolment process, teacher appointments and role allocation, etc., should be standardized and carried out by specific personnel. On the one hand, because these tasks should not be part of the teacher's responsibilities, the teacher is a user and not an administrator of the virtual environment but on the other hand, because the performance of these tasks needs to be uniform, since a diverse environment could be messy.

- Evolutionary aspect: Any virtual teaching and learning environment should take into consideration a potential evolution if it aims at being effective in changing times such as the present. Control and update process are necessary to adapt the virtual environment to new tools and create a virtuous circle to have it permanently optimized.

3 Suggested Areas

The aspects described above will be addressed by defining certain areas in the model that can act jointly and be supported by repetitive and well-document processes to approach any issues.

The figure below shows the areas identified:

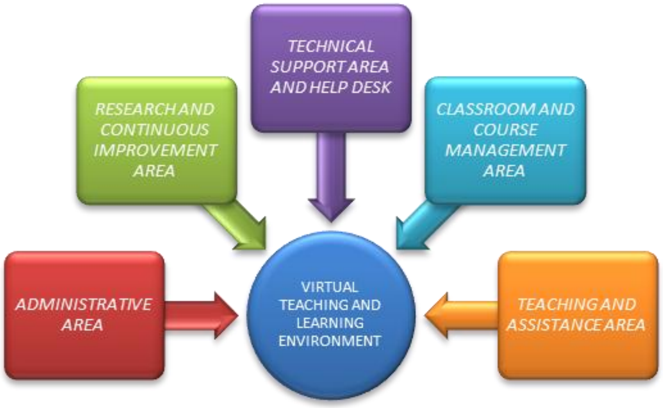

TECHNICAL SUPPORT AREA AND HELP DESK
RESEARCH AND CONTINUOUS IMPROVEMENT AREA
CLASSROOM AND COURSE MANAGEMENT AREA
ADMINISTRATIVE AREA
VIRTUAL TEACHING AND LEARNING ENVIRONMENT
TEACHING AND ASSISTANCE AREA

Figure 1. Areas suggested to supplement UNNOBA's virtual teaching and learning environment.

- Administrative Area: Considering the potential creation of a special area dedicated to distance learning, we are faced with the need of finding resources for it. This new area will need teachers and administrative personnel, and financial and technological resources, among others. Good management of these resources is essential to achieve the aim of the area. This area should liaise closely and work cooperatively with the other areas of the University so that they can address different issues with a global objective.

- Research and Continuous Improvement Area: Teaching entails harmonizing knowledge from different areas, teaching skills and educational materials as well as the use of different tools. Some of these tools can be older than other ones, be more or less tried or simply follow new trends; but what is really important in the use of teaching tools is to be aware of their potential so that teachers can fully benefit from them. They are as vital in face-to-face teaching as they are in distance learning. They entail an ongoing effort by the teachers and a development of their learning communities. But in the end, what they really are is an investment in human resources.

- Technical Support Area and Help Desk: Teachers and students may feel that they are left alone. This is typical of distance learning but it may bring about feelings of frustration in case of technical failures. This area is designed to provide ongoing guidance in the use of the virtual teaching and learning environment to users, teachers and students equally, helping them solve any issue that may arise.

- Classroom and Course Management Area: Implementing a virtual teaching and learning environment means that the people involved need to be knowledgeable about the software but also aware of the teaching and learning aims that teachers seek to achieve through the environment. Therefore, we believe that this area is necessary and it should liaise and interact with all the other areas.

- Teaching and Assistance Area: Some of the nuances of distance learning are related to the reformulation of the teacher's role. Teachers need to rethink their role, recreate it and try to intervene in the students' learning process in a different way. They will need to learn about new technologies and replace their classroom presence with other strategies. Materials should be taken into consideration. Distance learning needs different materials. In some cases, materials created for face-to-face lectures can be adapted but in others, they need to be created from scratch. In addition, the pedagogical capabilities of each of the teaching means should be assessed considering their financial and technological accessibility, level of abstraction, stimulation, suitability to transmit abstract concepts, students skills needed for their use, readability, the passivity and individualism they bring about, the kind of feedback, and the related motivation, etc. Finally, having a tutor becomes essential. The role of the tutor should be to accompany the teachers and their teaching team so that distance education can achieve its aims. Tutors will be the ones who will help and guide students in their paths on the virtual environment.

4 The Virtual Teaching and Learning Environment

The different LMS (Learning Management Systems) available in the market were analyzed. After a first analysis and considering the needs, advantages and disadvantages of each of them, we opted for free solutions, mainly because of their longevity and because, in addition to cutting costs, having access to the source code makes it possible to work with new functions, such as defining a new kind of content directly in the code, as required by the content management needs to introduce specific adaptations. This cannot be done with other proprietary tools. In addition, open source solutions have two important advantages:

- First, their survival does not depend on profit or marketing considerations. As long as the public is interested, the product will survive.

- The free availability of sources is a fundamental guarantee of durability even when, and this is noteworthy, one needs not master them.

Out of all the LMS assessed, we chose Moodle [5] because it was the one who met most needs, it is a free solution and because of our prior experience using it, since this was the tool we had been working with in prior years (although it was an older version). Teachers, students and IT staff had used it before.

The version we chose is Moodle 2.3, which we have internally called UNNOBA Virtual [6]. Adapting the tool includes the following tasks:

- Choosing a visual theme and customizing its style, visual mark and colors.

- Choosing a category and classification system for the courses on the environment.

- Choosing the most appropriate tools to be used in the courses. This includes the types of course that can be uploaded, types of tasks and blocks that can be enabled and built in, etc.

- Choosing and starting an automatic (or semi-automatic depending on the limitations) enrolment policy to enroll students in the courses.

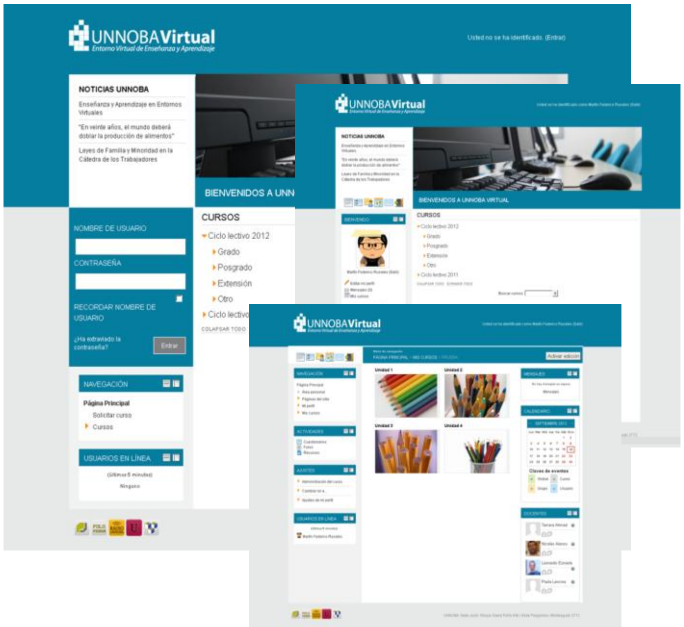
- Selecting and creating roles and their related permissions, for example: student, assistant teacher, teacher, tutor, etc.

- Selecting different formats for the courses, for example: by topic, week, tab, grid, etc.

- Determining what information will be displayed in the summary, for example: should a description be displayed? and if so, in what format, font, color, size, etc. Who is displayed as teacher of the course and the related relevance.

- Migrating courses, users and materials from the current LMS.

At present, the new environment is being adapted and assessed; the whole model is being tried with a reduced number of courses, teachers and students.



Usted no se ha identificado. (Entrar)

NOTICIAS UNNOBA

Enseñanza y Aprendizaje en Entornos Virtuales

"En veinte años, el mundo deberá doblar la producción de alimentos"

Leyes de Familia y Minoridad en la Cátedra de los Trabajadores

Figure 2. The look and feel of UNNOBA's Virtual Environment.

5 References

[1] Lion, C.: Imaginar con tecnologías, relaciones entre tecnologías y conocimiento. Primera edición. Editorial Stella. Buenos Aires, Argentina (2006) [Imagining with technologies, relationships between technology and knowledge].

[2] Alan P.: Enhancing learning and teaching through the use of technology: a revised approach to HEFCE’s strategy for e-Learning. HEFCE. United Kingdom (2009)

[3] Casamayor, G.; Alós, M.; y otros: La formación on-line, una mirada integral sobre el e-Learning, b-Learning. Primera edición. Editorial Graó. Barcelona, España (2008) [On-line education, a comprehensive view of e-learning]

[4] Manzanedo, J.: El e-Learning en España, modelos actuales y tendencias de actuación. Primera edición. Editorial EOI. España (2003) [E-learning in Spain, current models and trends].

[5] Learning Management System (LMS) Moodle, http://moodle.org

[6] Russo C.; Sarobe M.; Pompei S.; De Vito C.; Esnaola L.; Alonso N.: UNNOBA Virtual, una plataforma para la integración de sistemas, metodologías y herramientas de enseñanza y aprendizaje. Publicación presentada en el XIV Workshop de Investigadores en Ciencias de la Computación (WICC). Posadas, Misiones, Argentina (2012) [UNNOBA Virtual, an integration platform for teaching and learning systems, methodologies and tools. A paper presented at the XIV Workshop of Computer Science Researchers].

[7] Entorno Virtual de Enseñanza y Aprendizaje (EVEA) de la UNNOBA, http://virtual.unnoba.edu.ar [UNNOBA’s virtual teaching and learning environment].

What makes the difference, the active learning activities or the technology in the classroom?

Yolanda Martínez-Treviño
Computer Science Department, ITESM, Campus Monterrey
Av. Eugenio Garza Sada 2501 Col. Tecnológico
Monterrey, N.L. México. CP 64840

Abstract. - *In this paper we start by describing the active learning activities used in a CS1 course and how the course changed when we taught it in a classroom with technology.*
Then we describe the research done by comparing the grades of a test applied to the students who took this course on the technology classroom and on the traditional one.
At the end we can conclude that the technology is motivating the students, who think they learn better thanks to the technology; however, we found that what makes the difference is not the technology, but the teaching strategies used.

Keywords: Technology in the classroom, Active learning, CS1.

1 Introduction

On August of 2011 we started teaching a CS1 course in a classroom equipped with technology; since then, we have been gathering data to compare the difference between teaching in this classroom and teaching in a regular one. We have asked students to answer surveys and compared the grades of a test applied in the following course.

The objective of this paper is to present how the course was taught, the difference between teaching this course in the technology classroom and in the traditional classroom, as well as to present the surveys and comparison results.

1.1 Active Learning

Active learning is the practice of involving students in the learning process in the classroom, by reading, writing, solving problems, discussing, etc., instead of having them just listen to the professor's lecture [4].

Different studies [2,3] have shown that using active learning strategies in the classroom result in a lot of advantages for the students, including an increase in their comfort and confidence levels, improvement in the retention of material, etc.

1.2 The technology classroom

The Physics department at Tecnológico de Monterrey has a classroom specially designed for active learning activities. This classroom has been adapted from the SCALE-UP project [1].

This classroom has eight round tables to encourage collaboration between students. These tables allow the professor to have access to all the student's locations to answer their questions during class activities. The classroom also has 4 projection screens so that all the students can always see what is presented. Because of the size of the classroom, a microphone is also available.

Each table is designed to hold up to 3 Tablet PC computers; these computers have the possibility to use digital ink to make annotations in documents. The professor's computer has software that allows her to see what the students are doing with the classroom's tablet PCs while they are working on them.

The classroom has 4 whiteboards on different walls. It is possible to project from the professor's computer, from a camera pointing to one of the whiteboards, and there is also a document's camera, which allows the professor to make notes in a piece of paper and the students can see what is being written in real time.

2 The course

This research has been carried out in the "Fundamentos de Programación" course at Instituto Tecnológico y de Estudios Superiores de Monterrey University, in Monterrey, México. This is a CS1 course where the C++ programming language is used.

This course is intended for the first semester students whose mayors are related to information technology.

2.1 Previous versions of the course

The author of this paper has been teaching different versions of this course for students from different majors. She started teaching CS1 using C++ in 2003, and during this time she has been working on the improvement of the course, specially focusing on the design of active learning activities.

This is a list of the active learning activities that have been used:

- *Active learning presentations.* The course's presentations include slides with questions or activities for the students to perform when they are listening to the lecturer. This is useful because about 12 to 15 minutes after the start of a lecture, student attention starts to decrease, so breaking up the lecture helps to keep the students' attention, and therefore learn better [4].
- *Collaborative activities to learn a new topic.* These are learning activities designed to substitute a lecture. These kinds of activities were designed because in the CS1 course there are some topics that seem to be easy, like the arithmetic operators, and the professor noticed that, after a while, students stopped paying attention. Now, with the new collaborative activities, students have to read the information from handout material, and then they have to work in pairs to solve some exercises. This way the students learn the topic because they have to finish the activity during class time.
- *Written activities.* There are also activities in which students have to trace small pieces of code, or complete spaces in a piece of code, so that they can know for sure exactly how those statements work.
- *Class Practice.* These are activities where students have to make a program on their laptops during class time; the advantage is that if they have questions, the professor is there to help.
- *Completing the code activities.* These are activities where the professor gives a piece of code to the students and they have to complete it on their computers. These activities were designed to make better use of the class time, avoiding wasting too much time typing, for instance, the statements to ask the user for the data. In Figure 1 you can see an example of this kind of activity.
- *Live coding and examples available.* When the students have tried to do an exercise for a while and they have not finished it correctly, the professor starts using live coding [5] to explain the solution. In these situations, the professor has observed that when the students are copying what the professor is typing, in the end, they usually don't get their programs right, since they had been trying to type, to listen what the professor says, and to copy exactly what she typed at the same time. To solve this situation, the professor asked them to pay attention and to not copy the program; at the end of the class, the professor puts the final code in the Blackboard platform.

```
/* This program asks the user for the lengths of the sides of a
triangle.
The program has to display the kind of triangle that it is:
equilateral, isosceles or scalene.
You have to complete the program */
#include <iostream>
using namespace std;

int main ()
{       int side1, side2, side3;

        cout << "type the length of side 1 ";
        cin >> side1;
      cout << "type the length of side 2 ";
        cin >> side2;
        cout << "type the length of side 3 ";
        cin >> side3;

        cout << "Equilateral" << endl;

        cout << "Isoceles " << endl;

        cout << "Scalene " << endl;

        return 0;
}
```

Figure 1. Example of a "completing the code" exercise.

- *Blackboard quizzes.* Students have a quiz available on Blackboard each month before their monthly tests. These quizzes are not considered for their grade, but they help students to practice, and to know if they are solving the exercises correctly.
- *Auto review learning checklist.* This kind of activity contains a checklist with the list of the competencies the student must have before the monthly corresponding test. Students have to review the checklist and mark what he knows and what he doesn't. This is a way to help students to know what topics to pay more attention to before the test.

2.2 Teaching in the technology classroom

When we started teaching this course in the technology classroom, we kept using the same kind of activities used before; however, some adjustments were done to use the technology of the classroom. These adjustments are:

- *The class practice is monitored.* When students are working on a program, the professor watches in her computer what they are doing. This way it is possible to advise students who are solving the problem the wrong way.

- *Active presentations activities monitored.* We mentioned before that the presentations include questions or small activities. In a traditional classroom the students do this kind of exercises in their notebooks and they do not usually share their answers with the group.
  In this classroom the professor has put images with the questions' slides on a Microsoft Word document, and has sent this document to the students at the beginning of the class. This way, when the professor reaches a question or activity slide, she asks the students to solve it. She can see what they are writing, so she only has to wait the time students really need to solve the problem, avoiding wasting time. As she is watching the solutions, she explains the right solutions only when there are students with wrong answers.
- *Round Tables.* The other difference of working in this classroom is the inclusion of round tables, which help students to collaborate when working on exercises, and it's also useful because the professor can reach easily the location of every student to help them answering their questions.

2.3 The procedure

On August 2011 the professor taught the CS1 course on the technology classroom. Let's call this Group A.

On August 2012 the professor taught the same course in the same classroom; this is Group B. This time, however, she asked another professor to teach the course exactly the same way, only without the use technology; let's call this Group C.

Group C had a traditional classroom, with a computer for the professor and a projection screen, and the same difficulty level activities, assignments and tests were used, and the same schedule was followed, so that both groups (B and C) would work exactly the same way.

On both cases, at the beginning of the following semester a test was applied to the students who were enrolled in the subsequent course.

The author of this paper has taught Group A and she also taught the subsequent course in a traditional classroom; so at the end of the second course she applied a survey to students who were her students in both courses; this was done to eliminate the teaching style as a factor that would make a difference in their opinions.

On Table 1 you can see a list of the research activities done with the groups.

Table 1. Activities done with the 3 groups involved in this study.

August 2011	The technology classroom. Group A.
January 2012	Test applied to all students who took CS1 on August 2011. Survey applied to students from Group A, who finished the subsequent course and who took both courses with the same professor.
August 2012	The technology classroom. Group B. The control classroom. Group C.
January 2013	Test applied to all students who took CS1 on August 2012.

3 Results

3.1 What the professor observed

At the beginning of the first semester using the technology classroom, the professor didn't tell the students that she could see what they were doing on the computer, and she observed the students' computers to verify if they were working or were doing something else. The professor observed that the students were doing what they were asked to do, and sometimes, some students went to a browser or did something different, but it generally took a few minutes and soon they went back to the assignment they were supposed to be doing. In Figure 2 you can see an image of the professor's computer where she could see what the students were doing.

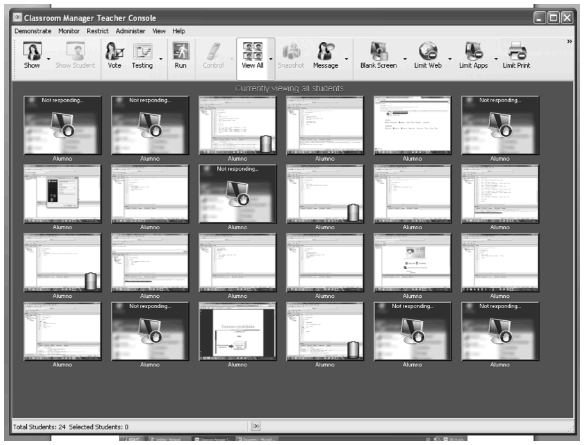
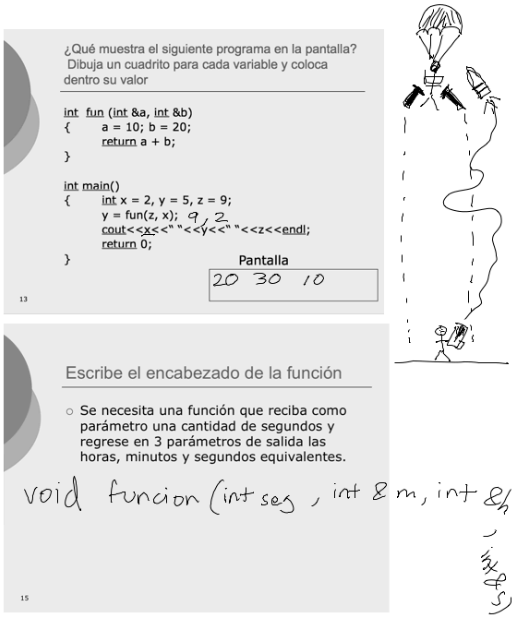

Classroom Manager Teacher Console
Demonstrate Monitor Restrict Administer View Help
Show
Show Student
Vote
Testing
Run
Control
View All
Snapshot
Message
Blank Screen
Limit Web
Limit Apps
Limit Print
Currently viewing all students...
Not responding...
Alumno
Total Students: 24 Selected Students: 0

Figure 2. Screenshot of the professor's computer.

The professor also observed that when students finish their activity before the rest of the group, they start doing something else. In Figure 3 we show the image of a file recovered from a student's computer, where we can see that they had finished their assignment, but also that they were drawing.



¿Qué muestra el siguiente programa en la pantalla? Dibuja un cuadrito para cada variable y coloca dentro su valor

```
int  fun (int &a, int &b)
{       a = 10; b = 20;
        return a + b;
}

int main()
{       int x = 2, y = 5, z = 9;
        y = fun(z, x);  9 , 2
        cout<<x<<" "<<y<<" "<<z<<endl;
        return 0;
}
```

Pantalla

20 30 10

13

Escribe el encabezado de la función

- Se necesita una función que reciba como parámetro una cantidad de segundos y regrese en 3 parámetros de salida las horas, minutos y segundos equivalentes.

void funcion (int seg , int & m, int &h , int &s)

15

Figure 3. File recovered from a student's computer.

3.2 The students' opinion

At the end of the second course, the professor conducted a survey with the students who were in the first course in the technology classroom and with those in the second course with the same professor in a traditional classroom.

Analyzing the results of this survey, we found that the students think they learn better in a classroom with technology.

Some of the most important students' comments, related to the technology classroom, are:

- I paid more attention to class.
- I could see at least one screen all the time, so I don't have problems due to sitting far from the screen, or behind a tall person.
- The class is fun, dynamic and interactive.
- This classroom is more comfortable than a traditional one.
- Some of them said that they like to receive feedback from the professor while they are working.
- It is easy to work in teams thanks to the round tables.
- They like to be able to use the classroom's computer, so that every student can use the computer for the activities even when they don't own one.

In appendix 1 you can see charts showing students answers to some of the survey questions.

4 Quantitative results

On January 2012 we compared the test grades from Group A with the rest of the students who took CS1 in the same semester.

We made a difference of means test and we found that the grades of Group A are significantly different from the grades of the rest of the students, with a p value of 0.0007.

This result means that Group A had arrived to the following semester with a better score than the rest of the students. But in this moment, we couldn't know if this difference was due to the technology, the active learning activities, the teacher's style or because something else.

Then, in the August 2012 semester, we compared the test grades from Group B and Group C (the experiment and the control group).

In this case, we made a difference of means test and we found that there is no significantly difference of the grades of those two groups, with a p value of 0.922.

This means that, in this case, the technology didn't make any difference in the students' learning. So, we thought it could've been the active learning activities what made the difference.

We joined the data of the experiment and the control group (groups B and C) and compared them with the rest of the students who took CS1 in the same semester.

Once again, we made a difference of means test and found that, in this case, this comparison was significantly different with a p value of 0.045.

This means that Groups B and C are starting their following course with a better score than the rest of the students.

5 Conclusions

We know that it is not possible to generalize the results of our study, since the sizes of the groups are not big enough, and there are some non-controlled variables, like professors' teaching styles, schedule of each course, and the fact that the samples used for the study are not random, but instead they were the existing CS1 groups.

However, we can think that it is not the technology, but the teaching strategies what makes the difference in the learning process.

We were also able to find some other conclusions:

- The students work in a collaborative manner because of the round tables.

- The students are motivated to work in the technology classroom because they think that they learn better because of the technology.
- The professor feels that she can teach better because she can monitor what the students are doing all the time and can give feedback to students who are writing a wrong solution.
- It is useful to have technology in the classroom, since it is good to have motivated students because of the use of the technology.

6 Acknowledgements

The author of this paper wants to thank professor María Esperanza Garza Leal for agreeing to work with her in the control group C for this study.

7 References

[1] Beichner, R. S., D, M. J., Deardorff, D, Allain, R., Bonham, S. D., et al. (2006). Students centred activities for large enrollment undergraduate programs (SCALE-UP) project. (P.-B. R. Phisics., Ed.)

[2] Briggs, T. (2005). Techniques for active learning in CS courses . *Journal of Computing Sciences in Colleges , 21* (2), 156-165.

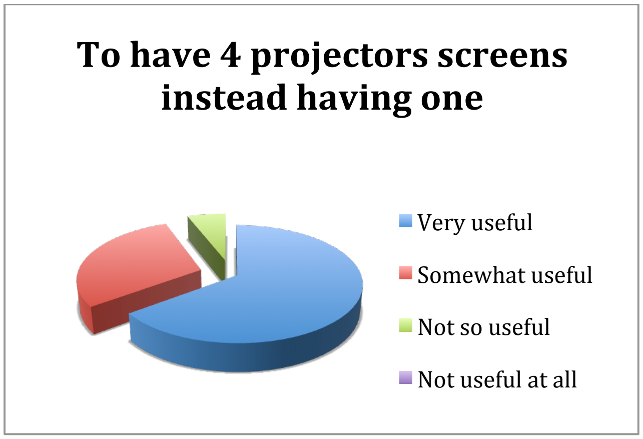
[3] Gao, J., & Hargis, J. (2010). Promoting Technology-assisted Active Learning in Computer Science Education. *The Journal of Effective Teaching , 10* (2), 81-93.

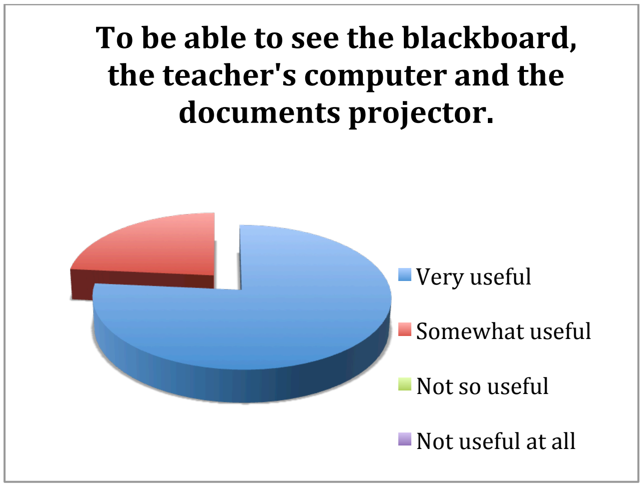
[4] McConnell, J. J. (2005, 07 24). *Active and Cooperative Learning*. (J. J. McConnell, Producer) Retrieved 4 3, 2013, from Dr. Jeffrey J. McConnell: http://www-cs.canisius.edu/~mcconnel/active_learning.html

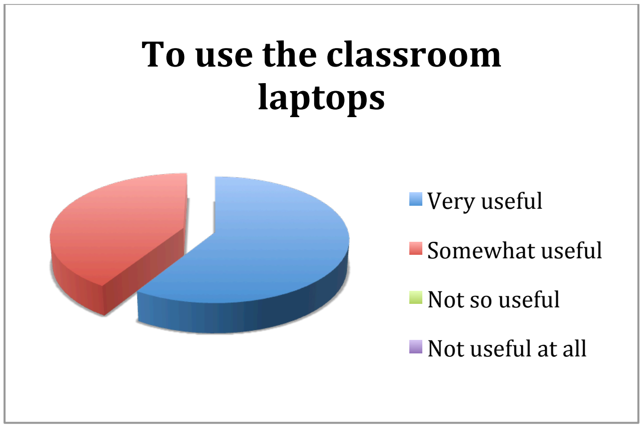
[5] Paxton, J. (2002). Live programming as a lecture technique. . *Journal of Computer Sciences in Colleges , 18* (2), 51-56.

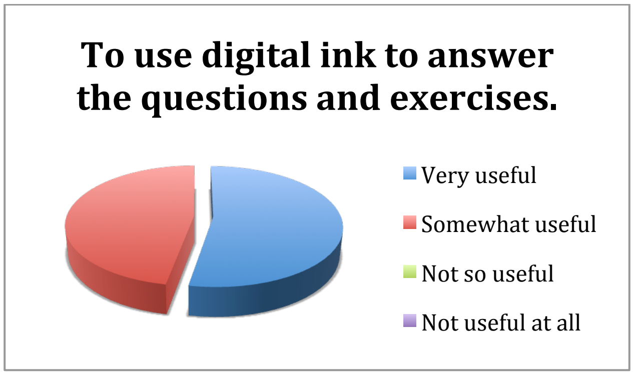
Appendix 1

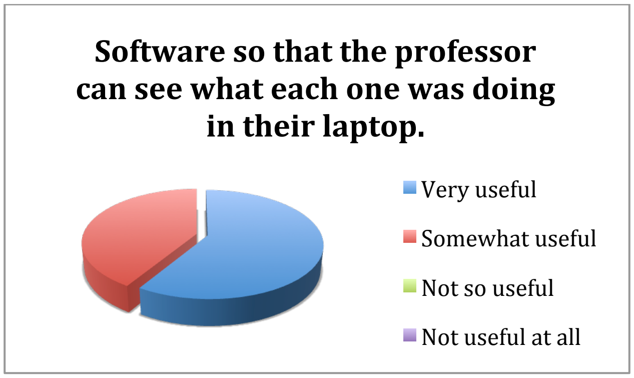
The following charts show some of the questions of the survey and the students' opinion.

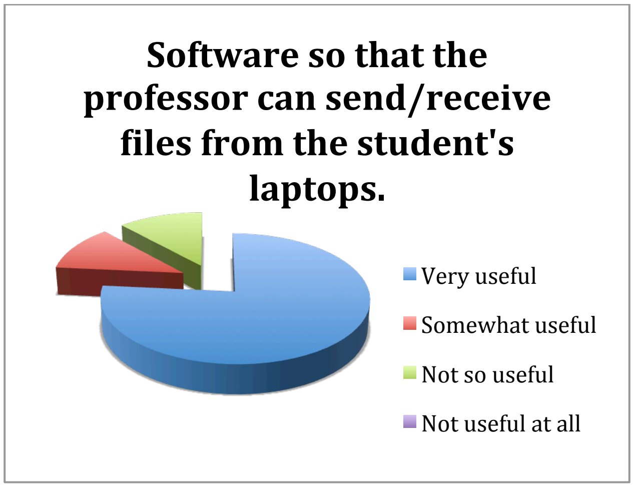

To have 4 projectors screens instead having one


To be able to see the blackboard, the teacher's computer and the documents projector.


To use the classroom laptops


To use digital ink to answer the questions and exercises.
Very useful
Somewhat useful
Not so useful
Not useful at all


Software so that the professor can see what each one was doing in their laptop.
Very useful
Somewhat useful
Not so useful
Not useful at all


Software so that the professor can send/receive files from the student's laptops.
Very useful
Somewhat useful
Not so useful
Not useful at all

Methodology strategies for a University entrance course: Coaching and tutoring, the experience at the National University of the North West of the Province of Buenos Aires, Argentina

C. Russo[1], M. Sarobe[1], H. Ramón[1], N. Alonso[1], L. Esnaola[1], P. Lencina[1] and T. Ahmad[1]
[1] Institute of Research and Technology Transfer (ITT), UNNOBA, Junín, Buenos Aires, Argentina

Abstract - *This document is a description of the experience of UNNOBA in the teaching of an entrance course for the courses of studies of the School of Technology. It describes the different modes, the students' interaction with the virtual teaching and learning environment used, the counseling provided by tutors, different experiences and the relevant analysis that allows us to develop as a public university in Argentina, with the aim of bringing more access opportunities to our community.*

Keywords: entrance course, blended learning, face-to-face learning, student adjustment course, virtual environment.

1 Introduction

The first steps of any student at university are part of a difficult path; most first year students face situations that make them feel anxious, confused and, in many cases, make them drop out of their courses. The University needs to work on policies to include its community as a whole.

It is usual to find certain academic weaknesses in first-year students, as well as financial constraints that force them to work and not focus on their studies. There are others who believe that University is simply not for them, since they come from socially excluded homes. As a result, the University needs to pay careful attention to first-year students, seeking strategies to involve and retain them as part of an educational process that teaches them to be fully-functioning and useful citizens. This is why the strategies implemented by our University become truly meaningful to guarantee student admission and retention.

UNNOBA is an Argentine, state-run university. It is based in Junín but it also has premises in Pergamino. This allows it to set up offices or dependencies in other towns in the north west of the Province of Buenos Aires, the south of Santa Fe, Córdoba and the North East of La Pampa.

UNNOBA meets the educational needs of first-year students by means of an entrance course and a tutoring system.

2 Entrance Course

Since 2005, a face-to-face entrance course is taught at UNNOBA. This is an unrestricted, one-month course comprising two subjects related to the students chosen course of studies. However, in 2010, an innovative proposal was added: a blended-learning entrance course. One of the factors that contributed to this and that has allowed us to work on an extended learning modality was the use of UNNOBA's teaching and learning environment based on Moodle [1].

As discussed above, the entrance course implemented by UNNOBA is taught either face-to-face or as blended learning. Both courses comprise two learning blocks:

- Module I: "*Specific Knowledge Development*": The contents in this module are related to the development of introductory knowledge and methodologies in specific areas related to the course of studies chosen by the student. This module includes two subjects: A and B.

- Module II: "*Introduction to University life*": It provides students with information about the syllabuses, academic structure of the courses, links between areas of knowledge, awareness of skills, attitudes and methodology to gain access to basic and specific knowledge. This module is the same for all the courses of studies taught at UNNOBA [2].

Regardless of the type of course, face-to-face or blended, the entrance course is a mandatory part of an unrestricted entrance system where follow-up and level-up activities are implemented. It is taught as a workshop. The two options, face-to-face or blended learning, cater for student needs. The aim is for students to become autonomous, critical and flexible learners who are able to interpret and handle bibliography and reality and to achieve critical thinking.

2.1 The entrance course experience at the School of Technology

The School of Technology at UNNOBA comprises three areas: Engineering, Design and IT. Module I of the Entrance Course includes two subjects based on the area.

Table 1. There are two subjects per area. This table shows the different subjects per area.

SCHOOL OF TECHNOLOGY		
Engineering	**Design**	**Information Technology**
Subject A: Problem Solving for Chemistry.	**Subject A:** Fundamentals and Theory of Design.	**Subject A:** Introduction to Information Systems.
Subject B: Math.	**Subject B:** Project Workshop.	**Subject B:** Math.

2.1.1 Blended learning course

The students who choose to take the blended-learning course, in August and November of the year before they start their first year at UNNOBA, will be required to do different activities on the virtual platform, tutored by the teachers; and attend 5-hour, face-to-face classes every fifteen days. During the 14-day period prior to the class, the students need to do the activities on the Moodle platform used by UNNOBA as virtual teaching and learning environment (known by its Spanish acronym, EVEA) [3]. These virtual activities [4] [5] are strategically designed so as to supplement the teacher's presence. They include: a welcome forum, an informal tool where students are encouraged to introduce themselves and share some information such as the school they come from, their expectations for the course and anything else they'd like to share with the rest of the class. They are also provided with a tutorial so that they can edit their profile on the platform. These sets of instructions are designed to help students become familiar with the virtual environment.

The class schedules, virtual activities, mandatory assignments and test dates are available on EVEA. Every week, the tutors make some material available to the students and use the internal messaging tool and calendars to let them know about the activities and the materials available.

Another activity that is carried out during the course is related to the students' involvement in a forum [6] but, in this case, students are required to write about a specific topic. For example, for Introduction to Information Systems, students are asked to write about the way they use software and hardware by answering questions such as "What kind of computer do you have?" "What operating system do you use?", "What kind of applications do you usually use and what do you use them for?", "What videogames do you play?", "What are your favorite websites?", "What social networks and forums do you use?" These are elementary warm-up questions that are then discussed in a face-to-face class.

When they are asked to take part in a forum, they are given a set of guidelines; for example, they are reminded that by clicking *Reply* they can continue a conversation, etc. In other cases, they are divided in groups and a coordinator is appointed for each group. Then they are asked to participate in a closed forum (for members only) and to reflect and discuss about a particular topic, for example, they are given a news article that they need to analyze and discuss for a week. After that, they are asked to write a summary of the exchange. This helps them to work as a team in a coordinated manner, where they need to listen to different opinions and reach an agreement [7]. In addition to group activities in a forum, they use the environment to download the theoretical material and reading guides and to submit the required assignments.

During the face-to-face classes, in addition to the lecture taught by the tutors that the students can then download, they are asked to carry out group activities. At times, the groups are organized by the tutors and, at others; students are free to choose their groups thus stimulating the discussion of different points of view.

Other group activities, with different aims and methodologies, are also carried out so as to foster student involvement, for example, discussing an algorithm or producing one for a set of instructions. And, finally, individual work is required to reinforce specific concepts.

As from the fifth week, students are provided with a set of problem-solving exercises and they are asked to work on some of them. In this case, there is a Q&A forum where they can ask questions and where tutors can help them with certain specific questions that may crop up during the face-to-face classes.

2.1.2 Face-to-face Course

For those students who fail the blended learning course or those who enroll directly in the face-face to course, the pace is different. Five-hour classes of the two subjects of the area are taught every day except Wednesdays.

The contents are the same as those in the blended-learning course but they are customized to this face-to-face modality. In the case of Introduction to Information Systems, in the face-to-face course, students are given two sets of exercises. Although most part of the course is taught face-to-face, the EVEA platform is also available for students and it is used as a repository of materials, a message system and it is where assignments are required to be submitted.

Students are invited to take part in the welcome forum just as in the blended-learning course but, for example, in the case of the questions about software and hardware, the discussion is first carried out in class.

Several group methodologies are used [8] for different purposes. For example, in one of the activities, they are divided in groups and given a statement to discuss and elaborate on, find solutions and discuss advantages and disadvantages. Once they reach an agreement, views and opinions are shared with the class and concepts are reinforced.

An activity known as participatory assessment [9] based on previously defined criteria is also worth mentioning. For this activity, students play a role in defining the criteria and in the assessment process itself. A collaborative area is opened on the EVEA platform. Here, tutors act as moderators of the interaction about assessment criteria that will then be used to assess specific content. The class is divided into groups and assigned a closed forum for their exchange. After some time, they are asked to write a document with the assessment criteria considered relevant by their group. To facilitate interaction, tutors appoint group moderators. This allows students to gain awareness of the importance of learning concepts and skills during the entrance course. The final result is a set of assessment criteria. The students can then study the course contents based on this set of criteria.

2.1.3 Course Design on the Virtual Teaching and Learning Environment

Usually, EVEA courses are designed following a weekly format, with tags, links and files which can be activated on a weekly basis. This strategy of hiding future weeks helps students focus on the weeks they can actually see. The aim of leaving past weeks visible is to give everybody the opportunity to revise concepts.

Another important point is the role of visual communication in the production of learning materials. To achieve the learning aims of the course, we take into consideration the fact that any piece of visual communication used should help the students' learning process. Therefore, user-friendly texts, soft fonts and appropriate sizes are used and special attention is given to contrast. When images are used, their relevance is considered. All images should be strictly related to the content. They are not mere decorations, on the contrary, they evidence the key cognitive role of design in the learning process and they cannot be an arbitrary decision made by teachers.

Although the use of the platform for the face-to-face course does not serve the same purposes as in the case of the blended learning course, the idea is to provide technical support and help to all students equally since EVEA will be one of the tools they will be using throughout their years at University.

3 The Tutoring System

This system works as a counseling system for students. Tutors are strategically appointed considering their academic background. They will try to accompany the student during their first year and to guide teachers in their teaching activities.

UNNOBA tutors conduct a yearly follow-up of first year students and try to solve any difficulty that may crop up during this period (for example, helping students to comply with simple administrative steps) and, at the same time, offer workshops to work on study strategies and time management.

Although the tutoring system is designed to help first-year students, students in other years are also welcome. Tutors have several tools to communicate with the students; they can use a course on EVEA or the Tutoring Facebook page. They also have an e-mail account that is informed to the students in case they need to ask questions.

Academic tutoring sessions were introduced in 2011. At these sessions students can ask questions regarding the subjects they find difficult. This aims at reinforcing content and achieving better results.

During the entrance course, the tutors' work is driven by the learning process in a different way. New technologies are used to supplement the teacher's presence in the classroom with innovative strategies.

4 Experience Analysis

The tutoring program of the School of Technology conducted a thorough analysis during 2010 as a result of a statistical survey of first-year students’ performance, comparing the global performance of those students who took the blended-learning course with those who attended the face-to-face classes.

The results were positive in terms of global performance and in connection with the results the students achieved through virtual learning. Since the two modalities and the tutoring system were implemented, academic performance and student retention have increased significantly: 27% and 18% respectively.

In addition we know that the students’ profile has changed, that their needs and backgrounds are different from those of prior years [10], for example, their time availability is different, many students devote part of their time to working and to leisure activities as well as to studying. As a state-run university, we seek to reduce the constraints that hinder learning. Physical distances cannot and should not be an obstacle for people, at any age, to study at University.

The analysis of the results discussed above in connection with the two modalities for the entrance course and the tutoring system is driving the implementation of new proposals to cater for the needs of prospect students. We are seeking to promote the blended-learning modality even more due to its positive results in our recent but valuable experiences.

5 References

[1] Learning Management System (LMS) Moodle, http://moodle.org

[2] Curso de ingreso de la Universidad Nacional del Noroeste de la Provincia de Buenos Aires (UNNOBA), [UNNOBA's entrance course] http://www.unnoba.edu.ar/academica/estudiantes/ingresantes/curso-de-ingreso

[3] Entorno Virtual de Enseñanza y Aprendizaje (EVEA) de la UNNOBA, [UNNOBA's virtual teaching and learning environment] http://virtual.unnoba.edu.ar

[4] Litwin, E.; Maggio, M.; Lipsman, M.: Tecnologías en las aulas, las nuevas tecnologías en las prácticas de la enseñanza. Casos para el análisis. Primera edición. Editorial Amorrortu. Buenos Aires, Argentina (2005) [Technologies in the classroom, new technologies in teaching, case analysis]

[5] Almenara, J.; Graván P.: E-actividades. Un referente básico para la formación en Internet. Primera edición. Editorial Eduforma. España (2006) [E-activities. A basic reference guide for Internet education].

[6] Begoña Gros S.; Mariella A.: Estudio sobre el uso de los foros virtuales para favorecer las actividades colaborativas en la enseñanza superior. Ediciones Universidad de Salamanca. España (2004) [A study on the use of virtual forums to favor colloborative tasks in higher education].

[7] Russo C.; Sarobe M.; Saenz M.; Pompei S.; Ahmad T.; Lencina P: Evaluación en la enseñanza virtual. Publicación presentada en el VII Congreso de Tecnología en Educación y Educación en Tecnología (TEYET). Pergamino, Buenos Aires, Argentina (2012) [Assessment in virtual education. A paper presented at the VII Conference on Education Technology and Technology Education]

[8] Souto, M.: Hacia una didáctica de lo grupal. Miño y Dávila Editores. Buenos Aires, Argentina (2007) [The path toward group didactics]

[9] Russo C.; Sarobe M.; Saenz M.; Pompei S.; Ahmad T.; Lencina P.; Alonso N.; Esnaola L.: Evaluación participativa por criterios, la experiencia de un modelo de evaluación en el curso de ingreso de la Universidad Nacional del Noroeste de la Provincia de Buenos Aires. Publicación presentada en el IV Congreso Internacional de Ambientes Virtuales de Aprendizaje Adaptativos y Accesibles (CAVA). Cartagena de Indias, Colombia (2012) [Criteria-based participatory assessment: an assessment model for the entrance course. A paper presented at the IV Conference on Adaptative and Accesible Virtual Environments.]

[10] Ferraté, G.; Alsina, C.; Pedró, F.; Epílogo: Internet como entorno para la enseñanza a distancia. La educación en la sociedad de la información. En busca de la clase virtual. PAIDÓS (1997) [Internet as a distance learning environment. Education in the information society. In search of the virtual class]

Improvements in Communication Process in Real Projects using Social Network

A. Eulália C. da Mata[1], B. Marcia F. Pinheiro[1], C. Antonio F. L. Jacob Jr.[1], D. Fábio M. F. Lobato[1], E. Carlos R. L. Francês[1], F. Ádamo L. Santana1[1] and G. João C. W. A. Costa[1]

[1]Faculty of Electric Engineering, Federal University of Pará, Belém, Pará, Brazil

Abstract ***- This paper presents an experimental study of integrating a social network to a virtual learning environment. This was done in a real project for a software factory as a tool for collaborative work, media and assist in monitoring and evaluation of activities. The social network organizes all the interactions of users, recording information about the process of serving customers and exchange experiences. A network system consists of technology to the academic environment and to the practice of software development. Students will be together in a virtual space developing projects with real clients.***

Keywords: Real project; social network; software factory; communication; computer engineering

1 Introduction

One of the dilemmas is in the process of learning how to integrate theory and practice. Students have variety of learning styles. The experience of different learning serves to maximize the educational process. In this context we have to use real projects in the learning process. The real projects addressing different learning styles and provide integrated learning situations [1].

The conduction of real projects by undergraduate students provides the possibility of the student experience the practical benefits of the principles of software engineering. The practice of centralizing the development process in user enables greater contact of the student with the real needs of the future users of the system. This means gains in the quality of system functionality and the importance of good communication.

Information technology and communication are possibilities of interaction in the educational system. Social networking has the characteristic of web 2.0. Emerged concepts of human relations in everyday systems. Subject studied by sociologists and anthropologists on aspects of everyday life for many cultures. Social networks propagated on the Internet through various softwares like Facebook, Youtube, Twitter, among others.

The web 2.0 tools in the educational context rescue the interaction, cooperation, collaboration and knowledge sharing in the learning process. The challenge is to incorporate learning activities tools and concepts with other functions. Social networks can be adapted to promote new strategies.

In implementing a collaborative system that consists of characteristics of a portfolio tool for organizing student activities, receive feedback from other students and teacher. And the system can still be used for the appearance of a social network and can be directed to the learning process. There is need for research to understand group behavior in the use of this tool.

In education uses various information and communication technologies to mediate the learning process. Some surveys are conducted in the field of architecture and to use social networking platform Moodle [2].

The research started from the selection of social networking software for the educational context. Went through integration steps, design, and implementation methodologies. Developed a pilot project with seven undergraduates course of Computer Engineering who participated in the software factory. The purpose of this research was to analyze the system of communication that happens in real projects in the context of a software factory with the use of ICT.

The paper is organized as following way: section 2 presents the related work. Section 3 discusses case study: design software with real factory. In section 4, results and discussion. In section 5 conclusion and future work.

2 Related Work

Garrison [3] submitted three groups of graduate students to conduct real projects. In this study, students faced the same problems encountered by professionals, mainly: follow the principles of software engineering; discover the importance of good communication between staff; and developing products for real customers. Upon completion of the development of software, students showed what techniques were used in software projects and lessons learned. In general, students: felt motivated to deliver a good product, worked hard to overcome the challenges of learning and interpersonal relationships; felt and realized gain experience for their résumés.

Hoxmeir and Lenk [4] present the methodology of service-learning as an opportunity for students of information systems to develop and implement real-world systems to nonprofit agencies and government. In these activities, students can actively evaluate and analyze complex taxpayers associated with areas of understanding and to design and implement real-world solutions. As a result students have a deeper technical knowledge learned, improved interpersonal communication skills, greater efficiency in project management skills and an appreciation of the technical knowledge of information systems for the company.

Soares [5] also describes the approach made use of practical work in teams in the discipline of software engineering for undergraduate students in Computer Science. Students have to design software systems simple proposing solutions, analyze and critically evaluate methodologies, techniques and tools. With the implementation of the projects, the students could still have the idea of learning by identifying their problems and correcting its conceptual flaws.

Ordieres, Gonzáles and Albas [6] presented monitoring skills acquisition by a team of students in engineering graduates who were working on projects and who are inexperienced in the area of project management. The experiment aimed to increase the level of skills acquired using a web-based portfolio structure to help reinforce how important different management approaches can result in final products and how important it is to keep integration throughout the project. As a main conclusion, students support the experience of how they perceive reality as a reality immersive enabling them to acquire specific competence. Additionally, students observed the adoption of a working methodology specifies that integrates the work of others.

Nygard et al. [7] conducted a study to assess the potential benefits of social networking tool Ning for collaboration within teams while carrying out a major course project. The work of each team required social and ethical analyses that pertain to a computing or internet related topic. Some of the teams utilized the Ning social networking tool as the primary communication and collaboration vehicle and others used more conventional means, such as face-to-face meetings. Teams with little diversity among its members usually work well in group projects, while team more diverse, often have difficulty. Aiming to evaluate the Ning social network as a tool to increase the performance of heterogeneous teams in order to match with the performance of homogeneous teams, the results indicated that students did not show a particularly favorable attitude towards the use of Ning, although this has proven useful to stimulate the performance and success of the project, the learning curve to master the Ning and heavy workload of the project itself dampened the enthusiasm of students when using the tool.

3 Case Study: Factory Design Software with Real Projects

The Software Factory concept emerged in the mid-80s through the project "Fifth Generation Project" developed by Japanese who have taken teams of highly trained insertion and use of metrics to measure improvements in the development process and software quality [8].

As a complement / improve this model, we sought to adopt the User-Centered Design (UCD). This process provides several moments of interaction with each artifact generated in order to make the client more participatory and decision-making power. In this sense, communication becomes the primary resource for understanding the project to be developed and the progress of each process step.

Since it reading email is still one of the main activities performed on the Internet (http://pewinternet.org/Reports/2011/Search-and-email.aspx), the members of the Software Factory use e-mail and face meetings for communication and exposure of tempos of research, ie, that happens through these means monitoring by the coordinator of the project and the client. Furthermore, email is used for sharing information about programming languages, software engineering and related topics to the themes of the research [9].

These activities provide software factory that has characteristics of active learning [8]. This type of learning is the incentive to search for new knowledge to solve problems, the starting point of the project, aiming to interfere positively on the skills of planning, organization, execution, those involved [10].

At the time of research, the factory had seven ongoing projects. And knowing the relationship of communication and collaboration that is needed in the development of research and practices of everyday life among the team members, it was proposed to study the implementation of a platform composed of web 2.0 tools to be used by the plant's human resources along with customers.

4 Results and Discussion

The first stage of the implementation process of the social network had external variables that hindered the events expected as ideal. But we know that this deployment experience in the context of a software factory reflects situations that happen in several companies that innovate in the use of information and communication technologies in order to enhance collaboration between the actors.

Despite having a team of factory reduced number of external customers still small, the process suffered impacts from the main feature of this academic factory software, which is software developments being made in decentralized locations. There was the challenge to gather all team members.

At the meeting presential social network was presented as a new possibility of communication with the supervisor and clients. And all members were invited to initiate access to the system. Rescued that the social networking site Facebook, more accessible at the time, emerged from academia to the global context, and that the implementation of the social network along with the virtual learning environment allows the recording of the collaborative process that occurs between all members team this tool and insert the customer, who can follow the step-by-step software development.

Team members were instructed to use the tools, organizing the portfolio on the project for which each is responsible. And after use to answer a questionnaire to assess the use of this new tool. And as was to document the practice of social networking by staff members of the software factory, analyze the responses in the survey. On the social network the item ease of access (25% excellent and 75% good), flexibility (50% excellent, 25% good and 25% satisfactory); system in general (75% good and 25% satisfactory) and process interaction or collaboration (75% good and 25% fair). The most positive responses demonstrates the acceptance by the implantation of the system.

The system records a group created - Software Factory - which contains fourteen users ranked institutions: client or moodle. There are four customer profiles, two administrators, seven students and a supervisor of the software factory. With the encouragement of participation were created 49 pages on the projects over a period of five days of collection of statistical data. We use Moodle that the two institutions associated with social network validates this new login system using the same user. The client institution is required to other participants of real projects that are not registered in the virtual environment.

Statistical data of the most visited pages and commented are also provided by the system. The most visited page and commented that it was the content includes the description of research activities and implementation of the social network. You must create tutorial to facilitate use of technology, despite the students being the field of computing.

The project has the real focus of customer involvement in software development. Using social network facilitates the communication process between the customer and factory software. Having a social networking system that enables the recording and monitoring will facilitate the involvement of customers with projects. Developers there is the possibility of having a faster feedback to proceed in steps of software development. Besides facilitating the monitoring of performance of students by teachers. The social network brings the possibility of greater customer participation to the project.

5 Conclusion and Future Work

This article presents a pilot project deploying a social network in academia seeking to provide better performance of real projects that are designed for students who are part of a software factory. The use of a social network provides greater organization to interactions.

The results on the social network showed that the system is simple to learn and help in the process of monitoring, evaluation and collaboration of activities performed on the stage of the software factory. The real projects earn an additional tool to assist in the communication process. The social network that is already known to a context of society returns to use in academia. This deployment extends the space for collaborative learning. The team shares information from software developments.

In future work, we plan to evaluate the system in a consolidated environment software factory and the proposed expansion to the college academic environment, aiming to provide a collaborative environment among students and teachers, groups currently use e-mail discussion groups

And future work include evaluating the use of the system and interactive design. Observe the behavior of users in the system. If changes are needed will be made

6 References

[1] SCHULDT , B. A., *“Real-world' versus `simulated' projects in database instruction”*, J. Educ. Bus., vol. 67, pp. 35-39, Sep/Oct 1991.

[2] BRAZ, L. M., SERRÃO, T., PINTO, S. C. C. S, and Clunie, G. 2012. *An Architecture to Promote Integration between Moodle and Social Networking Sites*. In Telematics and Information Systems (EATIS) Conference on Euro American. (May 23 – 25, 2012). ACM.

[3] GARRISON, C. P., *A Graduate Level Case Study Using a Real World Project: What Students Say They Learned, The Open Software.* The Open Software Enginneering Journal, 2008, Vol. 2, pp. 31-39.

[4] HOXMEIER, J.; LENK, M. M, Service-Learning in Information Systems Couses: Community Projects that Make a Difference. Journal of Information Systems Education, Vol. 14, No. 1, pp. 91-100, 2003.

[5] SOARES, M. S., *An Experience of Teaching Software Engineering oriented practical work*, I Workshop Education in Computing, pp. 1-10, 2004.

[6] MERÉ, J. O.; GONZÁLEZ, A. M.; ALBA, E. F., *The Use of Computers for graduate education in Project Management. Improving the Integration to the Industry*, The 2011 International Conference on Frontiers in Education: Computer Science and Computer Engineering, 2011

[7] NYGARD, K., E., et al., *Collaboration using Social Networks for Team Projects*, The 2011 International Conference on Frontiers in Education: Computer Science and Computer Engineering, 2011.

[8] TARTARELLI, R.V.; WINCKLER, W. S. *Organizational learning in software factories*, Anais, I Seminário de Gerenciamento de Projetos, PMI-RS, 2003

[9] PURCELL, K. Search and email still top the list of most popular online activities. 2011.

[10] NORMAN, D. A.; SPOHRER, J. C. *Learner-centered education*, Commun. ACM, 39, 1996, p. 24-27

[11] PRINCE, M. *Does active learning work*? A review of the research, J. Eng. Educ., 93, 2004.

[12] About Mahara. WebSite. <http://mahara.org/about>.

ACM ICPC In China : Learning From Contests

Dalie Sun[1], **Lixin Wang**[2], **Zhaonian Zou**[1]
[1]School of Computer, Harbin Institute of Technology, Harbin, Heilongjiang, China
[2]School of Electrical Engineering, Harbin Institute of Technology, Harbin, Heilongjiang, China

Abstract - *ACM ICPC is widely famous programming contest in the world. With its unique feature, it attracts more and more students who are interesting in programming. In China, it plays an important role in improving the college education and students' abilities. Taking HIT as an example, to explain how ACM ICPC has been developed in China and what they achieved by taking part in ACM ICPC. The experience of HIT shows that by taking part in ACM ICPC the students not only improve their algorithm and coding practice, but also promote their personalities, such as team spirit, learning mode, confidence, jobs, etc.*

Keywords: ACM ICPC, HIT, advantage, benefits

1 Introduction

1.1 Fast Development of ICPC

ACM International Collegiate Programming Contest (ICPC), which dates back to 1970, has become a famous programming contest for college students all over the world. After more than forty years of development, ICPC developed from a league match held by students in a few universities in Canada and the U.S. to a worldwide competition. As the table 1 shows, in the first 22 years all the ICPC world finals(WF) had been held in USA. Then it went around the world.

Table 1 City/Country Held WF

Year	City/Country
2013	St. Petersburg / Russia
2012	Warsaw / Poland
2011	Orlando, Florida / USA
2010	Harbin / China
2009	Stockholm / Sweden
2008	Banff / Canada
2007	Tokyo / Japan
2006	San Antonio, Texas / USA
2005	Shanghai / China
2004	Obecni Dum, Prague / Czech
2003	Beverly Hills, California / USA
2002	Honolulu, Hawaii / USA
2001	Vancouver / Canada
2000	Orlando, Florida / USA
1999	Eindhoven / The Netherlands
1977-1998	~ / USA

As the table 2 shows, 9900 groups from 2300 universities in 91 countries joined Regional contests in 2012, three times that of a decade ago. And 120 teams from all over the world participated in the annual World Finals in 2012.

Table 2 Growth in Amount

Year	Countries	Colleges	Teams	Finals
2012	91	2300	9900	120
2010	82	1981	7216	103
2004	75	1411	3150	73
1999	59	839	1900	62
1994			628	35
1989			400	25

In some countries, such as China, there are also many different kinds of programming contests, such as national invitation contests, provincial contests and college contests. If we were to add all these participants to the figure, it would be doubled. So we can see that ICPC is becoming more and more popular among college students.

1.2 Great Zeal of Chinese

ACM ICPC has been developed in China at an amazing rate. In 1996, Shanghai University hosted the first Asia regional contest(Regional) in China. And it opened the door for Chinese students to ACM ICPC. At first years, few students knew of this kind of contests. As Regionals were going on year by year in China, more and more college students got exposed to it and the amount of participation increased gradually in period 1996 to 2003. There was a big jump in amount of participants by 2005. And the number of regional site in China increased from 2 to 5 in 6 years.

Table 3 shows the number of Regional site in China and the city/cities where the Regional was held. From Table 3 we can see that Regional spread from Shanghai to all over the China. Ever though the Regional was held in the same city more times, the Regional was usually hosted by different universities.

Though there are 5 Regional sites in China, it wasn't able to accommodate all participants. There are too many students who wants to participate in the Regional. So online contests were set up before Regionals to screen participants. There are about 800 teams participating one online contest. And out of them about 120 teams are chosen to Regional.

Table 3 Regional Sites and City

Year	Sites	City/Cities
1996-2001	1	Shanghai
2002	2	Beijing/Xian
2003	2	Beijing/Guangzhou
2004	2	Beijing/Shanghai
2005	3	Beijing/Hangzhou/Chengdu
2006	3	Beijing/Shanghai/Xian
2007	4	Beijing/Nanjing/Chengdu/ Changchun
2008	5	Harbin/Beijing/Hefei/ Hangzhou/Chengdu
2009	5	Harbin/Hefei/Ningbo/ Shanghai/Wuhan
2010	5	Harbin/Tianjin/Hangzhou/ Fuzhou/Chengdu
2011	5	Dalian/Beijing/Shanghai/ Fuzhou/Chengdu
2012	5	Changchun/Tianjin/Hangzhou/ Jinhua/Chengdu

Table 4 Amount of Teams

Year	Online Contests	Regionals
2012	1200	520
2008	880	300
2004	0	180
1996	0	45

Besides Regionals, there are many kinds of contests of ACM ICPC in China. Such as National Invitation Contest, Northeast/Central China Contest, Guangdong/ Heilongjiang/Zhejiang/Fujian/Jilin/Liaoning/Sichuan Provincial Contests, etc. Many colleges even hold annual contests for all enrolled students, such as Zhejiang University, Harbin Institute of Technology(HIT), Heilongjiang University, etc. By this way, ACM ICPC provides opportunities for all students in China.

1.3 Role in Education

In China, ACM ICPC plays an important role in university education. Before admitted to their college, even in college, our students pay their almost all attention to kinds of courses examinations, which is just only writing their answers on the test papers. It results in that they never enter the laboratory to check their solutions, never do practice working. They learn from books to books, put the 'answers' into their brains just for passing the examinations, have no attention for using what they learned to solve the problems in the life. This kind of study mode is very sadness. Nowadays, Chinese universities want to change the situation, and one of the measures they are taking is to encourage students to join in contests and social practice. ACM ICPC is one of our recommendations.

In following sessions, it is described at first that ACM ICPC's unique styles, and the value that Chinese students get from these styles. Then to show how Chinese universities get involved in ACM ICPC and gain benefit from it by taking HIT as an example.

2 Advantage of ACM ICPC

It is ACM ICPC's unique styles that attract these youngsters. And these styles have special meanings for Chinese students.

2.1 Problem Solving

In ACM ICPC, every subject is described as a realistic story and problems are involved in these stories. Participants must understand the meaning of the story and get to know what the problem is at first. Then determine how to solve the problem ------find the algorithm. Finally give the final solution by coding.

It is a great challenge for Chinese students because they have had few such experiences ever. They are usually given questions directly, so they have no need to dig them out. However, problems in our life are always lying in 'stories'. The success may just due to how to understand the story.

2.2 English Description

Every story is described in English. For Chinese students, it is an extra burden. But it is also a great opportunity for them to improve their English. Our students have taken so much time in learning English from primary school to college. But their English is not good enough. Only in English course classes they speak/read/write English. There is rarely chance for them to practice their English in life. ACM ICPC provides an environment for them to practice English, especially in reading and writing in English.

2.3 Team motivation

In ACM ICPC, one team is composed of three participants and they use one machine together to solve 10 problems within 5 hours. First of all, what they do is distribute the problems among them three persons. In other words, to make sure who works on which problems. Then to distribute the time and order that each one has to occupy the machine. The third is helping and cooperating with each other. When a member of team works on his topics, he often needs his teammates to offer necessary support, such as data testing and debugging. All these matter require the three team members to have good understanding, cooperation and compromise. Chinese students are unfamiliar with this fashion, because most of them are the only child in their family and spoiled by their parents.

2.4 Test conditions

To be accepted, the submitted program must be satisfied the test conditions. Data tests include several parts, such as program running time, occupying memory sizes, input and output format. All these characters are basic requirement of software engineering. It is nessesary for a good software engineer to be good at these features. Because they have solid foundation in these aspects, the students who have experienced in ACM ICPC are well received when they look for job.

2.5 Competitive state

The final result are often depends on contestants' strength, but their performances during the contest is of particular importance. During the race, whenever the program submitted by a team get the test, a color balloon will be rose on their table. It is exciting not only for this team and also for all other teams. How do you feel? Can you control yourself as usual when looking the balloons of your neighbors rose one by one? It is a great challenge for all students' psychological quality. It has a major impact on students' ability on analyzing, judging and making decisions on that occasion. What the case in their real life?

2.6 Communication

ACM ICPC set up a platform for all university students. Every yrar there are over 50 thousand students taking part in all kind of ACM ICPC contests. They not only compete with each other in contests, but also learn, commucate with each otger. It formed a large community among these young peoples. Many of them become friends and the friendship continued until they graduated from universities.

3 ACM ICPC in HIT

Harbin Institute of Technology(HIT) has taken part in ACM ICPC since 1998. In the beginning, we take part in Asia Regional only because there are no other kinds of ACM ICPC contests in China. And we have one team participating Regionals until 2003(2 in 2002). As Table 5 shows, the amount of teams is increased gradually from 2 to 4 in period 2004-2007. And it has a jump in 2008.

Table 5 Teams and Prize - Regionals

Year	Teams	Prize
2012	14	Gold 3/Silver 3/Bronze 3
2011	15	Gold 1/Silver 7/Bronze 3
2010	15	Gold 1/Silver 9/Bronze 3
2009	12	Gold 1/Silver 5/Bronze 2
2008	12	Silver 4/Bronze 5
2007	4	Silver 2/Bronze2
2006	4	Gold 1/Silver 1/Bronze 1
2005	3	Silver 2/Bronze 1
2004	2	Bronze 1
2003- 1998	1/year 2/2002	Silver 2/Bronze2

The important reason is that they got a specific teacher to be their coach and obtained more support from university. In 2007, the university authorities evaluated the influence and effect of the contest, and they assigned a coach to be in charge of it. By the guiding of the coach the contest was expanded to the whole university. Table 6 and 7 show the amount of teams and prize in two kinds of contest, northeast China and Heilongjiang provincial. The contests began from 2007 and 2006.

Table 6 Teams and Prize – Northeast China

Year	Teams	Prize
2012	10	Gold 5/Silver 2/Bronze 3
2011	9	Gold 4/Silver 2/Bronze 3
2010	9	Gold 4/Silver 5
2009	10	Gold 2/Silver 4/Bronze 1
2008	7	Gold 2/Silver 4/Bronze 1
2007	7	Gold 2/Silver 2/Bronze 3

Table 7 Teams and Prize – Heilongjiang Provincial

Year	Teams	Prize
2012	15	Gold 7/Silver 1/Bronze 6
2011	15	Gold 6/Silver 8/Bronze 1
2010	15	Gold 6/Silver 6/Bronze 2
2009	12	Gold 6/Silver 5/Bronze 1
2008	11	Gold 3/Silver 6/Bronze 2
2007	7	Gold 5/Silver 1/Bronze 1
2006	7	Gold 4/Silver 3

We have confirmed our demand when developing ICPC in our university at beginning. That is to improve our students' algorithms and coding practice in ACM ICPC, and enrich other aspects. To fulfill this, we offer opportunities to participate as many as possible, so that students can experience the excitement and interest of the contest. We also integrate ACM ICPC into programming courses, and they can get mutual promotion.

3.1 Participating in contests

There are different kinds of ACM ICPC contests we take part in every year. In spring, there are Heilongjiang provincial programming contest, Jilin provincial programing contest, National invitation programming contests, Northeast China programming contest and ACM ICPC World Finals. In autumn, there are five Asia Regionals held in China. Table 8 shows contest list that we take part in this year.

Table 8 Contests List - 2013

Date	Contest	Teams
May 12	National invitation(Nanjing)	1
May 19	Heilongjiang provincial	15
May 26	National invitation(Changsha)	1
May 26	National invitation(Tonghua)	1
Jun 2	Jilin provincial	3
Jun 2	National invitation(Hangzhou)	1
Jun 9	Northeast China(Mudanjiang)	10
Jul 3	World Finals(St. Petersburg)	1
Oct x	Regional(Changchun)	3
Oct x	Regional(Hangzhou)	3
Oct x	Regional(Nanjing)	3
Nov x	Regional(Changsha)	3
Nov x	Regional(Chengdu)	3

3.2 Holding contests

We hold ACM ICPC HIT programming contest twice a year, during fall and spring semesters, and they are on site contests. We also hold 12 weekly contests every semester which are in form of online contests. All these contests are managed by our senior team members.

In addition, we undertook ACM ICPC Heilongjiang provincial programming contest in 2008 and Regional in 2009. Over 120 teams took part in both these contests, and almost 200 college students helped to organize the contest as volunteers.

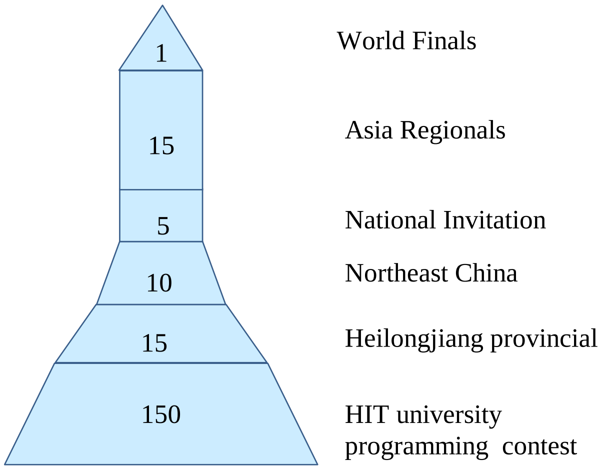


Figure 1 Teams Participating Contests

3.3 Holding lectures

There are two large-scale lectures every semester that focus on ACM ICPC, and nearly 200 students are present each time. These two lectures are just like advertisements about ACM ICPC. The main purpose is making ACM ICPC to be known around the fresh students. There are also 12 evening seminars followed the weekly contests every semester. The number of participants is about 30-60 each time. The first part of the content of seminars is talking about the solutions of problems of weekly contest, and the second is a free talk. All the seminars are presided by undergraduates who have taken part in Regionals and got a silver prize.

3.4 Setting up ACM ICPC course

We have set up a course of ACM ICPC, named Programming Practice 1, and all students can choose it no matter what major they are. The course is offered every semester, 30 hours in total, 3 hours a week. Students take class in the computing laboratory, one person, one machine. On the class, students write codes with teacher (TAs), discuss algorithm and coding each other. In each class we focus on one kind of algorithm, all the examples come from ACM ICPC's problems. All TAs are chosen from the undergraduate students who have got at least a silver prize on Regionals.

It is a new style of courses in our university. Even now our classes are almost in one mode in which teacher is speaking and students listening. If we take a picture that class is a drama performance, then teacher is only actor and director, all students are audience. It is not good, especially for practical courses like programming.

3.5 Undergraduate as TA

In 2008, we began to use undergraduates as the TA in the course of High Level Language Programming. This course is opened for the freshmen who major in computer. About 240 students choose this course every year, and if 15 students are in the charge of a TA, it needs about 20 TAs.

In 2011, we began to use undergraduate as the TA in the course of C Language Programming. This course was offered for all the freshmen whose majors are not computers. About 3,000 students choose this course every year. Now it is just a trial and only adopted by a few classes. Every year, there are 40-50 TAs, and all of them are volunteers, no any payments.

These TAs are selected from the members of ACM ICPC contestants, and most of them are sophomores. Juniors and seniors who are former members are in charge of them. They use vacation time to give a three weeks training program to the TAs and lead them to finish the TA's work. These TAs are responsible not only for tutoring students during the 30 laboratory hours, but also for designing the contents of experimental lessons, routine assessment, check homework, questions and answers etc. At the same time, one part of the undergraduate TA plan, is to do research about the course through questionnaires. It can help us to know more about students' real situations and their needs. What's more, it can provide powerful support to the course's improvement.

3.6 ACM ICPC volunteers

Most of the work that ACM ICPC members do is volunteers' work. Our online judge(HOJ) was developed by one of our contestants in 2002. Our university contests and weekly contests are operated by higher grade members. The TAs responsible for the course of C Language Programming are volunteers from our team members. In resent two years, we connected ACM ICPC with Olympic Information Contests among high school students, and assigned junior members to go to high schools to provide tutoring and lectures, to help these schools develop programming activities.

3.7 Self-management

We founded a society organization, ACM ICPC Club. All the students who have taken part in ACM ICPC can become club member. This club is responsible for all activities regarding ACM ICPC in HIT, including organizing students to take part in different kinds of contests, holding university contests, holding lectures, selecting and training TA's, fixing HOJ etc. The coach provides guidance, while all the specific tasks are done by students.

4 Benefits

It is most important achievement that our students have improved themselves greatly by taking part in ACM ICPC contests. They got advantages not only in improving their algorithm and coding practice, but also in other aspects. What's more valuable is that students have gained great perfection in many other fields.

4.1 Improvement on programming

The most direct benefit that students have got in ACM ICPC is great progress in algorithm and coding. Just like athletes, they spent much time on preparation works before participating the contests. They devote themselves to learning algorithms and coding day by day. They do many online contests. After such an experience, they are familiar with many kinds of algorithms, and know when to use them and how to implement them by writing code quickly.

Table 9 shows the rank of our teams in the World Finals. One of our teams have got the 5th rank in Regional (Fuzhou) in 2010 and won the opportunity to take part in the World Finals which was held in Orlando in 2011. It is the second time for us to World Finals. Our contestants solved 5 problems and got the 39th rank out of 110 teams. One of our teams got the 7th rank in Regional(Chengdu) last year, so we gained thirdly eligibility to World Finals, the 2013 Annual World Finals which is hosted by St. Petersburg University ITMO from Jun 30 to Jul 3.

Table 9 Rank – World Finals

Year	Rank	City
2013	?	St. Petersburg / Russia
2011	39/110	Orlando / USA
2010	70/105	Harbin / China

The good performance in ACM ICPC contests lay a solid foundation for students' future studying and working. Our team members have advantages over other students in their later study. Almost all specialized courses have a project which usually take most of students 3 weeks to finish, while our members can finish it with an A grade within one week.

4.2 Willpower and stamina

Students spent almost their leisure time to do contests when they joined ACM ICPC. They studied algorithms and practiced coding every day sitting in front of machine. For our undergraduate student, if he wants to take part in Regionals, he needs at least have two year experience in ACM ICPC. That is, he should stick to daily researching and coding in two years long. It is very boring to do exercises in front of screen for so long time.

It is not easy for our young people to devote themselves into contests when their classmates immersed in playing games day and night. It nurtures students' willpower and stamina if they can keep going. Putting up with loneliness is one of the most necessary qualities of a science researcher.

4.3 Interesting and goals

ACM ICPC may enhance the students' interesting on computer science and technology and help them getting their life aims. More than ninety percent of our students have no any experience in programming before they enter the university. Even though they have been provided courses on programming in university, they have seldom shown their interesting on programming, because the course is not stimulating enough.

By participating in ACM ICPC, students develop interesting in algorithm and programming gradually, so that they will spend more spare time on coding practice. It is very meaningful for them. It makes them set goals and what's more, they will spend less time on playing video games.

4.4 Learning mode

In the process of ACM ICPC, students may always be exposed to new knowledge. They acquire the knowledge through independent study and discussing with others. It is a new learning mode for our students. Our learning mode used to be that students just listen to the teachers without independent thinking. As a result, they are not good at discussing and independent study. ACM ICPC changes their learning style. When they face the contest problems, they know what they want to learn and why to learn it, learning ability and efficiency are greatly improved.

4.5 Team spirit

Students said that friendship between team members are more important than programming skills they got in taking part in the contests. Team mode that three students are in one team makes them understand the importance and necessity of cooperation. They begin to have team spirit, and master the skills and methods of cooperation. For Chinese students, it's a great progress.

4.6 Research and job opportunity

More and more undergraduates are inclined to practice in enterprises, but not all enterprises accept them. Some enterprises have very high requirements, such as Microsoft, Google, Baidu, Ali, Tengxun. Our students are eager to have a chance of intern in these companies. But it is difficult for them to get offers from these companies, while it is very easy for our team members.

The students who have taken part in ACM ICPC have much opportunity and good payment when they get job. The undergraduate students who major in computer in HIT earn 70,000-80,000 CNY per year on average after they graduate, while our team members' salary is at least 150,000 CNY after graduation.

Almost half of our ACM ICPC team members would like to continue their study and research on Master's degree or Ph.D. Because of their contest experience, they can get offers easily from famous universities over world.

Table 10 shows whereabouts of some of our Regional team members when they graduated. Row 1 is the year in which they take part in Regionals. Row 2 is the year when they educated. Row 3 is the name of students. Row 4 is what kind of work they do. Row 5 is the name of the companies and universities which they are employed.

Table 10 Whereabouts of Regional Members

Year 1	Year 2	Student	Category	Company/Universiy
2011	2013	Jing Yang	employment	Google / Australia
2011	2013	Lixin Fan	employment	Google / Shanghai / China
2011	2013	Jianliu Hu	employment	Ali / Hangzhou / China
2011	2013	Qiankun Zhu	Ph.D	Chinese University of Hongkong
2011	2013	Yi Lu	Ph.D	Chinese University of Hongkong
2011	2013	Yingying Ma	Ph.D	Tsinghua University / China
2011	2013	Yue Hu	Master Degree	Carnegie Mellon University / USA
2011	2013	Xi Zhu	Master Degree	Harbin Institute of Technology / China
2011	2013	Zhihang Fan	RA	National University of Singapore
2011	2013	Liangjun Song	RA	National University of Singapore
2009	2012	Yehong Zhang	Ph.D	National University of Singapore
2009	2012	Bei Chen	Ph.D	Peking University / China
2009	2012	Yu Ding	Master Degree	Harbin Institute of Technology / China
2010	2012	Wenzhe Pei	Ph.D	Peking University / China
2010	2012	Tao Guo	Ph.D	Nanyang Technological University /Singapore
2010	2012	Yijia Liu	Master Degree	Harbin Institute of Technology / China
2009	2011	Sheng Wang	Ph.D	National University of Singapore
2009	2011	Feifei Ji	Master Degree	University of Pennsylvania / USA
2009	2011	Zhen Jia	employment	Tengxun / Shenzhen / China
2009	2011	Yu Sun	employment	Baidu / Beijing / China

4.7 The effect of TA

It achieves a good result to use undergraduate as TA of courses.

- They are more responsible than postgraduates, and do the job more positively and carefully. What's more, they are strict with fresh students.
- They do the work more efficiently than postgraduates. The undergraduates with ACM ICPC experience are not less capable than postgraduates in algorithm and programming. They know more about freshmen, so they are easy to get along well with them.
- They tutor students and their algorithm and programming levels get a big promotion at the same time. What is the most important is that, for TAs themselves, their expression and communication ability improve greatly.

5 Conclusion

Our students have received textbook-based education for a long time, and they are lack of practice experience. They can make up for it through participating in ACM ICPC. HIT takes many ways to develop ACM ICPC for all students. They popularize ACM ICPC by setting up the course, holding week contests(lectures) and college contests, appointing undergraduates as TA and so on, to make more students participate in it and get improvement. In this process, they select high-level students to take part in ACM ICPC's Provincial contest, Northeast of China contest, Regionals , up to the World Finals.

The achievements are showing up gradually. Students with ACM ICPC experience gain popularity from research institutions and enterprises. When they apply for master and PhD or work in a company, this experience offers them a big advantage.

We are still making this project better now, to let more students join in ACM ICPC, be interested in computer and become a computer master finally. We plan only to use undergraduates as TAs in C Language Programming course. There are about 3,000 fresh students choosing this course every year. If we assign 15 students to a TA, we need about 200 TAs. We only have 50-60 undergraduates as TAs now, so there remains a big gap between these two figures.

6 References

[1] Eric Roberts, John Lilly, and Bryan Rollins.Using Undergraduates as Teaching Assistants in Introductory Programming Courses:An Update on the Stanford Experience. SIGCSE Bulletin. Volume 27 Issue 1, Pages 48-52, March 1995.

[2] Stuart Reges. Using Undergraduates as Teaching Assistants at a State University. SIGCSE '03 Proceedings of the 34th SIGCSE technical symposium on Computer science education. Volume 35 Issue 1, Pages 103-107, January 2003.

SESSION

TEACHING PROGRAMMING + SOFTWARE ENGINEERING AND SYSTEM DEVELOPMENT + TESTING AND DEBUGGING + ONLINE COURSES

Chair(s)

TBA

Software Development Educational Pathway: AS to BAS

Colin Archibald[1], Richard Grant[2], Craig Tidwell[3], Ali Orooji[4] and Greg Garver[5]
[1]Information Technology, Valencia College, Orlando, FL, USA
[2]Computer Programming, Seminole State College, Sanford, FL, USA
[3]Regional Campuses, University of Central Florida, Orlando, FL, USA
[4] Electrical Engineering & Computer Science, University of Central Florida, Orlando, FL, USA
[5] Computer Science, Eastern Florida State College, Melbourne, FL, USA

Abstract - *A consortium of four state colleges and the University of Central Florida, has created a Bachelor of Applied Science (BAS) in Software Development. The colleges offer a well established Associate in Science (AS) 2-year degree in Computer Programming. Graduates of this AS degree have had no logical pathway to a bachelor's degree until now. The bachelor's degree in Computer Science at UCF provides a pathway to a bachelor's degree for the Associate of Arts graduates, but is not appropriate, nor available, for the AS Computer Programming graduates. The missing link has been an applied bachelor's degree articulated with the AS program. Graduates of the AS program at the four partner colleges are guaranteed admission to UCF to pursue the BAS degree. The AS can be completed online or in traditional classrooms. The BAS is offered exclusively online. The BAS Software Development curriculum was developed with direction from industry partners.*

Keywords: software development, associates degree, applied bachelor's, applied science, articulated degrees

1 Introduction

There are many career options available for software developers varying from mathematical to artistic. The curriculum for computing in the universities is frequently limited to a Bachelor of Science in Computer Science (BSCS) which is one of the more mathematical and rigorous undergraduate degrees available. There is significant potential for an additional program that would address workforce requirements and provide a pathway for students who choose to be software developers as opposed to computer scientists. Curricula for applied bachelor's degrees are beginning to appear around the US; most of them called Bachelor of Technology. In this paper we describe the creation of the Bachelor of Applied Science in Software Development.

We know the job outlook for computer programmers and software developers is strong, growing at a rate of 12% and 30% respectively. [1,2] Anyone can examine the job requirements being posted online [3] and compare the requirements to the current academic curricula. The BSCS does not aim at the workforce, as it should not. However, graduates often have a hard time adjusting to software developer positions after completing a more theoretical science curriculum. [4]

The partner institutions have agreed that there is a difference between Computer Science and Software Development. Bachelor's degrees in computer science (BSCS) have emphasized the theoretical underpinnings of computing, including higher mathematics, algorithm analysis, the creation of operating systems, the theory of data storage and retrieval, etc. Industry advertises for employees who can work on teams, create interactive web sites, design efficient databases, front-ends, with knowledge of usability, testing, and current tools and technologies, and of course the main requirement of several years of experience. [1] The argument has been that the graduates of the theoretical programs can learn these skills with "on the job training." Some universities have addressed the issue by adding some applied courses, or an applied track, within the BSCS, while retaining the ABET accreditation requiring calculus and natural science courses. We arrived at this question: "Is there room for, and a requirement for, an applied bachelor's degree in software, in parallel with the more scientific Bachelor of Science degree."

1.1 Associate of Arts vs. Associate of Science

In Florida, more than half of high school graduates entering higher education attend the State College System (formerly the State Community College System). Arriving at the college, students considering a degree in computing choose between Associate of Arts (AA) and Associate of Science (AS). The AA degree prepares the student to enter the third year of the BSCS, and requires all general education for the BS degree to be completed in the first two years. This includes calculus, physics, natural science, social science, communication and humanities. In the first two years on this path, students can take a maximum of two or three courses in computing. The AS degree in Computer Programming is a workforce development degree including many technical classes in programming languages, web development, applied database, etc. and about six courses in general education.

Both of these paths have pros and cons. The 2-year AS degree does not articulate to the BSCS but it is a shorter path to the workplace than a bachelor's degree. The AA to BS articulation has existed in Florida law since the 1970s. Until recently the AS degree has been terminal and considered "vocational." The notion that vocational or workforce development programs are all two years long is deeply engrained. Most employers require a bachelor's degree and although the graduates of the AS Computer Programming and Analysis (AS CP&A) degree have been successful at obtaining employment, they have not always been able to obtain the career they seek and the lack of a bachelor's degree limits their advancement.

The AA program has a tendency to discourage students who do not have the willingness or capacity to complete calculus, physics, and chemistry / biology, before their first computing course. Students who had taken some computing in high school and choose to study computing in college are forced into general education when they would prefer to be studying computing. Two years is an eternity for college freshmen. Students without any computing classes in high school frequently find out well into their education they don't even like computing. This is something that would be better discovered in the first or second semester. Both situations result in a low retention rate for AA students who are aimed at the BSCS.

1.2 Academic Partners

In 2009, four colleges in central Florida, and the University of Central Florida (UCF) formed a coalition to address this issue. All of the community college partners have been transformed into State colleges during this project.

- Valencia College
  (formerly Valencia Community College)
- Seminole State College
  (formerly Seminole Community college)
- Eastern Florida State College
  (formerly Brevard Community College)
- Lake-Sumter State College
  (formerly Lake-Sumter Community College).

The National Science Foundation's Advanced Technology Education (NSF ATE) program funded the efforts to strengthen and align the AS Computer Programming degrees in central Florida and to create a new AS Computer Programming to BAS Software Development educational pathway.

Although it is now possible for the Florida State Colleges (formerly Community Colleges) to offer bachelor's degrees, the number of graduates of the AS Computer Programming (fewer than 100 per year combined) did not provide the critical mass required to sustain a bachelor's degree within one college, and surely would not sustain multiple bachelor's degrees in Software Development within 100 miles. The decision to combine these four colleges' efforts was encouraged by the agreement previously established called "Direct Connect to UCF." That agreement guarantees the graduates of the four partner-colleges admission to UCF. The value of an Associate's degree greatly increased with that agreement, and the completion rates also increased.

UCF is operating at, or near, the capacity of the main campus. A new division within the university called "Regional Campuses" has emerged to help provide access to UCF, outside of the main campus. Some programs are offered exclusively at Regional Campuses, including the Bachelor of Applied Science. UCF offices and classrooms are appearing on the campuses of the State Colleges in shared use facilities. Students who attend the four partner state colleges can visit the admissions offices and advisors for UCF without leaving their State College campus. Considering that this has occurred in the public higher-education domain, all of these changes have occurred with dizzying speed.

2 The Curriculum

The curriculum is driven by industry partners. Every company that was approached has agreed to serve on the Industry Advisory Board (IAB). Large and small employers from the main industries in central Florida are represented on this board. Tourism, defense industries, entertainment, health care, space, are among the highly diverse employers of software developers in this region. All of them are represented, and we made some attempt to avoid a single industry from driving the curriculum content. By insisting the IAB consist of current practitioners in software development, there has been very little disagreement. A 2-day DACUM was held. Two or three IAB meetings per year allow us to ask the local industry what would be the most useful curricular topics. The messages from industry were unmistakable, and after examining existing curricula in software development, it was clear that a BAS curriculum articulating from the AS in Computer Programming would be unique. The recommendations included both broad and specific issues. The three general requirements are that the program must produce graduates who:

A. *Work well on teams.* This includes communication skills, and experience using tools and techniques that facilitate software development in teams.
B. *Have significant depth of capabilities.* This drove the decision to develop a progressive series of courses, where the same topics, and the same programming language, are carried from course to course, increasing in size and complexity, culminating in a capstone course called Software Development Project.
C. *Understand the importance of the business context* in which the software development process exists. For almost all

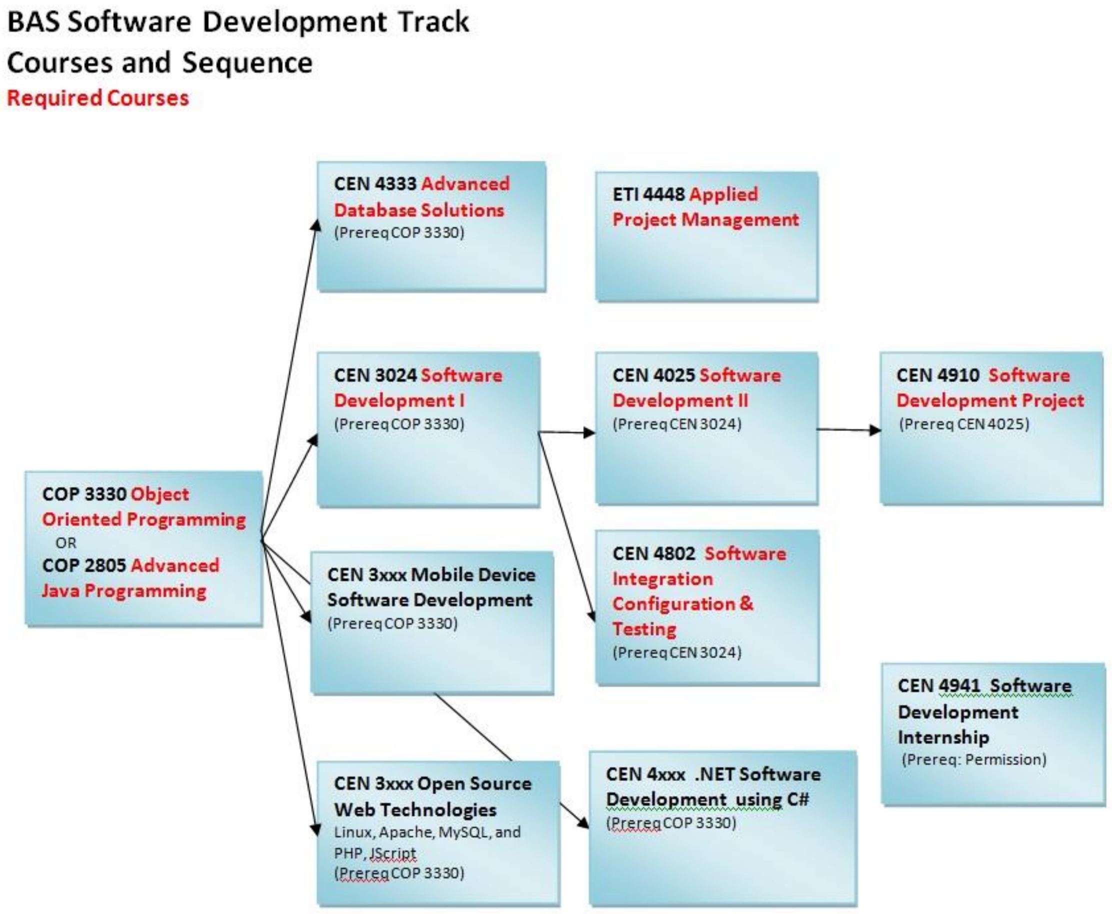

BAS Software Development Track
Courses and Sequence
Required Courses
CEN 4333 Advanced Database Solutions (Prereq COP 3330)
ETI 4448 Applied Project Management
CEN 3024 Software Development I (Prereq COP 3330)
CEN 4025 Software Development II (Prereq CEN 3024)
CEN 4910 Software Development Project (Prereq CEN 4025)
COP 3330 Object Oriented Programming OR COP 2805 Advanced Java Programming
CEN 3xxx Mobile Device Software Development (Prereq COP 3330)
CEN 4802 Software Integration Configuration & Testing (Prereq CEN 3024)
CEN 4941 Software Development Internship (Prereq: Permission)
CEN 3xxx Open Source Web Technologies Linux, Apache, MySQL, and PHP, JScript (Prereq COP 3330)
CEN 4xxx .NET Software Development using C# (Prereq COP 3330)

Fig. 1 Required and elective courses on the Software Development track.

companies, software does not have a purpose outside of the context of their business.

The AS in Computer Programming is 63 credits. The BAS at UCF is 120 credits. Students take the 63 credits to UCF as a block, and additionally can take up to 15 credits from the college to the university as general education or electives. A minimum of 42 credits must be taken at UCF. They are not usually the last 42 hours the student completes, and it is common that students are taking courses at both institutions in the same semester. Details of the AS Computer Programming degree can be examined online. [5] The AS degree produces *programmers*, and the BAS produces *developers*. The difference between a programmer and a developer is somewhat arbitrary. A *programmer* is an expert in languages and coding, but somewhat less prepared for software design and business requirements analysis. The *developer* also has leadership roles within software groups. This roughly coincides with what the academic programs can achieve in two years, and four years respectively.

The BAS program contains a *Core* and a *Track*. The Core courses include technical writing, business management, and ethics. The Track contains the new software development courses. 21 credits are required, and up to 12 elective credits in software. The details can be found in the UCF online catalog. [6] The sequence of courses is shown in Fig. 1. The courses with XXX are elective courses and are currently in the state course-numbering process. The seven courses with course numbers are the 21 credits required for the SD track. The Internship is elective. Although it seems that this would be valuable as a requirement, almost all of the students obtain employment in the field before they complete their degree, and are completing the program on a part-time basis.

All of the new software development courses in the BAS were created at a week-by-week level of detail, and the design assumes it will only ever be delivered online. For the creation of the course content, a professor is paired with an industry representative to ensure the course material is reasonable from both the perspective of pedagogy, and industry requirements.

The courses are modified somewhat during the delivery, by the professor leading the course. These modifications are based on the feedback from the students, and the bi-annual advisory board meetings.

3 Progress in Online Education

Some of the partner colleges offer all of the AS degree online. All of the courses required for the BAS are available online. Most are available exclusively online. The students remain living at home, and do not have to commute. The pros and cons of online higher-education are discussed widely in the literature. All of our students are software development majors, and are already comfortable interacting online. We are providing online communication mechanisms to mitigate the disadvantages of not being physically in the same room. These tools should be familiar, and without significant cost to either the university or the students.

3.1 Facebook Group

The first mechanism created was a Facebook group. It was publicized before the program was actually offered, and we invited students to congregate in this online group to discuss the creation of the curriculum. There are currently more than 225 participants in this group. They are mostly students, but some faculty and advisors are participating. The original objective of discussing the curriculum was not the most valuable outcome from this group. As the students looked at the curriculum, they discussed other educational pathways they had discovered. The pros and cons of each option were discussed openly.

As the students transitioned to UCF from the colleges, we discovered the articulation was not exactly seamless. Although UCF guarantees admission to the graduates of the colleges, the requirements to transition were unfamiliar to the students, and even a clear description of the transition was not always interpreted correctly by the online students. When students got stuck in the transition process, we knew on the same day because of the Facebook discussions. The students frequently solved the other students' problems without need for faculty or advisors. The transition needs to happen very quickly. Admission to UCF cannot happen until the AS degree appears on the students' transcript, and only then can the student request their transcript be sent to UCF, attend the online orientation, obtain proof of immunization, and some must deal with financial aid issues, veteran's administration issues, and finally, register for UCF courses. If a student graduates in December and needs to attend classes in January, there is a risk of losing an entire semester because the administrivia cannot be completed on time as the offices are closed for Christmas holidays. This situation is improving dramatically because it is discussed openly on the Facebook group. Everyone involved knows immediately when there is a snag. UCF admissions and advising personnel have responded and the loss of a semester due to "process" is now quite rare.

3.2 Skype

Advising one-on-one at a distance is beginning to be implemented using Skype. All of the students at the partner colleges are issued a Microsoft Windows Live email address. Since Microsoft purchased Skype, this has provided a standard way to explain to students how to contact their advisor using a webcam. Surprisingly, the advisors are more difficult to get onboard with this mechanism than the computer programming students. It is a cultural shift, and cannot be forced; only encouraged.

Online student engagement hours (office hours), and supplemental instruction are also being introduced at the partner colleges using Skype. This varies among colleges and the individuals involved. Most colleges have a tutor available for the students in the AS courses. Making tutoring available for online students is a challenge, but not nearly as difficult as convincing the faculty their engagement hours should be accessible online when teaching online classes. Screen sharing with the student's computer is often a better way to discuss the snags in learning computer programming than an in-person visit to a professor's office. It is not possible for the student to describe what went wrong while creating a program. If they knew what went wrong they could simply fix it. It's very easy to talk through a problem while watching the student work. This is true for both professors and tutors.

Fig. 2 Professor meeting with four students in a Google+ Hangout.

As we move toward this more successful mode of engaging the students, we are frustrated with the pace of the cultural shift within the colleges. For example, professors must drive to their on-campus office to engage online students using Skype. There is no way to force the realization that this is not efficient.

3.3 Google+ Hangouts and YouTube

During the class, much of the material is communicated through recorded video lecture supplementing the material from the textbooks, and keeps the students comfortable that they are engaged with their instructor. Synchronous communication with groups is achieved via Google+ Hangouts, and the Learning Management System, currently Canvas. The students have not had trouble using this technology. Presentation of project results is done using screencasts. Students record the presentation of their results, describing their documents and software in a video. The screencast is uploaded to *YouTube*, and the link is submitted to the professor, and sometimes to the other team members for discussion. Students have been finding their own screencast recording tools, and there are several free ones available. This simulates a live presentation, but doesn't fully capture the experience of making a presentation in front of a group.

4 Measuring the Impact

The first measurable impacts of the new pathway are on the AS Program. Three questions were asked of the available data from the partner colleges. *Question 1*. What is the change in the number of declared majors in the AS Computer Programming? A desirable pathway to a bachelor's degree should increase the number of students who choose this AS program. *Question 2*. What is the impact of the new program and other project activities on retention among students who are declared in the AS CP&A? This was done using cohorts of students who are newly declared in this AS. Studying the data is only the first step toward understanding how to make an impact on retention. *Question 3*. What is the change in the number of students graduating with the AS Computer Programming? We need to know if the number of graduates and the pipeline of declared majors forms the critical mass required for a successful bachelor's program. These questions have formal definitions so each partner college can count in exactly the same way.

The first declared majors in BAS SD were registered at UCF in the fall 2012 semester. The base year for all data is 2009-10 academic year. This is the last year the BAS was not being discussed with the students.

4.1 Question 1

The number of AS CP&A declared majors has been increasing quite dramatically since 2009. There are currently more than 1000 students in the AS CP&A. Fig. 3 includes only students who are registered for classes in the reporting year, as opposed to including the "stop-outs." This is a sufficient number to support a bachelor's program needing around 50 new students transferring to UCF each year.

	AS CP&A Majors (Valencia+Brevard+Seminole)				
	2009-10	2010-11	Change	2011-12	Change
Valencia	304	402	32.2%	439	9.2%
Brevard	243	322	32.5%	455	41.3%
Seminole	146	234	60.3%	171	-26.9%
Total	693	958	38.2%	1065	11.2%

Fig. 3 Change in the number of declared majors in the AS Computer Programming program.

4.2 Question 2

Retention is measured by collecting a cohort of students who are newly declared to be in the AS CP&A program in each fall semester. Each cohort is followed to the next spring, and the next fall. If the students are still declared in this program and taking classes, they are counted as retained.

Fig. 4 shows the data for the Fall – Spring retention for three cohorts. Getting through the first semester, and attending classes in the second semester is the largest single place where we can measure students leaving the program. With data combined from the three colleges, around 25% of the cohort does not reappear as registered AS Computer Programming students in the second semester.

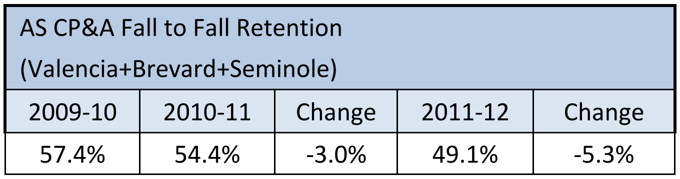
AS CP&A Fall to Spring Retention (Valencia+Brevard+Seminole)				
2009-10	2010-11	Change	2011-12	Change
75.0%	76.6%	1.6%	72.7%	-3.9%

Fig. 4 The one semester retention rates for 3 cohorts, combining data from all three colleges.

The change in the retention rates is disappointing to some, and as usual, data tend to raise more questions. These retention rates are not bad when compared to other academic programs that have a significant quantitative component in the curriculum. There are multiple causes for the retention rates not improving. The increase in enrollment probably impacts this, as well as local economic factors, combined with the availability of student financial aid. Currently we are looking into the success rates in the first course in the AS curriculum; COP1000 Computer Programming Concepts. Hopefully the correlation between the relative success rates at the different colleges will provide some actionable information. The one year retention rates, measured fall-to-fall are shown in Fig 5. Only about one half of the students from the fall cohort are enrolled in the following fall, and still declared to be in the AS Computer Programming. We don't have empirical explanations for why this is happening, but at least we know what needs to be explained.

4.3 Question 3

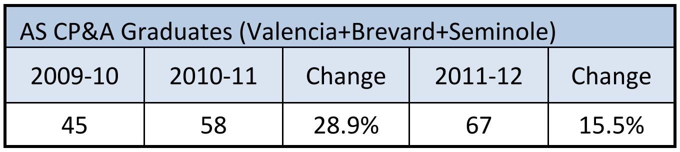
Graduates from the AS CP&A are increasing. The total of 67 graduates in academic year 2012-13 is approaching the critical mass required to achieve a sustainable BAS program. This completion rate is still very low. If we compare these numbers to the number of declared majors in the AS CP&A, it appears approximately 10% of the number of declared majors are completing two years later. With 693 declared majors in 2009-10, and 67 graduates in 2011-12, we can estimate around 90 graduates in 2012-13, and 100 in 2013-14. This is speculative, but observers are generally accepting this is a conservative estimate. Fig. 6 shows the trend in AS Computer Programming completion.



AS CP&A Fall to Fall Retention (Valencia+Brevard+Seminole)

2009-10	2010-11	Change	2011-12	Change
57.4%	54.4%	-3.0%	49.1%	-5.3%

Fig. 5 The one year retention rates for 3 cohorts, combining data from all three colleges.



AS CP&A Graduates (Valencia+Brevard+Seminole)

2009-10	2010-11	Change	2011-12	Change
45	58	28.9%	67	15.5%

Fig. 6 The number of graduates from the AS CP&A program combining results from all three colleges.

5 Conclusion

Computer Science departments are under some pressure to increase capacity to meet demand [7], and pressure from industry to produce a graduate prepared for the workforce, as opposed to prepared for graduate studies. We addressed these by creating an alternate educational pathway that does not weaken the BSCS, but instead provides a completely separate option aimed at the students who are interested in an applied program of study. The program is being offered online to allow students to remain in their current homes, and to set a study schedule that meets their personal requirements.

The first cohort will graduate in the Spring of 2014. It is too early to have statistics on retention and completion in year three and four of the BAS degree. We can observe that many of them have already accepted employment in the field simply on the basis of the AS degree and being enrolled in the BAS Software Development program. Providing an online, largely asynchronous program encourages these students to *merge* into the software development field.

With a total of more than 1000 students attending the AS, and 67 graduates in 2011-12, it is evident that there are enough students in the pipeline to support the BAS Software Development program, even though we have some serious issues to address in retention and completion. With three years of comparable data from three colleges, we believe that these issues can be addressed, and this work has already begun.

6 References

[1] US Department of Labor, Bureau of Labor Statistics, Occupational Outlook Handbook: Computer Programmers, March 2012.

[2] ibid, Software Developers.

[3] SimplyHired.com

[4] "Struggles of New College Graduates in their First Software Development Job", Andre Begel, Beth Simon, Proceedings of SIGCSE 2008.

[5] Details of the AS Computer Programming. http://valenciacollege.edu/asdegrees/information-technology/documents/Valencia-College-Computer-Programming-and-Analysis.pdf

[6] Details of the BAS Software Development. http://catalog.sdes.ucf.edu/academic_programs/pdf/applied_science_articulated_as_to_bas_track.pdf

[7] Computing Research Associates Taulbee Survey, 2011-12 http://cra.org/govaffairs/blog/wp-content/uploads/2013/03/CRA_Taulbee_CS_Degrees_and_Enrollment_2011-12.pdf

Analysis of suitable languages to teach Procedural Programming at the Universidad Nacional del Noroeste de la provincia de Buenos Aires

Germán Osella Massa, Cecilia De Vito, Claudia Russo, Mónica Sarobe, and Sabrina Pompei
Escuela de Tecnología – Instituto de Investigación y Transferencia de Tecnología,
Universidad Nacional del Noroeste de la provincia de Buenos Aires (UNNOBA),
Roque Sáenz Peña 456. Junín, Buenos Aires, Argentina.

Abstract - *Based on the need to update the language and tools used to teach Procedural Programming during the first year of computer science careers, a cross descriptive research was conducted by assessing different programming languages. The analysis involved a total of twelve languages with different characteristics. A new methodology was proposed based on the conclusions obtained from this work. Teachers have considered this proposal highly appropriate and it has been implemented from the beginning of 2013.*

Keywords: Programming Languages, Teaching, Comparison, Methodology.

1 Introduction

Both the Escuela de Tecnología at UNNOBA and the staff in charge of teaching Procedural Programming to first year students in the Computer Science courses saw the need to rethink the methodology that was being used. The use of PASCAL as an introductory language to teach fundamental concepts has given rise to an increasing number of difficulties that are needed to be solved. Some of these issues are related to technical difficulties (i.e., compilers and IDE for obsolete platforms, lack of modern support tools, etc.) whereas others are related to the way in which students see this language. Therefore, it was decided to conduct a study to compare a set of twelve programming languages with the aim of identifying their strengths and weaknesses as regards teaching procedural programming. In the rest of this article, the reasons that led to discard most of the languages assessed as a tool to teach procedural programming will be mention and the new methodology designed based on the chosen languages will be analyzed. Finally, the results expected from the change will be described.

2 Analyzed and discarded languages

The languages under analysis were C, C++, C#, D, Java, Go, Javascript, Objective-C, Pascal, Python, Ruby and Scala. They all have different characteristics: some of them are procedural languages, others are object-oriented and some are even multi-paradigm ones. They can be divided into interpreted and compiled languages. Some of them are quite new (Go was announced in 2009) whereas others have made history (such as C). In this section, the reasons why most of the languages were discarded will be discussed.

2.1 Pascal

Although Pascal was a popular structured language designed to teach the fundamentals of procedural programming, it unfortunately has fallen into relative disuse (both in the industry and in academia) and, as a result, there are no modern support tools (integrated development environments and debuggers) or any active communities promoting it. Delphi [12] and FreePascal [13] are exceptions which have found their niches and are still being developed. Delphi, however, as it is a commercial product, has certain constraints that make it less attractive: it is not multiplatform, it requires relatively new hardware to be usable and there is an associated cost per use license. On the other hand, FreePascal does not present any of these inconveniences and the Lazarus project, which seeks to implement an environment similar to Delphi, could be a convenient development environment. However, students believe that they will never use Pascal and they receive the same message from senior students. This causes demotivation and lack of interest in the language thus having a negative impact on the teaching of the curricula. The use of a language that the students consider more attractive, either because of its commercial application or because of its popularity, can boost interest in learning it and, in consequence, all the topics involved in the courses.

2.2 C

C [1] is a relatively small and very popular system programming language. It supports mostly all of the procedural programming topics required, although sometimes it turns out to be a very low level language, causing concepts to become blurred by this. To mention an example, concatenating two strings of characters becomes an extremely complex and prone to error task, involving pointers, dynamic memory allocation and tests to avoid memory overflows.

C makes it necessary to understand how memory is managed both by the operating system and the program. It allows writing code with errors that sometimes are very hard to detect and may cause the program to terminate unexpectedly. Therefore, C was discarded. It is probably more suitable to learn it along with an assembly language (where the distance between both of them would be relatively small and the concepts would be more related).

2.3 Objective-C

Objective-C [11] is a popular language on the Apple platforms: It is one of the core languages on Mac OS X and iOS (iPhone, iPad and iPod). This language is a hybrid of C and Smalltalk. It expands the C language adding classes and objects using the model proposed in Smalltalk. The difference between both styles of programming can be clearly seen in the following code since the latter is written in square brackets:

```
#import <stdio.h>
#import "Fraction.h"

int main( int argc, const char *argv[] )
{
  Fraction *f = [[Fraction alloc] init];
  [f setNumerator: 1 denominator: 3];
  printf("Fraction = %d\n", [f value]);
  [f release];
  return 0;
}
```

Unfortunately, as there is no need to teach the contents related to object oriented programming (the aim is to teach procedural programming only) what is left is basically C, which was rejected for the reasons discussed before.

2.4 C++

C++ [2], as an evolution of C, was considered the ideal language to teach programming, since it allows more advanced constructions to be used thus making it easier to carry out many of the tasks that would require considerable effort in C. Additionally, all the items added to C++ are optional: classes may be used or not, operators may be overloaded, and generic functions or classes may be defined but are not mandatory, etc. The philosophy has always been that if something is not used, no additional cost is paid (in memory or speed) for the possibility of using it. However, C++ is a huge language and it was not designed to be used as a first programming language. The fact that C++ supports the C language almost entirely and also adds more concepts such as classes, operators, overload, name spaces, generic programming, templates and so on, makes it a huge language. Explaining the following trivial program to an inexperienced student is not a simple task:

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6    cout << “Hello, world!” << endl;
7    return 0;
8  }
```

The first line requires an understanding of header files and the preprocessor (both inherited from C). The second line involves the concept of name spaces. Line 4 requires explaining what a function is and why it is necessary the returned value in line 7. Line 6 involves so many concepts that it is difficult to list them without leaving some of them out: `cout` is an instance of the class `std::ostream`, “<<” is an overloaded operator to work with “`char *`” and explaining the mechanism behind “`endl`” is certainly not an easy task. One can obviously follow the *“Do not worry, you will understand it later”* and *“for now, just type it and it’ll work”* approach, but this is not the best teaching strategy since it causes confusion and frustration to students when something does not work as expected. For all this, C++ was initially discarded for been too complex. Nevertheless, now that C++11 is becoming more mainstream and the new features it introduces really make a difference in the way of using the language, this decision is been reconsidered.

2.5 Java and C#

Java [8] and C# [3] are two very similar languages of widespread use. They are both, multiplatform, using a virtual machine to run on different operating systems. In the case of Java, Oracle provides virtual machines that runs on Windows, Linux, Solaris and OS X. In the case of C#, Microsoft provides the .NET platform (for which C# compiles) which is only for Windows, but there is a project called Mono [4], which seeks to develop a .NET compatible platform (if possible) and provides versions for Windows, Linux, Solaris and OS X, and even making it possible to compile for iOS and Android. The main disadvantage of these two languages is that they only support the object-oriented programming paradigm and they almost reject the procedural paradigm altogether. It is not possible to define an independent function in any of them; methods (the closest thing one can get to a function) can only be defined inside a class. This leads to an overload of unnecessary structures in a program where they only aim is to teach procedural programming, as seen in the following examples of a "Hello world” program written in Java and C#:

Java

```
package hello;

public class HelloWorld {
  public static void main(String[] args)
  {
    System.out.println("Hello, world!");
  }
}
```

C#

```
using System;

namespace Hello {
 public class HelloWorld {
  public static void Main(string[] args)
  {
    Console.WriteLine("Hello, world!");
  }
 }
}
```

The definition of a class, a method for a class, visibility modifiers for that method and the use of an instance method and a class to be able to produce an output do not contribute anything to the teaching of procedural programming. In addition, Java lacks pointers and C# supports them but their use is restricted. As a result, both languages were discarded for the teaching of procedural programming.

2.6 Scala

Scala [16] is a language designed to program in a concise, elegant and secure way (it is said to be secure because of its static typing). It incorporates certain features of object-oriented and functional languages and it can inter-operate with Java in a transparent way. However, the procedural paradigm is not one of its strong points. The most concise way to write a "Hello world" is as follows:

```
object HelloWorld {
  def main(args: Array[String]) {
    println("Hello, world!")
  }
}
```

As compared to Java, the number of concepts decreases notably and despite of been a more compact code, it is equivalent. In fact, if we were to choose between Java and Scala as a base language, the latter would be a very interesting option. Unfortunately, it is clearly an object-oriented language (where it is impossible to define a function other than an object or a class) making it impossible to make use of it as a procedural language.

2.7 Go

Go [7] is a relatively new language (announced in 2009), created and sponsored by a group of developers from Google. It aims at being a simple language but adds features such as automatic memory management (using garbage collector) and concurrent programming support (introducing the goroutines concept). It moves away from the syntax and control structures proposed by C, and it suggests simpler and more powerful alternatives. It supports object-oriented programming but does not use the classic class-based model that is common in other programming languages.

```
package main
import "fmt"

func main() {
  fmt.Println("Hello, world!")
}
```

The type system in Go uses the concept of *interfaces*, where if a type defines a certain set of methods, such type will automatically define and implement an interface (whether or not that was desired). However, it lacks generic types, which can be a constraint. On the other hand, the development tool support is poor and there is practically no literature. Although this language has recently reached its 1.1 version, which marks a certain level of stability, it is still really new and not altogether mature to be used as a base language.

2.8 Ruby

Ruby [15] is a very popular dynamic language frequently used on web development. Unfortunately, it has certain features that do not make it eligible as a procedural language. Firstly, the fact that it is purely object-oriented cannot be overlooked (for example, the suggested way to iterate on a sequence is by using one of the variants of the “each” method, on which a block will process each of the elements in the sequence, just as in Smalltalk but far from the way in which this is done in a procedural paradigm). It also has many alternative ways of writing exactly the same thing (there is if and unless, while and until) all of them as control structures or statement modifier). Some examples of Ruby, showing in each column variants of exactly the same code are presented next. Having such an expressive language allows expert programmers to write a clear and natural code. However, inexperienced programmers will be confused and doubtful about the right way to write code.

```
if test then              |  unless test then
   puts("Yes")            |     puts("No")
end                       |  end
                          |
                          |
if test                   |  unless test
   puts("Yes")            |     puts("No")
end                       |  end
                          |
                          |
puts("Yes") if test       |  puts("No") unless test
```

2.9 JavaScript

JavaScript [9, 10] is the de facto language on the Internet. Most current browsers (if not all of them) support it. However, it has certain features that do not make it an ideal language to start programming. In general, although there are exceptions, it lacks input/output functions (as it is intended to be run on a browser, the ability to read and write files in an arbitrary manner entailed an unnecessary risk). It also shares one of the problems of Ruby; it is hard to disregard the fact that you are working with objects. The type system has implicit conversions that more than once catch programmers unprepared (expert and novices alike). The result of the following expressions is almost unpredictable if the rules governing conversions among different types are really well known.

```
> 1 + 2            |  > 1 + "2"
3                  |  "12"
> []               |  > [] + 1
[]                 |  "1"
> [1, 2] + 3       |  > {}
"1,23"             |  {}
> {} + 2           |  > [] + []
2                  |  ""
> {} + []          |  > [] + {}
0                  |  "[object Object]"
> {} + {}          |
NaN                |
```

Where `[]` *is an empty array and* `{}` *is an object literal.*

The greatest advantage that JavaScript could offer would be its ubiquity and its almost immediate business application. However, these two reasons are not enough to choose it.

3 New methodology proposal

At the UNNOBA, the first contact that students have with procedural programming is during the first term of the first year in the Computer Sciences courses, when they take the subject "Introducción a la Programación Imperativa". They then take "Programación Imperativa" in the second term. Based on this division, our proposal is to use different languages in each subject, each of them with a different approach. First, the idea is to start with a very high level programming language, which interferes as little as possible with the teaching objectives when starting to program (simple data types, clear control structures, basic input/output) and which can help students acquire good habits when writing code (consistent indentation, documented code, use of test cases). Having an interactive session to test code and obtain an immediate response also turned out to be a very valuable feature: Interaction with an interpreter provides an immediate response, helping the student to experience with language and be able to form a mental model to explain the obtained response. Compiled languages need a debugger to be able to carry out a step-by-step trace and the necessary skills to be able to use it. This discourages students from exploring it. **Python** turned out to be a language which met all these requirements and it will be used for students to learn the fundamentals of procedural programming. In addition, it is essential for students to be exposed to concepts that, due to their nature, cannot be easily stated or are simple impossible using Python: Compiled vs. interpreted language, static type checking, pointers and, static vs. dynamic memory management, just to mention a few. To solve this problem, we resorted to another programming language where the student can be in contact with these concepts. In this case, the decision was harder. Several languages that could meet these requirements to different extents were assessed. Finally, the language selected for this was **D**, which clearly covers all these concepts and more. The features that led us to choose Python and D are discussed in the following subsections.

3.1 Python

Python [14] is a clear and minimalist language which seeks to embrace the *"There should be one –and preferably only one– obvious way to do it"* motto. Using indentation to define blocks of code forces students to acquire the good habit of writing clear code. This habit is expected to be transferred to other languages where blocks are written using delimiters. Although Python is an object-oriented language, it can easily hide it and appear as a completely procedural language. In addition to being a very high-level language, at first, students will need to handle only a limited number of concepts. It is a strongly typed, dynamic and interpreted language. Interactive sessions can be run with helps such as online documentation and code completion. The online documentation system is applicable both on a built-in function of the language or on one written by the students. Finally, test cases, known as *doctests*, can be written in the documentation, using a syntax that is identical to that of the interactive session. The following function has documentation and test cases contained in the same documentation.

```
def factorial(n):
    """
    The factorial function, n!
    >>> factorial(1)
    1
    >>> factorial(2)
    2
    >>> factorial(4)
    24
    """
    if n > 1:
        return n * factorial(n - 1)
    else:
        return 1
```

3.2 D

D [5, 6] is compatible with a large subset of C (similar syntax, compatible types, same control structures) and just as C++, it increases C by incorporating object-oriented and generic programming, but it does this in a simpler and more organized way, learning from the mistakes made in C++. It does not try to be absolutely compatible with C, including associative maps and dynamic strings as primitive types, automatic memory management (with garbage collection) and a more robust type checking than the ones provided by C. D incorporates some of the concepts of the object-oriented and functional paradigms but it does not turn its back to its procedural programming roots. It supports assertions and test cases in a native manner, as follows:

```
int add(int x, int y) {
  return x + y;
}

unittest
{
    assert( add(1, 2) == 3 );
    assert( add(-7, 3) == -4 );
}
```

The code within the `unittest` block will be executed only in a special test mode and it will be ignored in normal execution. With this feature, it is easy to incorporate test cases for the written code, just as in Python, thus fostering the habit of testing programs this way. Another related feature is programming by contract, with preconditions and postconditions of a function:

```
double func(double x)
in {
    assert( x > 0);
}
out(result) {
    assert( result >=0 && result <= 1 );
}
body {
    // func implementation...
}
```

The function `func` receives the `x` parameter which is a floating point value, and returns another floating point value. The code within the `in` block will verify that the value of `x` is always higher than 0 (otherwise, it will fail), then the body of `func` will be executed and finally, in the `out` block, the value returned by the function (stored in `result`) will be verified to check whether it is within the interval [0, 1]. A problem that usually shows up when implementing a data structure (a linked list or a stack, for example) is the type of the values stored in this structures. In the case of linked lists, an implementation defining the type of the value stored in each node as `int` will not work to create a list of strings or floats or anything else. In D, this is solved using generic types (there are similar mechanisms in C++, Java or C#). A generic definition for a node of a simple linked list can be written as:

```
struct Node(T) {
   T    value;
   Node *next;
}
```

The structure now has a parametric type T, which can be instantiated as follows:

```
Nodo!int    n1;
Nodo!double n2;
Nodo!string n3;
```

The three definitions above will result in three different structures (`n1`, `n2` and `n3`), each of them specialized with the type indicated after the exclamation mark. Functions and methods can also be written in this way, making possible to implement truly generic algorithms. D has other interesting features that are not discussed here for the sake of brevity.

4 Expected results

The new methodology that is been implemented in 2013 is expected to motivate students of the courses "Introducción a la Programación Imperativa" and "Programación Imperativa", bringing fresh air to the way in which knowledge is acquired. Python is a very popular language, with many practical uses ranging from the web to desktop applications or games. In addition, there are modern tools that are expected to make it easier for students to learn the contents taught in each subject. In particular, Online Python Tutor [17] is expected to be together with Python's interactive shell an invaluable tool to understand the control flow of a program. D will be used to apply the concepts that were formerly described on paper or on a concrete language but with restrictions that were artificial imposed. All these changes will also have an impact on "Estructuras de Datos", a second year course which requires the students to pass both "Introducción a la Programación Imperativa" and "Programación Imperativa". Plans are made to use D as the language where the structures and algorithms will be studied and implemented in a really generic manner.

5 References

[1] Kernighan, B., Ritchie, D.: "The C Programming Language". Prentice Hall; 2nd edition (1988)

[2] Stroustrup, B.: "The C++ Programming Language". Addison-Wesley Professional; 3er ed. (2000)

[3] The C# Language Specification, http://go.microsoft.com/fwlink/?LinkId=199552

[4] Mono: Cross platform, open source .NET development framework, http://www.mono-project.com/

[5] D Programming Language – Official Website, http://dlang.org/

[6] Alexandrescu, A.: "The D Programming Language." Addison-Wesley Professional; 1st ed. (2010)

[7] The Go Programming Language – Official Website, http://golang.org/

[8] Java official website at Oracle, http://www.oracle.com/technetwork/java/index.

[9] Flanagan, D.: "JavaScript: The Definitive Guide". O'Reilly Media; 6th edition (2011)

[10] Crockford, D.: "JavaScript: The Good Parts". Yahoo Press; 1st edition (2008)

[11] The Objective-C Programming Language, http://developer.apple.com/documentation/Cocoa/Conceptual/ObjectiveC/

[12] Embarcadero Delphi XE2, http://www.embarcadero.com/products/delphi/

[13] Free Pascal – Advanced open source Pascal compiler for Pascal, http://www.freepascal.org/

[14] Python Programming Language – Official Website, http://www.python.org/

[15] Ruby Programming Language – Official Website, http://ruby-lang.org/

[16] The Scala Programming Language – Official Website, http://www.scala-lang.org/

[17] Online Python Tutor, http://www.pythontutor.com/

Advanced Java Features for Using JUnit in CS1/CS2

Hans Dulimarta Scott Grissom
{dulimarh,grissom}@gvsu.edu
School of Computing and Information Systems
Grand Valley State University
Allendale, Michigan 49401

***Abstract*—The Computer Science Curriculum 2008 [2] (CS2008) published jointly by ACM and IEEE includes "software testing" as one of the core topics in several areas of knowledge (Human-Computer Interaction, Social and Professional Issues, and Software Engineering). Out of 31 core hours of Software Engineering knowledge, CS2008 recommends a minimum of 3 hours dedicated to Software Verification and Validation. It is important that software testing is introduced early in the curriculum and we believe that CS1 is an appropriate course to do so. The first exposure to using testing tools in CS1 will prepare students to continue using automated testing tools when they work on larger projects in future classes.**

In this paper, we describe a few techniques in designing JUnit test cases to address some challenges commonly encountered in CS1 programming assignments. Some of these techniques are also applicable to programming assignments in CS2.

***Index Terms*—Unit Testing, Introductory Course, Test Case Design**

I. What is JUnit?

As a unit testing framework for Java programs, JUnit [6] provides a large set of assertion methods for comparing the actual value of variables obtained from the program under test with their expected value. In addition to this main feature for asserting various conditions, JUnit also provides powerful and extendable features that have attracted professional software developers as well as computer science educators to use JUnit in their software development.

Test cases developed as supplemental material to a programming assignment can benefit both the instructor and the students. The instructor will feel confident that the solution to the assignment is correct and the test cases can later be used to automate most of the grading[4]. Likewise, students will have the assurance that the work they submit satisfy the assignment requirements.

As one of the most popular choices for unit testing framework, JUnit has been adapted to support more advanced features for software testing. Szeder [5] implemented a unit testing framework for multithreaded Java programs by integrating JUnit with Java Path Finder. All the possible scheduling of threads are first discovered using Java Path Finder and JUnit test methods are then applied to each thread schedule. Artho and Biere [1] developed JNuke that integrates automatic test coverage measurement based on the result produced by GNU C compiler utility `gcov`. This approach enables JNuke to detect untested or dead code.

The rest of this paper is organized as follows. The next section describes the common challenges faced in using JUnit for testing programming projects assigned to CS1/CS2 students. In Section III we briefly describe how our CS1 and CS2 courses are administered and how projects are assigned. In Section IV we present relevant assignments where the above challenges appeared. By presenting these assignments, we hope that readers will be able to adopt the techniques for their own courses. Section V presents the advanced Java features we used in conjunction with JUnit assertion methods to address the above challenges. In section VI, we describe some of the issues encountered by our students when they ran our tester code. These issues served as feedback for improvement in designing our test cases. And finally, we conclude with new insights we learned from using JUnit in our CS1 and CS2.

II. Challenges

A. Capturing Standard Output

When a class is designed to include the typical setter and getter methods, writing a JUnit test suite for this class is straight forward; the test cases will mainly consist of test methods that call one of the

″setter″ methods followed by one or more assertions on the corresponding “getter(s)”. However, during the first few weeks of a CS1 course it may be too early for students to understand the concept of returning a result from a function. In fact, first-time programmers may (mis)understand that “returning a result” means printing the result to standard output. Consequently, the first few programming assignments in the semester are typically designed to include methods that print the results via standard output. This approach allows students to immediately verify the results. However, designing automated testing for this kind of assignment is not straight forward.

B. Generating Random Input

Programming assignments that require conditional statements are usually designed to include methods that process varying input data. Based on the range of input values, these methods are supposed to produce different output. A common approach for handling this situation is to require the student program to read input from a file, but it may add extra complexity of using files and handling input/output exceptions. To prepare such an assignment, the instructor will have to prepare both the input data and the expected output for each corresponding input. An alternative technique is to read data from standard input. The JUnit tests for this kind of assignment must be designed to “inject” input data into the student’s program standard input. In addition to injecting input data, the instructor’s tester code can also generate the expected result for the corresponding injected input.

C. Escaping Infinite Loops

Incorrect implementation of a programming assignment that requires loops may contain bugs that cause a method to run in an infinite loop. Consequently, a JUnit test method that attempts to verify the correctness of the method may get trapped inside the infinite loop. These loops must be detected and escaped.

D. Testing Extra Credit

Instructors occasionally offer extra credit or programming assignments are designed to allow students to implement additional features. Testing the additional features poses a design challenge for the instructor. If the test cases for the extra features are hardcoded into the test file, it may generate compile errors for students who decide not to implement the extra features. One solution is to provide a separate test file for testing only the extra features, but this may require extra setup on the students’ part as well as maintenance overhead for the instructor. A better solution is to use Java reflection to detect at runtime whether a student’s implementation includes the extra method(s).

E. Setting Up Preconditions

It is quite common to find methods that require a precondition be true before we can verify that these methods satisfy the requirements. For instance, a method that draws cash from a bank account requires a non-zero balance. Setting up the precondition for such simple methods is straight forward, but other methods may require more involved setup. In a game of Chess, testing whether a King is “in-check” requires the chess board be setup by placing certain chess pieces at selected positions.

F. Discovering Class Information at Runtime

Java interfaces give programmers freedom to choose the name of the implementing class. For a programming assignment that requires students to implement instructor-provided interfaces, students may choose any name for their implementing class. As a consequence, instructor tester code cannot assume a particular class name will be used in students’ implementation; the class name must be determined at runtime.

III. Course Administration

Our CS1 was taught in Java with BlueJ 3.x [3] as the required IDE. Five projects were assigned in the semester, three of them were released with JUnit test cases. The first and last projects were ″do-it-yourself″ projects where each student was given the flexibility to design the final content of the projects, with some minimal features provided by the instructor; no unit testing was used for these two projects. Our CS2 was also taught in Java but we let our students use an IDE of their preference. Unlike in CS1 where test cases were provided by the instructors, our CS2 students are required to write JUnit test cases.

A team of several instructors taught multiple sections, and one of the instructors was assigned as the course coordinator. For our CS1, all projects were designed by the course coordinator, but the corresponding test cases were developed by a different instructor on the team. For our CS2, each instructor designed one project which later shared by the team.

Having a different test developer helps to reduce ambiguous instructions in the project handouts. Each project release included an online handout of the project requirements and a set of JUnit test cases contained in a single .java file.

IV. Programming Assignments

A. CashRegister

The class `CashRegister` simulates a simple cash register object that maintains two instance variables: `currentAmountDue` and `totalDailySales`. In addition to the typical get methods, the class also provides the following methods:

- `void scanPrice(double price)`: add the given price to the current amount due
- `void completeSale()`: calculate the tax and add it to the current amount due. This method is invoked after the last item is scanned
- `void cancelSale()`: cancel the current transaction
- `makePayment(double pay)`: receive payment of the given amount from the customer
- `clearAllSales()`: clear the total daily sales. This method requires the user to confirm the operation by answering a ″y″ or ″n″ via the standard input.

In contrast to typical getter methods that return non-void, the above methods return a `void` and they must display (via System.out.print[ln]) textual output that contains either a monetary amount formatted with `NumberFormat` or an error/warning message.

For this project, the following challenges were addressed when writing the JUnit test cases:

- to capture these textual outputs and verify that they are correct and satisfy the requirements.
- to automatically generate a random ″y″ or ″n″ response when testing `clearAllSales()`.

B. A Game of Pig

The class `Pig` is an implementation of a two-dice game in which two players take turns to earn at least 100 points. Rolling a pair of 1s $(1, 1)$ causes the current player to lose all his points, while rolling a 1 $(1, n)$ causes the current player to lose only the points earned in the current round. In both cases, the current player loses his turn and the opponent continues the game. At any time, the current player may decide to hold and let the opponent continue. One of the players is human and the other is the computer.

In addition to the typical getter methods, the class must provide the following methods:

- `void playerRolls()` to simulate the current player rolling the two dice
- `void playerHolds()` to simulate the current player holding his current turn
- `void computerTurn()` to simulate the computer's entire turn, the computer continues to roll until 20 points are accumulated o turn is lost
- `void playerTurn()` to simulate the human player's entire turn
- `void restart()` reset all the instance variable to start a new game
- `boolean playerWon()` and `boolean computerWon()` to check whether the human/computer player has won the game

Unlike the `CashRegister` project, this project has a GUI component with most of the solution provided by the instructor. The students were required to adapt the provided GUI code to display appropriate states from the `Pig` game engine. In addition, this game requires a supporting class `Die`, which is also provided by the instructor.

The challenge in designing JUnit test cases for this project is to capture and verify textual output produced on the program's standard output.

C. The Game of Chess

The game of chess assigned to our CS2 students was designed to enforce the concept of behavior-based inheritance. For this assignment, we provide the following interfaces in a JAR file.

- `IChessPiece` an interface that defines the behavior of a chesspiece. The core method of this interface is `boolean isValidMove (Move m, IChessBoard b);`
- `IChessBoard` an interface that mainly serves as a wrapper of a two-dimensional structure of a chessboard.
- `IChessModel` an interface that defines the behavior of a typical chess game.

In the project, students are required to implement a base class (`ChessPiece`) that implements the `IChessPiece` interface. Other chesspiece classes(Pawn, Rook, Knight, Bishop, Queen, King) then inherit from `ChessPiece`. Each chess piece has a unique set of rules to move, but they all also share a common set of rules:

- A chess piece can only be moved to a different location (the source and destination must be two different locations on the board)

- A chess piece can only be moved by the player who owns the piece
- A chess piece can capture only another piece of a different color

The above common set of rules are verified by the base class `ChessPiece`, while rules specific to individual chess piece are verified by the corresponding classes.

For our CS2 projects, students must implement provided interfaces and develop their own test cases. Unlike CS1 projects where students must use specific class names in their projects, CS2 students were allowed to use any name for their Java classes as long as they implement the specified interfaces. The main challenges for writing JUnit test cases for such project is to use Java reflection to discover the actual class names at runtime.

V. Developing Test Cases

This section describes how we developed the project test cases.

A. Capturing Standard Outout

To capture a program's standard output, we use the `System.setOut` method for output redirection. Combining this method with an `OutputStream` subclass that overrides its `write()` method, we were able to log any `System.out.print[ln]` originating from a student's program to a `StringBuilder` object created in the JUnit test fixture. This technique allows the test cases to verify the expected output by inspecting the `StringBuilder` object. The overall setup is shown in Figure 1. Once the output redirection setup in a JUnit test fixture, a test method to verify the System.out.println() output produced by one of the `CashRegister` methods is shown in Figure 2.

B. Generating Random Input

The `CashRegister` project requires our tester framework to supply a random ″y″ or ″n″ for testing its `cancelAllSales()` method. The technique we used to achieve this task is to combine input redirection with a `PipedInputStream`. The overall setup is shown in Figure 3.[1]

With the above input redirection initialized in a JUnit test fixture, a test method that requires a controlled input to the standard input (`cancelAllSales()` in the following example) can now be written as follows:

```
public class CashTest {
  private static StringBuilder textLog;
  private static PrintString originalStdout;

  private static OutputStream logger =
    new OutputStream() {
    @Override
    public void write (int b) {
      textLog.append ((char) b);
    }
  };

  @BeforeClass
  public static void setUpBeforeClass() {
    textLog = new StringBuilder();
    originalStdout = System.out;
    System.setOut (new PrintString (logger));
  }

  @AfterClass
  public static void tearDown() {
    System.setOut (originalStdout);
  }
}
```

Fig. 1. Redirecting Standard Output to an OuputStream

```
  @Test
  public void testScanPrice()
  {
    CashRegister cash = new CashRegister (...);
    // reset the text logger
    textLog.setLength (0);

    cash.scanPrice (25.99);

    String tester = textLog.toString();
    assertTrue (tester.indexOf(EXPECTED) >= 0);
    assertEquals (EXPECTED_LENGTH, t
       ester.length());
    // more Assert calls
  }
```

Fig. 2. Using `textLog` in a test method

In Figure 4 the invocation of the method `yesNo.generateReponse()` seems to be out of order; it was called prior to calling `clearAllSales()`. The sequence is actually correct because the "n" input must be injected to the program standard input prior to the actual reading of the standard input by `clearAllSales()`. Swapping the order of the two calls will result in a deadlock.

C. Escaping Infinite Loops

Getting trapped in an infinite loop is a common mistake made by novice programmers. When a method may potentially get stuck in an infinite loop,

[1] `try-catch` block for exception handling is intentionally removed from the code snippet

```
public class CashTest {
  private class YesNoResponder extends
    PipedOutputStream {
    public void generateResponse (boolean flag)
    {
      write (flag ? ’y’ : ’n’);
      // place a newline character to simulate
      // the <ENTER> key
      write (’\n’);
    }
  }

  private static YesNoResponder yesNo;

  @BeforeClass
  public static void setUpBeforeClass() {
    // other code here
    yesNo = new YesNoResponder();
    PipedInputStream fakeKbd =
       new PipedInputStream();
    yesNo.connect (fakeKbd);
    System.setIn (fakeKbd);
    // other code here
  }
}
```

Fig. 3. Using PipedOutputStream for injecting input

```
  @Test
  public void testClearSales()
  {
    CashRegister cash = new CashRegister (...);
    textLog.setLength (0);
// reset the text logger

    boolean resp = myRandom.nextBoolean();
    // place a random ”y” or ”n”
    yesNo.generateResponse (resp);
    cash.clearAllSales();
    // more Assert calls
  }
```

Fig. 4. Generating a random response

the associated tester method should be designed to provide a graceful way of escaping the infinite loop. JUnit4 provides the @Timeout annotation for this purpose.

An alternate solution is to run the *testee* method in a thread separate from the *tester* thread. After spawning a new thread, the *tester*/main thread can use a timer to set a time limit for the *testee* thread. The overall setup is shown in Figure 5[2]

```
  private class TesteeThread implements Runnable
  {
    @Override
    public void run() {
      // invoke the testee method here
    }
  }

  @Test
  public void testWithTimeLimit() {
    Thread thr = new Thread (new TesteeThread());
    thr.start();
    long mark = System.currentTimeMillis();
    long now;
    do {
       now = System.currentTimeMillis();
    } while (now - mark < LIMIT && thr.isAlive());
    if (thr.isAlive()) {
      thr.interrupt();
      Assert.fail (”Your loop runs too long”);
    }
  }
```

Fig. 5. Imposing time limit

D. Exploiting Inheritance

The test cases for the Game of Chess are organized using the same structure as the inheritance hierarchy of the classes under test. We first created a JUnit test ChessPieceTest for testing ChessPiece. The rest of JUnit test files inherit from ChessPieceTest. Moreover, to handle differences between the black and white pawns "forward" directions, we created an intermediate abstract parent PawnTest that inherits from ChessPieceTest. Then, we created BlackPawnTest and WhitePawnTest that overrides the "forward()" method.

E. Include Required Constructors

As Java interfaces are introduced in a programming assignment, the instructor test cases must be designed to discover the implementing classes at runtime. Since the signature of a constructor cannot be enforced via an interface, the assignment should specify the expected signature of the constructor(s) of the implementing classes. Without this requirement it would be impossible for the instructor tester code to instantiate Java objects needed for testing. Some of the constructors can also be used to facilitate testing those methods that require a particular precondition.

VI. Issues in Designing Test Cases

A. Avoid Silent Failures

For each assert method, JUnit provides at least the following two variants:

[2] the code shown here uses a busy loop technique. A better approach would be to replace the busy loop with a timer

```
assertTrue (expectedValue, actualValue);
assertTrue ("hint / message",
    expectedValue, actualValue);
```

Using the first variant does not help students diagnose why their code failed. JUnit simply reports that the test failed. The second variant can include useful hints, provided in the string message, to help students fix their code. For example, consider a method designed to calculate the average of numbers within an array. The following messages provide a range of feedback.

- **Poor**: no message
- **Good**: `calcAverage()` does not work correctly
- **Better**: `calcAverage()` result is close but not correct. Are you using a double for the average?
- **Best**: `calcAverage()` result is not correct. Did you consider the possibility of zero elements?

B. Avoid False Positives

We found cases where an incomplete student implementation produced positive test results. One cause of these false positives was due to Java default behavior of automatically initializing variables to zero,false, or null. In other situations, there was an insufficient number of test cases to cover all possible conditions. For example, we neglected to test for a divide by zero exception in the `calcAverage()` method. Student solutions passed the test but did not check for zero elements. The error was identified when reading student solutions but was not identified as intended with the automated JUnit tests. Test cases must be robust and anticipate a range of input permutations and error conditions. Otherwise, student may develop a false-sense of confidence in the instructor-provided test cases.

C. Provide JUnit Documentation

JUnit testing can be used throughout the first programming course. Students can start by using JUnit tests as a black box and eventually progress from reading the instructor-provided test cases to writing their own by the end of the course. An effective way for students to learn is to read the instructor-provided cases if they are well commented. The more advanced students are naturally curious and want to know what the test cases do.

D. Avoid Shadowing of Protected Variables

We also found cases where students' test cases may show positive results while instructor's provided test cases did not due to shadowing of a protected variable. In one of the assignments where students were required to use inheritance, a student incorrectly redeclared an instance variable in a child class which has been declared in the parent class, thus hiding the visibility of the parent's variable. This type of bug can be difficult to locate (even for an experienced programmer), but very easy to fix. If type of bugs causes a potential source of confusion to the students, the instructor's test cases should use Java reflection to identify such bugs.

VII. Conclusion

Novice programmers can use and benefit from JUnit tests in an introductory programming course. Robust tests can be created using advanced Java features such as reflection to detect at runtime if a method is present, capturing standard output using `OutputStream`, generating random input from a `PipedInputStream`, and using the `@Timeout` annotation for detecting infinite loops.

There are several best practices that help make the process more effective from the student's perspective. First, it is essential to write clear warnings and error messages that provide helpful clues. Second, warn students not to get too stressed about a single failed test. It could be a minor error and they should use their time implementing additional methods before returning to identify the error. Otherwise, students spend too much time on trivial problems and fail to make sufficient progress on the entire assignment. Third, JUnit java code should have internal comments written for a novice programmer. Students can read the code to learn how to adapt and write their own tests. And finally, tests must be accurate and comprehensive. Students must be able to trust and rely upon the instructor-provided test cases.

Acknowledgment

We would like to recognize our colleague, Zack Kurmas for the time and energy he spent on designing the Game of Chess assignment. We believe the assignment is very well-designed and has helped our students to gain comprehensive understanding of inheritance, interfaces, and polymorphism.

References

[1] Armin Biere Cyrille Artho. Advanced unit testing. In *Proceedings of the 2006 International Workshoo on Automation of Software Testing*, Shanghai, China, May 2006.

[2] Association for Computing Machinery and IEEE Computer Sociery. Computer science curriculum 2008: An interim revision of cs 2001, December 2008.

[3] BlueJ IDE. http://bluej.org, 2013.

[4] Michael Olan. Unit testing: Test early, test often. In *The Journal of Computing Sciences in Colleges*, December 2003.

[5] Gábor Szeder. Unit testing for multi-threaded java programs. In *PADTAD*, Chicago, Illinois, July 2009.

[6] JUnit Team. Junit, March 2013.

Fuzzy Logic Model for Predicting the Number of Online Courses Needed from Number of Students Enrolled in Higher Education

Rosa Leonor Ulloa Cazarez[1], Cuauhtémoc López-Martín[2]
[1, 2] Information Systems Department, CUCEA, Universidad de Guadalajara, Jalisco, Mexico
[1] rosi_ulloa@cucea.udg.mx, [2] cuauhtemoc@cucea.udg.mx

Abstract –

Context: In Higher Education where online courses are offered, one need is to predict the number of courses to be open. At date, some types of models have been used for this goal, such as models based upon machine learning, statistical and softcomputing approaches.
Goal: To propose a softcomputing model for predicting the number of online courses (NOC) needed from the number of students enrolled in Higher Education.
Hypothesis: Prediction accuracy of a fuzzy model is better or equal than a statistical regression model.
Results: Prediction accuracy of a fuzzy model was slightly better than that of a statistical regression model
Conclusion: Fuzzy logic could be applied for predicting the NOC needed from the number of students enrolled in Higher Education.

Keywords: Higher Education, Online courses prediction, fuzzy logic, statistical regression.

1 Introduction

Online education or e-learning, are terms used for the instruction facilitated and delivered online, through different technologies and typically have no face to face meetings [1].

Several institutes have incorporated into their curriculum online courses, in order to respond different purposes: improve quality of instruction, improvement of the use of spaces and attend the demand for enrollment [5]. Some institutes have offered full online programs and there are others which have born as online educative institutions. As a result of this variety, the definition of the online higher education system is a difficult job and is more accurate to the purposes of this study, called it as the offer of online education.

Because the nature of the online education, institutions have acquired benefits as the opportunity to provide more students with their educational services and the reduction of expenditures, and adult learners have been able to combine higher education and training with their work and home responsibilities [5].

These characteristics have allowed the growth of the online education offerings [3] whom have attracted an increasing number of participants and as a result, some projections of online education demand growth indicates that this effect continue for the next ten years [1] [7].

We believe that the faster growing of the enrollments in online education offerings, needs more attention in order to improve the planning processes through a proper analysis of the trends [1].

In this study we propose a fuzzy logic model for predicting the number of online courses (NOC) needed from the number of students enrolled in Higher Education. We have found studies using other dependent and independent variables, which have been used for generating and validating models based on (1) machine learning, such as neural networks, support vector machines, and genetic algorithms, (2) statistical models, such as simple and multiple regressions, and softcomputing, such as fuzzy logic.

The accuracy of our proposed model is compared with that of a statistical regression model. Data for generating and validating the models were obtained from a study realized by Kardan et al. [3] who predicted course selection of students in the context of two online courses. Data from the year 2005 to 2011 were used for generating the two models, and data of 2012 were used for validating the models.

Hypothesis of this research is the following:

> Prediction accuracy of a fuzzy model is better or equal than that of a statistical regression model.

The rest of this study is structured as follows: in section 1.1 a brief introduction to fuzzy logic is done, in Section 1.2 the studies related to ours are analyzed, whereas in Section 1.3, the criterion for evaluating the model is described. In Section 2, the data and the process followed for generating and validating the models are detailed. Sections 3 and 4, the models are generated and validated, respectively. Finaly, in section 5 the conclusions and future work are mentioned.

1.1 Fuzzy logic

The prediction techniques related to on-line courses have as characteristic that their variables use to be described using categorical data (nominal or ordinal scale) such as *small*, *medium*, *average*, or *high* named linguistic values. A more comprehensive approach to deal with linguistic values is by using fuzzy set theory [13].

A fuzzy model has the following two main properties [9]: (1) It operates in at the level of linguistic terms (fuzzy sets), and (2) it represents and process uncertainty.

A fuzzy logic model facilitates the representation and manipulation of uncertain, incomplete, imprecise or noisy data. Specifically, fuzzy logic offers a particularly convenient way to generate a keen mapping between input and output spaces thanks to natural expression of fuzzy rules [13]. In accordance with on-line courses in Higher Education issue, two considerations justify the decision of implementing a Fuzzy model: first, it is impossible to develop a precise mathematical model of the domain; second, measures only produce predictions of the real complexity. Thus, according to the previous assertions, formulating a set of natural rules describing interactions between the number of students and the number of on-line courses in Higher Education estimation could reveal their correlation. In this study a rule induction system replacing the crisp facts with fuzzy inputs, an inference engine uses a base of rules to map inputs to a fuzzy output which can either be translated back to a crisp value, is constructed.

A software tool was used to generate the fuzzy logic system with type: mamdani, *and* method: *min*, *or* method: *max*; implication: *min*, aggregation: *max*, and defuzzyfication: *centroid*.

1.2 Related work

Our literature review was focused in studies related with the following two main searches: (1) online higher education; and (2) prediction, estimation and forecasting models, (once we identified that these three words have indistinctly been used).

In the papers found, we targeted the following questions by study: (1) what was the purpose of the study? (2) What models/techniques were used? (3) What variables were involved? (4) What was the result? and (4) How the models/techniques were generated and validated?

Lykourentzou, I. GIannoukos, V. Nikolopoulos, G. Mpardis and V. Loumos [4], had the main purpose of predicting students dropout through the combination of three machine learning techniques. The three were a Feed-Forward neural network, a Support Vector Machine and a Probabilistic Ensembled Simplified Fuzzy named ARTMAP. The dependent variable was student dropout and the independent variables were of two types, (1) demographic (gender, residency, working experience in the field of the course, educational level, fluency of language) and (2) time variant (student's progress, level of engagement and participation). According to the results of the study, the combination of the three machine learning techniques, leads to more accurate and pront identifying of the students dropout.

N. Nistor and K. Neubauer [8], predicted dropouts from student's participation through Discriminant Analysis statistical technique. The dependent variable was students dropout and the independent variable was participation. In this study there were not intention of evaluating the discriminant analysis technique used, but a propose of strategie to measure and predict dropout takes place.

O. Yildiz, A. Bal, S. Gulsecen and F. Damla Kentli [11], evaluated academic performance in Distance Education using a Genetic-fuzzy model to predict academic performance from recency, frecuency and monetary. The prediction accuracy of students' academic performance using the Genetic-Fuzzy model got an acceptable value.

S. Huang and N. Fang [2], predicted student performance in an engineering dynamics course through the comparation of four mathematical models: Multiple linear regression, two neural network models (Multilayer Perception neural network, and Radial Basis Function network), and a Support Vector Machine. The predictor variables were student's problem solving skills, level of statisticals knowledge, student's mathematical skills and students understanding of physical concepts. The analysis of results revealed that the type of mathematical model had only a slight effect on the average prediction accuracy and on the percentage of accurate predictions; on the other hand, the combination of predictor variables had only a slight effect on the Average Prediction Accuracy (APA) but a profound impact on the Percentage of Accurate Predictions (PAP). Support vector machine models had the highest PAP; and adding more predictor variables did not help improve the average prediction accuracy of any of the models.

A. A. Kardan, H. Sadeghi, S. S. Ghidary and M. R. Fani Sani [3], predicted students course selection through a Multi-layer Perceptron neural network from nine predictor variables: course characteristics, instructor's characteristics, student's workload, course grade, course type, course time, number of time conflicts, final examination time and student demands. In this study, the neural network proved to be effective and outperformed other methods predicting course selection considering students demands, but without considering the student demands, the neural network was not as accurate as the model with this consideration.

H.-W. Vivian Tang and M.-S. Yin [10], compared two grey prediction models and exponential smoothing for accuracy in prediction of the education expenditure from school enrollment. Forecasting efficiency of one of the grey prediction models used, was superior to exponential smoothing and the other grey prediction model used.

Jung Jae Yup [12], used statistical techniques derived from self-determination theory, expectancy-value theory and research on occupational indecision to predict enrollment in higher education from amotivation and indecision. The results of the study proved that both models had a good fit.

Based on the advantages described in section "1.1 fuzzy logic" as well as the previous related work analysis in which we did not find any study where a fuzzy model has been applied for predicting the number of on-line courses in higher education, we proposed a fuzzy logic model.

1.3 Accuracy criterion

A common criterion used to assess prediction models is the Magnitude of Relative Error (MRE) [6]. The MRE is defined as follows:

$$MRE_i = \frac{\left| Actual\ Courses_{\mathrm{i}}\ \text{-}\ Predicted\ Courses_{\mathrm{i}} \right|}{Actual\ Courses_{\mathrm{i}}}$$

The MRE value is calculated for each observation *i* whose number of courses is predicted. The aggregation of MRE over multiple observations (N) can be achieved through the Mean MRE (MMRE) as follows:

$$MMRE = (1/N)\sum_{i=1}^{N} MRE_i$$

The accuracy of a prediction model is inversely proportional to the MMRE.

2 Experimental design

2.1 Data description

Table 2 shows the data of courses from 2005 to 2011 years obtained from [3] where CNE and ITME are the names of the courses (CNE: Computers Network Engineering, and ITME: Information Technology and Management Engineering).

2.2 Process for generating and validating the models

The steps for generating and validating the models were the following:

1. Selection of a sample involving data from 2005 to 2011 years.
2. Scatter plot analysis. The dependent variable versus independent variable.
3. Calculation of the coefficient of correlation.
4. Calculation of the coefficient of determination.
5. Linear regression equation generation.
6. ANOVA for the linear regression equation.
7. Fuzzy rules determination based on a correlation analysis.
8. Membership function selection.
9. Fuzzy logic model generation based on adjusting membership function parameters.
10. Validating the linear regression equation and fuzzy logic models using data of the 2012 year.

Table 2. Data for generating the models

Year	Season	Program	Number of students	Number of courses
2005	Spring	CNE	194	23
2005	Fall	CNE	180	22
2005	Spring	ITME	146	19
2005	Fall	ITME	161	21
2006	Spring	CNE	205	24
2006	Fall	CNE	188	23
2006	Spring	ITME	156	20
2006	Fall	ITME	177	22
2007	Spring	CNE	172	21
2007	Fall	CNE	186	22
2007	Spring	ITME	158	20
2007	Fall	ITME	184	23
2008	Spring	CNE	208	24
2008	Fall	CNE	197	23
2008	Spring	ITME	152	20
2008	Fall	ITME	160	21
2009	Spring	CNE	224	25
2009	Fall	CNE	196	23
2009	Spring	ITME	169	21
2009	Fall	ITME	175	22
2010	Spring	CNE	180	22
2010	Fall	CNE	215	24
2010	Spring	ITME	172	21
2010	Fall	ITME	191	23
2011	Spring	CNE	231	25
2011	Fall	CNE	194	23
2011	Spring	ITME	183	22
2011	Fall	ITME	179	22

3 Generation of models

3.1 Data analysis and linear regression model

Figure 1 shows a scatter plot correlating the number of students to number of courses. The relationship between these two variables is the following: The higher the value of students, the higher the number of courses.

The correlation value is r = 0.97, that is, there is a strong relationship between the two variables. The linear regresion model generated is the following:

Number of Courses = 9.3751 + 0.0698416*Number of students

The coefficient of determination is $r^2 = 0.94$, it means that the model as fitted explains 94% of the variability in courses.

The p-value of the analysis of variance (ANOVA) of the linear regression equation (Table 2) shows that there is a statistically significant relationship between the number of courses and the number of students at the 99.0% confidence level.

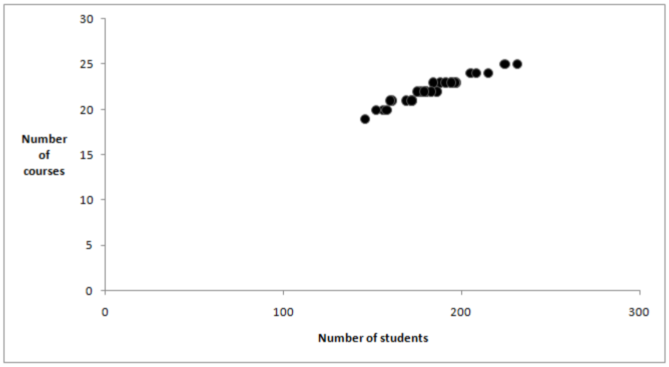


Figure 1. Scatter plot of Students vs. Courses

Table 2. ANOVA for the linear regression equation

Source	Sum of Squares	Degrees of freedom	Mean Square	F-Ratio	P-Value
Model	58.9738	1	58.9738	489.35	0.0000
Residual	3.1333	26	0.1205		
Total	62.1071	27			

3.2 Fuzzy model

The fuzzy model can be created from the expert knowledge that in a verbal form is translated into a set of if–then rules. A certain model structure is created, and parameters of this structure, such as membership functions and weights of rules, can be tuned using input and output data.

The following fuzzy rules were formulated based on the correlation (*r*) showed in Figure 1:

1) If (*Number of students* is *Small*) then (*Number of courses*) is *Small*
2) If (Number of students is *Big*) then (*Number of courses*) is *Big*

Implementing a fuzzy system requires that the different categories of the different inputs be represented by fuzzy sets, which in turn is represented by membership functions (MF). The MF type considered to this experiment is triangular (because of this type has demonstrated acceptable results when it has been applied to prediction [4]).

The input (*Number of students*) and output (*Number of courses*) was composed with two membership functions: *small* and *big*.

Parameters of membership functions for the input and for the output were iteratively adjusted until obtaining the smallest MMRE possible.

A triangular MF is a three-point (parameters) function, defined by minimum (a), maximum (c) and modal (b) values, that is, MF(a,b,c) where $a \le b \le c$ [9]. Their scalar parameters (a, b, c) are defined as follows:

$$MF(x) = 0 \text{ if } x < a \quad MF(x) = 1 \text{ if } x = b \quad MF(x) = 0 \text{ if } x > c$$

Figures 2 and 3 show the membership functions, whereas Table 3 shows the parameters for the input and output by triangular membership function.

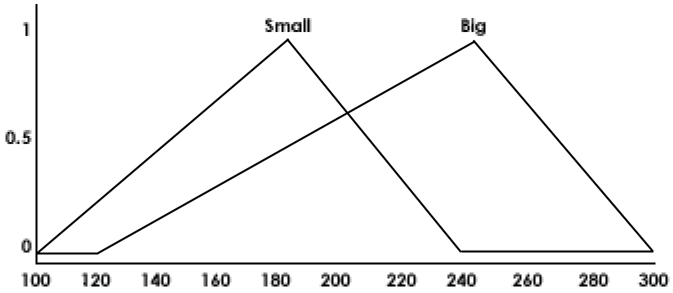


Figure 2. Membership functions for number of students (input)

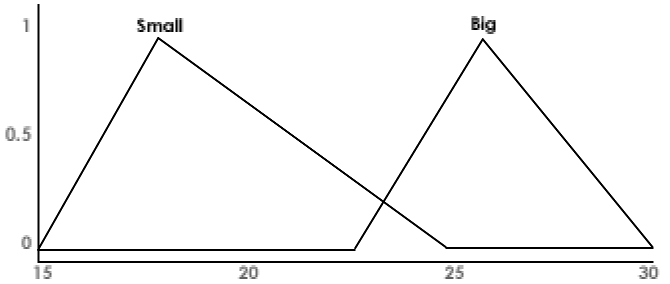


Figure 3. Membership functions for number of courses (output)

Table 3. Parameters of fuzzy model membership functions

Type of variable	Variable	MF	Parameters		
			a	c	b
Input	*Number of students*	Small	100	182	238
		Big	121	243	300
Output	*Number of courses*	Small	15	18	25
		Big	23	26	30

3.3 Model adequacy checking

The linear regression equation and the fuzzy model were applied to original data set (Table 1). The accuracy obtained (MMRE) by model is showed in Table 4.

4 Validation of models

Once that the linear regression equation and the fuzzy model were generated and their adequacy was checked, the two models were applied to a new data set obtained from [3] involving data of the 2012 year. The accuracy obtained (MMRE) is showed in Table 5. It can be showed that the fuzzy logic model had a slightly better accuracy (MMRE= 0.28).than the regression model (MMRE= 0.30).

Table 4. Prediction accuracy by model (LRE: Linear regression Equation; FLM: Fuzzy Logic Model)

Number of students	Number of courses	LRE	MRE	FLM	MRE
194	23	22.92	0.004	22.01	0.043
180	22	21.94	0.003	21.68	0.015
146	19	19.57	0.030	21.08	0.110
161	21	20.61	0.018	21.37	0.018
205	24	23.68	0.013	22.42	0.066
188	23	22.50	0.022	21.84	0.050
156	20	20.26	0.013	21.29	0.064
177	22	21.73	0.012	21.63	0.017
172	21	21.38	0.018	21.55	0.026
186	22	22.36	0.016	21.79	0.009
158	20	20.40	0.020	21.32	0.066
184	23	22.22	0.034	21.75	0.054
208	24	23.89	0.004	22.57	0.060
197	23	23.13	0.005	22.10	0.039
152	20	19.98	0.001	21.22	0.061
160	21	20.54	0.022	21.36	0.017
224	25	25.01	0.000	23.76	0.050
196	23	23.06	0.002	22.07	0.040
169	21	21.17	0.008	21.50	0.024
175	22	21.59	0.019	21.60	0.018
180	22	21.94	0.003	21.68	0.015
215	24	24.38	0.016	22.99	0.042
172	21	21.38	0.018	21.55	0.026
191	23	22.71	0.013	21.92	0.047
231	25	25.50	0.020	24.70	0.012
194	23	22.92	0.004	22.01	0.043
183	22	22.15	0.007	21.73	0.012
179	22	21.87	0.006	21.66	0.015

MMRE	0.013		0.038

Table 5. Prediction accuracy by model (LRE: Linear regression Equation; FLM: Fuzzy Logic Model)

Number of students	Number of courses	LRE	MRE	FLM	MRE
235	25	25.78	0.031	25.46	0.018
223	24	24.94	0.039	23.65	0.014
166	21	20.96	0.002	21.45	0.022
180	23	21.94	0.046	21.68	0.057

MMRE	0.030		0.028

5 Conclusions

In this study, data of the number of students enrolled in on-line Higher Education and number of courses were used for generating and validating independent data samples. The models for predicting the number of courses were a linear regression and a fuzzy model. These models were generated from a data set composed with 28 records that were obtained from 2005 to 2011 years. These two models were validated when they were used for predicting the number of courses of four records corresponding to 2012 year. The hypothesis accepted was the following:

> Prediction accuracy of a fuzzy model was better than a statistical regression model.

This result suggests that a fuzzy model using as independent variable (input) the number of students, can be used for predicting the number of on-line courses needed.

Future research involves the use of additional independent and dependent variables, obtained from other datasets, in models based on fuzzy logic and neural networks.

6 References

[1] Allen & Seaman, "Class Differences. Online Education in the United States, 2010". United States of America: Babson Survey Research Group. 2010. Available at: http://sloanconsortium.org/publications/survey/class_differences

[2] Huang, S. & Fang, N. "Predicting student academic performance in an engineering dynamics course: A comparison of four types of predictive mathematical models." Computers & Education. 61 (133-145) (2013).

[3] Kardan, A., Sadeghi, H., Ghidary S. & Fani Sani, M. "Prediction of student course selection in online higher education institutes using neural network". Computers & Education. Thompson Reuters. 65 (1-11) (2013)

[4] Lopez-Martin, C. Yañez-Marquez, C. & Gutierrez-Tornes, A. "Predictive Accuracy Comparison of Fuzzy Models for Software Development Effort of Small Programs". Journal of Systems and Software, Vol. 81, No. 6, Elsevier, 2008

[5] Lykourentzou, I., GIannoukos, I., Nikolopoulos, V., Mpardis, G. & V. Loumos. "Dropout prediction in e-learning courses through the combination of machine learning techniques". Computers & Education. Elsevier. 53. (950-965) (2009).

[6] Myrtveit, I., Stensrud, E. "Validity and reliability of evaluation procedures in comparative studies of effort prediction models". Empirical Software Engineering. Springer. 17 (1-2) (2012) 23-33. DOI: 10.1007/s10664-011-9183-7

[7] National Center for Education Statistics, National Center for Education Statistics, Institute of Education Sciences, January 2013. Available at: http://nces.ed.gov/programs/projections/projections2021/. [Accessed 14 March 2013]

[8] Nistor, N. & Neubauer, K. "From participation to dropout: Quantitative participation patterns in online university courses". Computers & Education.

Thompson Reuters. 55 (663-672) (2010).

[9] Pedrycz W., Gomide F. “An Introduction to Fuzzy Sets”; The MIT Press, 1998.

[10] Vivian Tang, H. & Yin, M. “Forecasting performance of grey prediction for education expenditure and school enrollment”. Economics of Education Review. Elsevier. 31 (452-462) (2012).

[11] Yildiz, O., Bal, A., Gulsecen, S. & Damla Kentli, F. “A Genetic-Fuzzy based mathematical model to evaluate the Distance Education students' academic performance” Procedia. Social and Behavioral Sciences. Elsevier. 55 (409-418) (2012).

[12] Yung, J. “Amotivation and indecision in the decision-making processes associate with University entry. Research in Higher Education. Elsevier. 54 (115-136) (2012).

[13] Zadeh. Lotfi A. “From Computing with Numbers to Computing with Words – From Manipulation of Measurements to Manipulation of Perceptions”. IEEE Transactions on Circuits and Systems – I: Fundamental Theory and Applications, vol. 45, no. 1, pp. 105-119, 1999

An Empirical Study on Remote Lectures Using Video Conferencing Systems

Tetsuya Oishi, Weiqiang Kong, Yasutaka Kamei, Norimichi Hiroshige, Naoyasu Ubayashi, and Akira Fukuda
Graduate School and Faculty of Information Science and Electrical Engineering, Kyushu University, Japan
744 Motooka, Nishi-ku, Fukuoka 819-0395, Japan
oishi@qito.kyushu-u.ac.jp

***Abstract*—We have been holding remote lectures among 5 universities in Japan using video conferencing systems supported by Polycom. Initially, we connected Kyushu University, the center and information source of the remote lectures, with 4 other universities using the function of Multi-point Control Unit (MCU) located in Kyushu University. However, this setting was unstable and the quality of the lectures could not be guaranteed. Therefore, we tried later to employ a new set-up by utilizing dedicated equipment for MCU for enhancing the lecture quality. In this paper, we describe and report our experience on the remote lectures using MCU. In addition, we have conducted a survey of the students on the environment and the method of giving the remote lectures. The analysis results of the survey and our corresponding proposals will be presented as well in the paper.**

***Keywords*-Remote Lectures, Video Conferencing Systems, Multi-point Control Unit**

I. Introduction

Remote lectures have gradually become more popular and been expanding their usage for sharing lectures among multiple universities. **QITO** (**Kyu**shu University **I**nformation Communication **T**echnology Architect Educational Pr**O**gram) has also been holding remote lectures using video conferencing systems supported by Polycom (http://www.polycom.com/). However, some fatal problems were encountered such as lecture interruption due to breakdown in Internet communication. To resolve these problems, we introduced **MCU** (**M**ulti-Point **C**ontrol **U**nit).

In addition to traditional lectures given so far, the QITO program, a two-year education course for master students, provides special curriculums centering on “Project-Based Learning (PBL)”, “Long-Term Internship”, and “ICT All-Round Education/Omnibus”, which were insufficient in the past.

In PBL [1], learning basic skills such as software development and project management is scheduled for the 1st-year 1st semester. Moreover, in the 2nd semester and the 2nd-year 1st semester, students are requested to deal with practical problems assigned by teachers from industries or universities.

The long-term internship does not aim at company experiencing. The 1st-year students in QITO program are anticipated to participate in the internship for more than one month (up to two months) in summer, and to work on a real company project for software development.

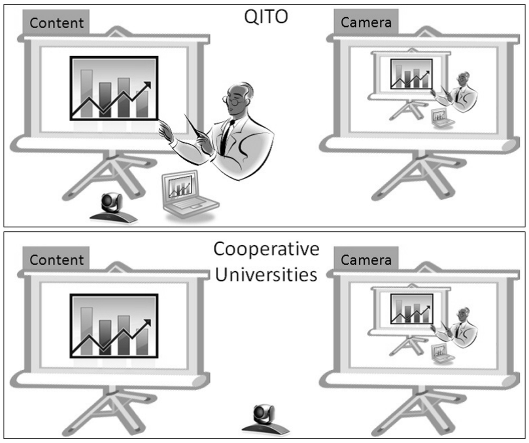


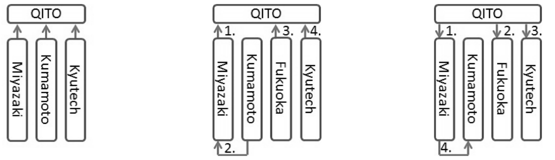
Figure 1. Screen location

The omnibus lectures consist of ICT Society Business Special Course (1st-year 1st semester), Advanced ICT Engineering Special Course (1st-year 2nd semester), Project Management Special Course (1st-year 2nd semester), and Advanced ICT Leaders Special Course (intensive course in 1st-year only). These courses, given by first-class managers, technicians, and researchers from industries, explore the most advanced ICT technologies and their trends.

Omnibus lectures, especially the ICT Society Business Special Course and the Advanced ICT Engineering Special Course, are remote lectures that utilize video conferencing systems in between Kyushu University and “cooperative universities” (Kyushu Institute of Technology (Kyutech for short), Kumamoto University, University of Miyazaki, and Fukuoka University). The ICT Society Business Special Course is held every Thursday from 2.50pm to 6.10pm in 1st-year 1st semester, and the Advanced ICT Engineering Special Course is held every Thursday from 2.50pm to 6.10pm in 1st-year 2nd semester. Totally, 10 and 7 lectures for each respective course have been held in 2012.

QITO, Kyutech, University of Miyazaki, and Fukuoka University use HDX 9002, while Kumamoto University uses PVX. These video conferencing systems are products of Polycom. HDX9002 is a video conferencing unit and PVX is a video conferencing software application.

Information distributed among QITO and the cooperative



(a) 4 locations (b) 5 locations (c) 5 locations (improved)

Figure 2. Traditional method to connect to some locations

universities includes: screen pictures (e.g., powerpoint lecture slides) displayed in a PC placed at QITO (hereafter called ***content***), the image of a camera also placed at QITO for filming a lecturer, and sound from all universities. A sketch of the screen of all universities is shown in Figure 1. The upper part of the figure shows the screens at QITO and the lower part shows the screens at the cooperative universities. The screen for the camera at QITO can also be switched to show screens of cameras located in each cooperative university.

The previous version of this paper was presented at ICET2013 [2]. In this paper, we extend the work in [2] by including a survey on the remote lectures, the analysis results of the survey, and our proposals. In particular, the following additional information will be presented:

- We describe the primary purpose of the survey and the timing when it was conducted. In addition, we show the number of students who attended the remote lectures and the number of valid answers.
- We obtained some opinions about the *environment* of the remote lectures from the survey, which was designed originally for finding out students' opinions about the contents of the lectures. We extracted some complaints from those opinions and classified them into ones for the environment of the remote lectures and ones for the method of giving the lectures.
- We describe how our solution for improving the environment of the remote lectures, which have already been proposed in [2], relate to the survey results, especially to the complaints. We report, as an analysis result of the survey, a fact that the more we improve the environment of the remote lectures, the less the complaints occur.
- For the complaints, we describe what we have already solved and what we plan to solve in the future.

II. Traditional Problems

We have been holding remote lectures using MCU (Multipoint Control Unit) function of HDX9002 at QITO till now. MCU facilitates remote conferencing (in our context, lectures) among multiple locations. In particular, the MCU function in HDX9002 supports connection of 4 locations at the same time. Since there were only 3 cooperative universities before 2007, we did not encounter any problem using HDX9002 (Figure 2(a)). From 2008, we encountered problems since one more cooperative university (Fukuoka University) was added and we had to use HDX9002 for connecting 5 locations all together.

A. Connecting to 5 locations

We adopted the method shown in Figure 2(b) to connect 5 locations. Each cooperative university connected to QITO. Here, the numbers attached in the figure represent the order of connection. Kumamoto University was set to connect to University of Miyazaki because MCU function in HDX9002 can only connect 4 locations at most at the same time. As a result, some problems occurred.

- Connection to the cooperative universities was down unexpectedly.
- *Content* was not distributed to the cooperative universities.
- QITO could not receive sound and images from the cooperative universities.

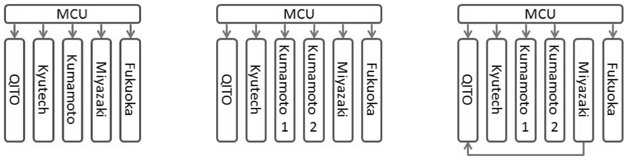
When the connection was down unexpectedly, a lecture had to be interrupted. Therefore, students in the cooperative universities might not be able to understand the lecture sufficiently. Moreover, the screen for the camera at QITO could not automatically switch to the camera image of the cooperative university, whose students were talking through a microphone (e.g., for answering a question posted by the lecturer at QITO). This is contrary to our expectation and the lecturer at QITO could not see the student who was talking.

Since a connection might become even more unstable if it lasted for long time, we periodically turned off the connection to all locations manually and re-established the connection again later on. However, this method could not essentially solve the problems. In addition, following the suggestions of an engineer from Polycom, we reversed the connection direction (as shown in Figure 2(c)) between QITO and the cooperative universities, but the problems still could not be solved.

B. Introducing dedicated equipment for MCU

MCU function in HDX9002 can connect maximally 4 locations at the same time. Using HDX9002 for connecting 5 locations is not recommended. Therefore, we decided to use a RMX2000 located in Information Infrastructure Initiative center of Kyushu University. RMX2000 is dedicated equipment for MCU that supports connecting 5 locations. Through utilizing this dedicated equipment, we expected that the remote lectures could be held smoothly and the problems (Section II-A) could be solved.

Before using the dedicated equipment for MCU, we made the following settings: 1) we set the duration of a conferencing call to 360 minutes so as to cover the whole lecture till its end, 2) we enabled H.239 [3] protocol. H.239, which is a recommendation of the ITU-T (International



(a) Basic

(b) For Kumamoto University

(c) For University of Miyazaki

Figure 3. The connection method through the dedicated equipmemnt for MCU

Telecommunication Union Telecommunication Standardization Sector), is a standard to show not only the images from a video conferencing system but also the pictures of PCs.

Previously, we had to make the connection manually among QITO and each university. After we introduced the dedicated equipment for MCU, all universities were connected by MCU automatically at 2:45pm (Figure 3(a)). MCU was set to turn off the connection automatically at 6:45pm, but each university could turn off the connection manually because the lecture ended at about 6:10pm.

We could change the screen at QITO in a way shown in Figure 4 by operating HDX9002 manually. The image from the camera at each respective university is automatically switched and displayed at QITO at 15-second intervals. In addition, if a student of a certain university talks through a microphone, the image of that university will be displayed at QITO.

Although the connection process was simplified thanks to the introduction of the dedicated equipment for MCU, some new problems occurred. We describe these new problems and our solutions to them in Section III.

III. New Problems Occurred and Our Solutions

In this section, we show concrete examples of the problems occurred after introducing the dedicated equipment for MCU, and our solutions. The time slots when problems occurred after we introduced the dedicated equipment for MCU are shown in Figure 5. During holding the remote lectures, we used Skype (http://www.skype.com/) to communicate with the staffs of the cooperative universities. The graph is made from the log of Skype. In the figure, vertical axis shows the days of lectures and the names of universities, and horizontal axis shows the time of the days. The lines painted denote time slots when some problems occurred.

A. Content was not shown when lectures started

[Problem] The resolution of the PC used by lecturers was set to 1600×900, which is the maximum value. On this setting, the *content* was not shown in all cooperative universities when lectures started.

[Solution] We understood that the *content* would not be shown at the cooperative universities if the resolution was too high. Therefore, we changed the resolution to $1024{\times}768$.

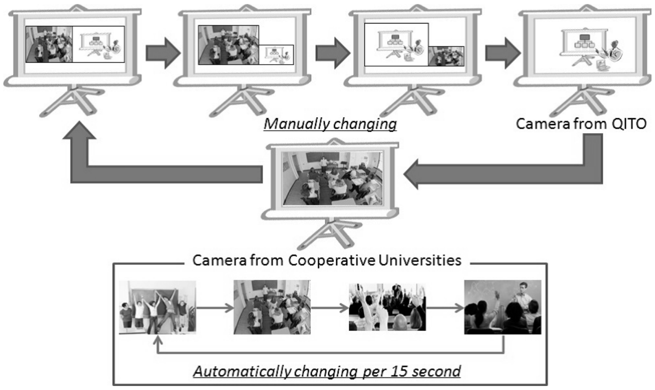

Manually changing
Camera from QITO
Camera from Cooperative Universities
Automatically changing per 15 second

Figure 4. Screen transition

B. MCU could not connect to the cooperative universities when lectures started

[Problem] Some cooperative universities were not connected from MCU automatically. This happened when video conferencing systems located in the cooperative universities were not started up. If MCU could not connect to a video conferencing system when a lecture starts, MCU will try to re-dial to that system and thus the connection could be restored. Normally, such re-dial actions are done automatically until MCU could connect with the conferencing system.

[Solution] However, if the connection could still not be able to restore, a MCU manager is requested to make the re-dial and connection manually.

C. Content became invisible in lectures

[Problem] The *content* became invisible during lectures at some cooperative universities. The resolution of the PC used by the lectures had already been lowered (Section III-A).

[Solution] As mentioned earlier, in remote lectures, not only the screen pictures of the lecture PC but also the images from cameras will be distributed. We found that, when the *content* became invisible, the transmission rate of the images from cameras was set to 1920Kbps. Our network was jammed by this setting. This problem was solved after we changed the transmission rate of the images from cameras into 384Kbps, which was the minimum value. Although this solved the unexpected loss connection problem, the image quality became worse especially when lecturers moved quickly. It is possible to set the transmission rate to a value of 384Kbps, 512Kbps, 768Kbps, 1024Kbps, 1472Kbps, or 1920Kbps. After examination, we found that 768Kbps was the best balanced value that could avoid the loss connection problem as well as provide a satisfactory image quality.

D. The connection was down after 90 minutes (Kumamoto University)

[Problem] The connection between MCU and Kumamoto University was down 90 minutes after a lecture started.

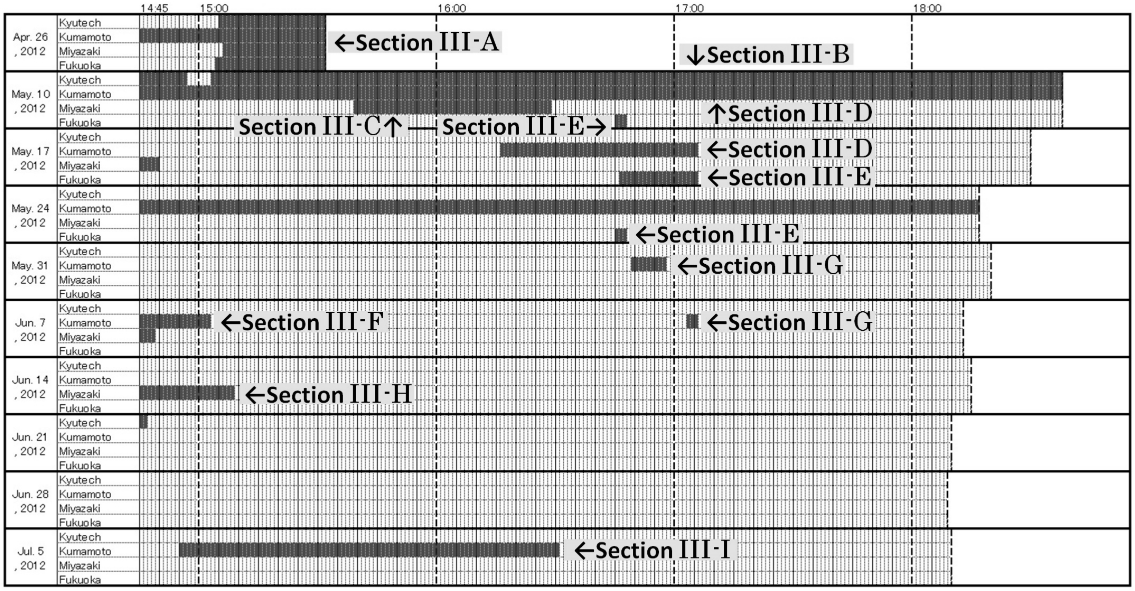


Figure 5. Time slots when problems occurred

[Solution] The reason was that the maximal conferencing time of the PVX used at Kumamoto University was set to 90 minutes, the default value. We solved this problem by changing the default maximal time from 90 minutes to unlimited

E. The connection was down after 120 minutes (Fukuoka University)

[Problem] The connection between MCU and Fukuoka University was down 120 minutes after a lecture started. We confirmed the default maximal conferencing time of the HDX9002 used in Fukuoka University, but that value was not 120 minutes. So the problem was not related to this setting.

[Solution] Video conferencing system of Polycom uses the H.323 [4] protocol. Using this protocol, MCU keeps a connection with each location with UDP after the connection is initially made with TCP. Moreover, both systems send keep-alive packets to each other. The connection will be turned off if there are no responses.

Having this fact in mind, we reconfirmed the network of Fukuoka University and found that timeout of firewalls was set to 30 minutes in Fukuoka University. In other words, TCP connection between MCU and Fukuoka University was disconnected after 30 minutes. When 120 minutes passed, the connection was turned off due to no responses to keep-alive packets.

Therefore, through the Information Processing Center of Fukuoka University, we changed the timeout of firewalls from 30 minutes to 180 minutes. As a result, the connection between MCU and Fukuoka University became normal.

F. Connection failed (Kumamoto University)

[Problem] At the start of a lecture, only Kumamoto University was not able to be connected automatically by MCU. Kumamoto University used the same lecture room as before when normal connection had ever been successfully made. Moreover, the manager of MCU at QITO side could not make the connection manually either.

[Solution] The students in Kumamoto University then moved to another room. And this time, the MCU manager succeeded to connect the MCU with that room.

Since there was the possibility that MCU could not connect one of the two rooms, we decided to connect, as shown in Figure 3(b), MCU with both lecture rooms for Kumamoto University. As a result, if MCU cannot connect the first lecture room, the second lecture room will be tried (Section III-B). After students in the first room move to the second room, the connection should be established automatically.

G. The connection was down after 120 minutes (Kumamoto University)

[Problem] This problem was similar to the one of Fukuoka University (Section III-E). However, we did not find any problem about firewalls in Kumamoto University, and thus, we could not solve the problem by following the method discussed in Section III-E. Moreover we had no idea on how to solve this problem.

[Solution] We divided the connection duration into 2 parts. In the first part, MCU was set to connect QITO and the cooperative universities from 2:45pm to 4:30pm. In the second part, a connection was made from 4:30pm to 6:45pm. More precisely, the connection was turned off at 4:29pm and re-established at 4:30pm. Since the turning-off and re-establishment happened in the break between two lecture sessions, there was no influence to the lectures. Thanks to the above division, the connection between MCU and Kumamoto University returned to normality because the connection was turned off after 105 minutes. In addition, in the second part, the connection duration was 135 minutes in total, and thus, there would be no problem even if the lecture was extended over 6:10pm (when the connection duration should be 100 minutes).

H. Connection failed (University of Miyazaki)

[Problem] At the start of a lecture, University of Miyazaki had not been able to be connected by MCU.

[Solution] This problem occurred because IP address of University of Miyazaki was not registered in MCU correctly. Since the lecture could not continue as it was, we decided to let University of Miyazaki connect QITO directly (as shown in Figure 3(c)). Later, we registered the correct IP address of University of Miyazaki in MCU and the connection returned to normality in the form shown in Figure 3(b).

I. Sound could not be sent to other universities (Kumamoto University)

[Problem] It often happened that the sound of Kumamoto University could not be sent to other universities. We supposed that the cause of this problem is due to that Kumamoto University used PVX, a video conferencing application, but other universities used HDX9002, a video conferencing unit. However, we have not found the true reason for this problem so far.

IV. Questionnaire Survey

We have conducted a questionnaire survey on the remote lectures in 2011 and 2012. While we asked the students for filling the questionnaire after all remote lectures ended in 2011, the students were given the questionnaire before the first lecture started in 2012. This questionnaire consists of the following questions for each lecture.

1) Which part of the lecture are you interested in?
2) An overall impression of the lecture.
3) Comments for next year's lectures.

Although the primary purpose of the questionnaire is to obtain students' opinions about the contents of the remote lectures, we supposed that there might exist potential opinions about the environment or the method of giving lecture. After we reviewed the results of the questionnaire survey, as we expected, the results of the questionnaire did contain opinions about the environment of the remote lectures or the method of giving lectures. This questionnaire was conducted for the students who attended the remote lectures. They can decide whether to return the questionnaire by themselves. In Table I, we show the number of the students who attended the remote lectures and the number of answers we received. In this section, we give description of our analysis results.

Table I
ATTENDANCE AND ANSWERS

	2011		2012	
	1st semester	2nd semester	1st semester	2nd semester
Attendance	73	58	73	57
Answer	47	22	68	44

A. Opinions about the environment of remote lecture

The opinions about the environment of the remote lectures are summarized as follows.

- *I couldn't concentrate on the lecture enough because the network connection was disrupted. (Kumamoto University, the 1st semester, 2011)*
- *I wish the staff could improve the image quality of the camera. (Fukuoka University, the 2nd semester, 2011)*
- *I have to study by watching a DVD record of the lecture because a network disturbance happened in the lectures. (Kumamoto University, the 1st semester, 2012)*

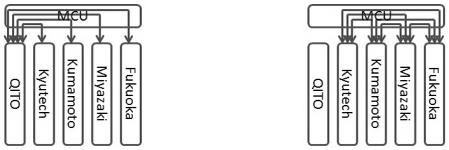
In 2011, some students who attended the remote lectures were dissatisfied with the lecture environment because some serious troubles occurred that interrupted the lecture (Section II-A). In the 1st semester of 2012, we introduced a dedicated equipment for MCU to try to improve the environment of the remote lectures (Section II-B). However we still got some complaints about it because there were various problems caused by a lack of knowledge of using MCU (Section III). In the 2nd semester of 2012, the problems, such as connection failure, did not occur. Then there was no complaint from the students for the environment of the remote lectures.

B. Opinions about the method of giving lectures

We had concentrated on improving the environment of the remote lectures from 2011 to the 1st semester of 2012. We thought that being interrupted was the biggest problem for a lecture. Although we achieved to improve the environment, some students left their opinions as follows.

- *I would be happy if the lecturer could listen to opinions of the students. (Kyushu Institute of Technology, the 1st semester, 2011)*
- *The lecturer should take care of not only Kyushu University but also the cooperative universities. (Kyushu University, the 2nd semester, 2011)*
- *I think that the lecture was conducted at Kyushu University only. (Kyushu Institute of Technology, the 1st semester, 2012)*

The lecturers did their best to accomplish their lectures. Thus, they sometimes tended to continue their lectures without listening to opinions of the students because they concentrated on explaining their themes. In addition, some



(a) Distributable path (b) Undistributable path

Figure 6. Camera information path

lecturers questioned the students in Kyusyu University but did not question the students in the cooperative universities.

Therefore, we requested the lecturers to ask questions to the students in all universities in the 2nd semester of 2012. Moreover, we asked the lecturers to designate in advance the students whom they expect to speak. These means allowed the students in the cooperative universities to speak positively without being left.

Furthermore, we obtained the following opinion.

- *It was difficult to read some letters or pictures drawn in whiteboards located at each university. (Kyushu Institute of Technology, the 1st semester, 2012)*

Some of the remote lectures used whiteboards. In such lectures, the information written in whiteboards at each university had to be distributed to each university. The information in Kyushu University's whiteboard can be distributed to the cooperative universities using the camera equipped in Kyushu University, and the information in the cooperative universities' whiteboards can be also distributed to Kyushu University using the cameras equipped in each cooperative university (Figure 6-(a)). However, the information in a cooperative university's whiteboard can't be distributed to other cooperative universities (Figure 6-(b)). Then, to distribute an image from a cooperative university to other cooperative universities, we distributed the image, which was distributed to Kyushu University, to other cooperative universities through the camera equipped in Kyushu University. Using this way, it was difficult for the students in the cooperative universities to read clearly the details on the whiteboard because the image was corrupt during Internet transmission.

Therefore, based on these facts, we decide to utilize digital cameras for those lectures that use whiteboards from 2013. The staffs of each cooperative university will be requested to take pictures of their whiteboards and send them via emails to Kyushu University, then we will distribute the whiteboard pictures to the cooperative universities using the same way as we transfer the *content* mentioned in Section I. We believe, in this way, the students in the cooperative universities will be able to read the images of whiteboards equipped in other universities clearly.

V. Related Work

In this section, we discuss two papers about the consortium of higher education in Shinshu, Japan. Eight major universities constitute the consortium: Shinshu University, Nagano College of Nursing, Saku University, Tokyo University of Science-Suwa, Seisen Jogakuin College, Nagano University, Matsumoto Dental University, and Matsumoto University.

A. Practice of "K^3 Salon"

"K^3 Salon" is an activity hosted by the consortium of higher education in Shinshu for practicing remote lectures and promoting communication among universities. K^3 is an abbreviation of the Japanese spelling (**K**outou **K**youiku **K**ouryu) of "higher education exchanging". The paper [5] reports results of constructing and operating a remote lecture system that is claimed to be easy-to-use by teaching and technician staffs themselves.

In this consortium, remote lectures among universities were operated by using dedicated equipment for MCU (RMX2000 of Polycom), which was located at Shinshu University. Staff members of each university used a touch panel to start a remote lecture manually and to end the lecture manually after 90 minutes. When a remote lecture was extended over 120 minutes, the firewall problem as described in Section III-E occurred and the staffs manually re-established the connection.

In Kyushu University, the dedicated equipment for MCU is managed by Information Infrastructure Initiative Center. QITO and the cooperative universities cannot connect to it forwardly. Instead, MCU connects to each university automatically and remote lectures are able to be operated easily. Moreover, in QITO, we deal with long-hours remote lecture.

B. Automatic Control and Extension of Control Devices

To solve the problems mentioned in paper [5], the paper [6] reported construction and implementation of "automatic remote lectures starting and ending", "automatic lecture content sending", "remote control of cameras", and "control program using iPad". In QITO, we record remote lectures just for those universities to which connections fail, not for later distribution. In addition, controlling remote cameras and using control panels through touch panels are not considered as important for us. Thus, we focus on "automatic remote lectures starting and ending".

In the consortium of higher education in Shinshu, duration of remote lectures is just 90 minutes. To operate the remote lectures, staffs start and end the lectures automatically. Due to this, even if a lecture is extended, the remote lecture (content distribution) will be turned off automatically before the real end of the lecture. Furthermore, when using RMX2000, an announcement will be sent to all cooperative universities 5 minutes before a remote lecture is to be turned off automatically. This announcement becomes a noise and makes it hard for lecture audience to listen to the lecturer clearly. The consortium solved this problem by delaying the scheduled ending time for 5 minutes.

QITO introduced the mechanism of automatic connection and disconnection as well (section III-G). However, the disconnection (i.e., turning off) is not made during a lecture. Since our remote lectures last for long hours, the disconnection is automatically made at break time in between two lecture sessions. Moreover, the announcement does not cause any harm because the automatic ending time is set to be 30 minutes after the supposed lecture ending time.

VI. Conclusion

This section concludes the MCU introduction (Section VI-A), the questionnaire survey (Section VI-B), and the whole of our paper(Section VI-C).

A. MCU Introduction

When we simply used MCU function, many problems occurred, which became obstacles to the remote lectures. To solve the problems, we decided to utilize dedicated equipment for MCU.

After that, the connection became stable, but some new problems occurred (Section III). These problems are divided into those caused by general network (Section III-A, III-C, and III-E), and those caused by device units (Section III-B, III-D, III-F, III-G, and III-H). We believe that our solutions to the problems of the second category are also applicable outside QITO.

The problems considered in Figure 5 comprise various problems mentioned earlier in this paper. As we continue operating the remote lectures, quality of the lectures becomes gradually better. After we introduced the dedicated equipment for MCU, the connection became more stable and lecturers could confirm now students of the cooperative universities who were answering their questions.

Although we have solved almost all problems encountered, the problem described in Section III-I has not been solved. The difference between the device units used by the universities or the construction of network may be one of the causes of the problem. In the future, we plan to further investigate solutions to this problem for providing more comfortable circumstance for remote lectures.

B. Questionnaire Survey

We conducted a questionnaire survey of the remote lectures, and then we obtained some complaints about the environment of these lectures (Section IV-A). Before confirming the opinions from the questionnaire survey, we had already recognized the problems described in the complaints. No complaints occurred after we solved the problems mentioned in Section II and Section III.

While we gave improvement of the environment of the remote lectures the highest priority, we obtained some opinions about the method of giving lectures (Section IV-B). The students in the cooperative universities became not to be left because we decided to propose some questions to these students consciously. To distribute among the universities the images of the whiteboards equipped in each university, we decided to distribute some pictures of whiteboards taken by digital cameras.

Thus, some complaints that we obtained from the questionnaire survey are expected to be resolved.

C. Summary

In this paper, we reported device setting experience obtained through holding remote lectures in QITO. Moreover, we have conducted a questionnaire survey and solved some problems realized from the survey.

We changed the environment and the method of giving the remote lectures in 2011 and 2012. Our work contributed to reducing students' complaints, that is, to improve the quality of the remote lectures. In our future work, we will investigate the method to improve the remote lectures in more detail, and consequentially increase the students' motivation of learning.

Acknowledgment

The research is carried out in the program of *Promotion Program to Foster Leading IT Specialists* supported by Ministry of Education, Culture, Sports, Science and Technology (MEXT), Japan.

References

[1] N. Sakamoto, M. Fukase, T. Mine, S. Kusakabe, T. Nakanishi, Y. Omori, M. M. Uddin, K. Araki, A. Fukuda, H. Yasuura, and T. Kitasuka: Large Scale Business-Academia Collaboration in Master Education Course. The proceedings of the First International Conference on Computer Supported Education (CSEDU), pp. 159-166, 2009.

[2] T. Oishi, Y. Kamei, W. Kong, N. Hiroshige, N. Ubayashi, and A. Fukuda: An Experience Report on Remote Lecture Using Multi-point Control Unit. 2013 International Conference on Education and Teaching (ICET 2013), pp. 352-359, 2013.

[3] Recommendation h.239. http://www.itu.int/rec/T-REC-H.239/ (refers at 2012.09.10)

[4] Recommendation h.323. http://www.itu.int/rec/T-REC-H.323/ (refers at 2012.09.10)

[5] T. Morishita, K. Chino, H. Suzuki, K. Nagai, M. Niimura, and M. Yabe: Practice of "K3 Salon" with Intercollegiate Distance Learning System in the Consortium of Higher Education in Shinshu. Journal for Academic Computing and Networking, Vol.14, pp. 105–116, 2010.

[6] T. Morishita, K. Chino, H. Suzuki, K. Nagai, M. Niimura, and M. Yabe: Development of Intelligent Intercollegiate Distance Learning Systems and an Extension of System Control Device. Journal for Academic Computing and Networking, Vol.15, pp. 70–81, 2011.

Empirical Development of a Mobile Application: UVA-Wise Undergraduate Software Engineering Capstone Project

I. Weissberger, S. Showalter, T. Deel, M. Ward, M. Whitt, and A. Qureshi
University of Virginia's College at Wise, Wise, VA, United States of America

Abstract - *Capstone projects allow students to put to use the knowledge they have acquired in the classroom. Projects such as these will serve to reinforce the concepts that may otherwise seem vague or abstract to students when they first encounter them. Capstone projects in software engineering give students the full experience of developing a product from gathering requirements to testing the software to verify those requirements. This paper presents a software engineering capstone project conducted by students at the University of Virginia's College at Wise. Their work and the methodology and tools that they used are recorded with a focus placed on the experience they have gained and additional methods which could be used to enhance the capstone experience.*

Keywords: Capstone Project, Software Engineering, Software Requirements, Software Design, Requirements Traceability, Project Management

1 Introduction

While students pursue degrees in software engineering, they may take several courses that relate to various aspects of the software development life cycle. Courses such as requirements, design, configuration management, project management and testing are all necessary because they teach students about important software engineering concepts that they will need to use in their careers. One of the drawbacks to some of these classes is that they teach concepts in relative isolation to other aspects of software development. In reality, many of these concepts go together. For example, requirements developed in the beginning of a project are traced throughout the course of the development effort to several documents, such as the design document and the testing document. Also, errors made early on during the requirements phase can have repercussions for later phases, which can often include rework to correct the problems. These important concepts might be lost if a student takes a course in requirements and completes a requirements development project but does not need to design, implement, and test those requirements.

Capstone projects allow students to connect the different phases of a software engineering development effort by working on a software project over two semesters. Students typically develop a new piece of software, so they are responsible for everything from requirements development to verifying the functionality of the software through formal test procedures. By taking a customer's high level business requirements and completing a product to satisfy those requirements, students can understand how all the different aspects of software engineering connect together. This serves as a complimentary activity to learning theory in the classroom. Often, through practice, students can gain a better grasp of the theory of software engineering rather than solely studying texts.

This paper presents work done during the Fall 2012 semester for the software engineering capstone project conducted at the University of Virginia's College at Wise. The software to be developed was intended to be the official application for the university. The Office of Media Relations and Communications was the customer and they had requested a mobile application which would allow users to gain access to information about the school. Currently information about the college is readily available, but it is difficult to find particular pieces of information due to the vast amount of college departments, services, and activities. Android was chosen as a development platform because the department already had mobile phones to support this and it was thought that the Android platform would be the easiest to develop software for. The paper focuses on the development methods the students used to complete the first part of the project (through the beginning of code construction) and could be used to start a template for future software engineering capstone projects.

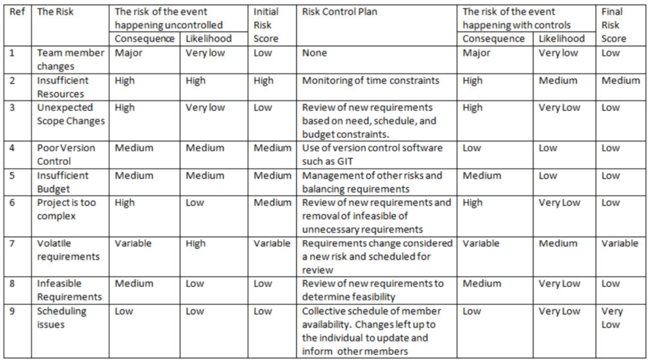
2 Project Management

The capstone project is designed to encompass two semesters. During each of the semesters, the students have to complete different software engineering related tasks. The first semester is designed to make the students complete requirements development activities, produce a design, and to begin coding. The second semester will have the students complete their coding and then verify the software works by running a formal system test. As with any software development effort, there is need for project management activities to track and correct issues to minimize their impact on the project [6]. This is especially true since the students have a fixed schedule that does not allow for very much slack in case the project should start to fall behind. Both semesters include project management activities. The activities during the first semester included creating a schedule complete with milestones, developing a risk mitigation plan, and conducting weekly status meetings. Since the capstone project is supposed to be completed within a given amount of time each semester, the students did not have much leeway in truly estimating how long tasks would take to complete. They had to work being constrained to one semester. However, in creating the schedule, they have learned that it can be a valuable tool to track program status against pre-determined milestones.

A risk mitigation plan was developed by the students to think of different scenarios which could happen over the course of the project that would impede their progress. Risks were identified by team members individually and then combined together in a brainstorming session. Each team member came up with a list of risks associated with the project with an estimation based on the probability of the risk occurring and how much of an impact the risk would make on the project. The list of risks were then compiled and discussed to determine final weights associated with those risks. Risks were identified to have two main criteria, probability and impact. Probability was the chance of a risk happening based on a scale of 100. Impact was how fatal the risk could be to the project, on a scale of 10. The risk was then given a score based on the formula as shown in (1).

$$Score = \left(\frac{Probability}{10}\right) * Impact \qquad (1)$$

This means that a risk with 30% probability and an impact of 8 had a total score of 24. The scores were then grouped together into general categories ranging from “very high” to “very low”. The list of risks identified is presented in Figure 1.



Ref	The Risk	The risk of the event happening uncontrolled		Initial Risk Score	Risk Control Plan	The risk of the event happening with controls		Final Risk Score
		Consequence	Likelihood			Consequence	Likelihood	
1	Team member changes	Major	Very low	Low	None	Major	Very low	Low
2	Insufficient Resources	High	High	High	Monitoring of time constraints	High	Medium	Medium
3	Unexpected Scope Changes	High	Very low	Low	Review of new requirements based on need, schedule, and budget constraints.	High	Very Low	Low
4	Poor Version Control	Medium	Medium	Medium	Use of version control software such as GIT	Low	Low	Low
5	Insufficient Budget	Medium	Medium	Medium	Management of other risks and balancing requirements	Medium	Low	Low
6	Project is too complex	High	Low	Medium	Review of new requirements and removal of infeasible of unnecessary requirements	High	Very Low	Low
7	Volatile requirements	Variable	High	Variable	Requirements change considered a new risk and scheduled for review	Variable	Medium	Variable
8	Infeasible Requirements	Medium	Low	Low	Review of new requirements to determine feasibility	Medium	Very Low	Low
9	Scheduling issues	Low	Low	Low	Collective schedule of member availability. Changes left up to the individual to update and inform other members	Low	Very Low	Very Low

Figure 1 – Identified Risks

Each week, the students would meet to discuss any new risks and to recalculate the total scores of each identified risk as a means of monitoring them. Any risk with a change in total score greater than 10 would be marked and further actions discussed to manage the volatility of the risk. In addition, each type of risk had its own monitoring activities and requirements. For instance, scheduling risks were monitored through a collective schedule of each team member’s availability and each member was responsible for informing the team of any changes that needed to be made to their portion of the schedule. The students also developed risk mitigation plans based on the type of risk. Continuing on the previous example, to mitigate schedule risks each team member must keep close tabs on their schedules and inform the other members of the team of changes in their schedule in a timely fashion as well as take steps to ensure flexibility in their scheduling.

Status meetings were also held weekly so that students had a chance to discuss their status and any issues that were impeding their progress. Each team member was responsible for presenting a list that included what they had worked on in the past week, their plans of what to work on for the following week, and any issues they had encountered that were holding up progress. This allowed the team to learn about issues and correct them before they became a bigger problem which could greatly affect the project schedule. These items were presented to the group so that each team member was kept up to date on each part of the project. The team understood what was completed, what was being worked on, and when these were supposed to be completed, as well as what hasn't started. With this information, the team could compare the items to the schedule and figure out what needed to be changed to keep the team on track. Items are listed on a percentage complete scale from 0 to 100. This detailed the progress done to the item in order to provide more information for the team.

3 Requirements

In order to develop a set of requirements for the project, the students had to perform all four phases of requirements development: elicitation, analysis, specification, and validation [1]. Rarely are all these four phases performed in a one-time linear sequence. Most of the time, the phases are performed in an iterative fashion [2]. This benefits both the software development team and the customer because rework done during the requirements development phase is less costly than rework at the later stages of the software development life cycle.

The students started the requirements development phase by having a meeting with the customer to go over what they wanted from the software. At first, they only retrieved very high-level requirements. The initial requirements included having the application retrieve and display the following items:

- University news feed
- Campus directory
- Campus map
- Directions to the campus
- University sports feed
- Alumni information
- Campus dining information

The students met with the Office of Media Relations and Communications several times to elicit requirements. They then scheduled a more detailed interview with the customer in order to formulate an adequate list of requirements. The second meeting with the customer was more informative, and the students were able to start a list of requirements to review with the customer. They took those requirements and considered how they would design the system around them and their feasibility. The students also encouraged the customer to think of more requirements during the time between the second meeting and the third meeting so that they could consider all the requirements before the design phase. To capture the expected behavior of the system from a user's point of view, the students developed use cases to model the interactions between the user and the software. These use cases formed the early part of testing done in parallel with the coding done during the first semester of the capstone project. The use cases would also be used to develop unit tests for the second semester and the test document which would be used during formal system testing.

The third meeting reviewed the requirements from the previous meetings and the students also retrieved new ideas and requirements that the customer had thought of since the last meeting. By encouraging the customer to think of any extra requirements early in the requirements gathering phase, the students hoped to avoid any costly additions of extra requirements in the design and coding phases. This was successful because they were able to retrieve a good comprehensive set of requirements. The fourth meeting covered finalizing requirements and reviewing any issues with them. The students and the customer discussed any issues with resources and the feasibility of specific requirements. They also unofficially finalized the list of requirements that were gathered to that point.

The final meeting of the requirements gathering phase concerned the formal finalization of the requirements that had been gathered. This included reviewing the entire software requirement specification (SRS) document and noting any missing items. The SRS was based off the IEEE standard 830-1998 [4], and the students had tagged each requirement to provide requirements traceability for other documents to be constructed, such as the software design document and the software test document. The final meeting concluded the requirements phase and the students proceeded to the design phase. During the requirements review, the customer signed off on the requirements, so that they had an understanding that the SRS represented the scope of the project.

Over the several meetings that the students had with the customer, the list of the requirements grew. The finalized requirements had all the initial requirements included plus these additional features:

- Social media site links
- Student services links
- Convocation center website access

It was beneficial to both parties to agree to baseline the requirements so that would narrow the expectation gap between what the students were developing and what the customer was expecting.

4 Design

Once the requirements had been baselined, the students began to create the software design based off the requirements. The students used the IEEE standard 1016-1998 [5] to create the software design document (SDD). The SDD contained a requirements traceability matrix so that it was ensured that no requirement was overlooked when making the design. The system was built in three layers: the user interface, logic, and communication. As shown in Figure 2, each of these layers had several corresponding classes that would need to be implemented.

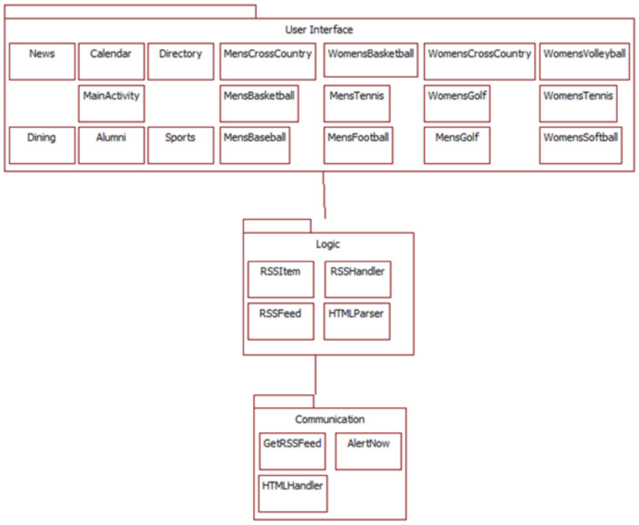


Figure 2 – Subsystem Packages

Due to the unique architecture of the Android operating system, the user interface must be combined with some logic and communication classes. The user interface layer displays all items on screen that the user will need to control, understand, and navigate the application. This layer also holds all user interface items that the system uses. The user interface layer is also a container for the logic layer, and sends input to each of the logic layers functions as they are needed. The logic layer is what holds the methods and classes needed to perform each system action. This layer also holds the data needed to communicate with the user interface layer and the communication layer. The logic layer is also the main level of the application, since it performs the necessary functions. In order to design the interactions between the different layers, the students created sequence diagrams similar to the example shown in Figure 3. The application was designed in a layered architecture due to the need for only three levels of system performance. The user needed to perform an action, the system needed to respond and obtain the data required, which needed a communications level, and then it needed to display that data. The students chose this because it fit the application's needs perfectly and works well with the way the Android OS is designed.

The main tool used for the user interface design was Adobe Photoshop CS5.1. The icons for the different pages were designed so that their purpose would be easily recognized. Most icons are original designs with the exception of a few logos. The style of the icons is designed to look like frosted glass buttons which is based on current popular mobile phone applications. Photoshop allowed the students to quickly mock up what a screen would look like in the finished project without having to develop it in an Android application. Based on the functionality required for each screen to operate correctly and fulfill the requirements completely, classes were able to be developed for each screen. The home page contains icons that direct the user to the different pages. The icons are designed so that the user can quickly recognize what type of information which is displayed on the page where the button will direct them. The icons include news, calendar, directory, dining, maps, sports, Moodle, bookstore, the weather channel, and social media. The home page is shown in Figure 4.

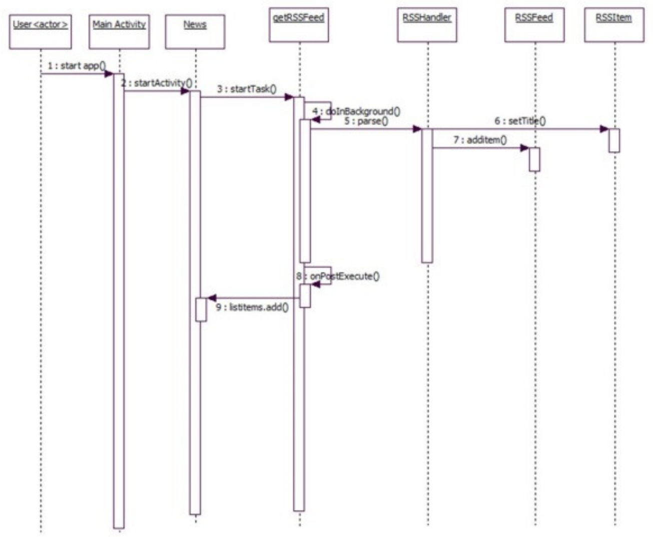
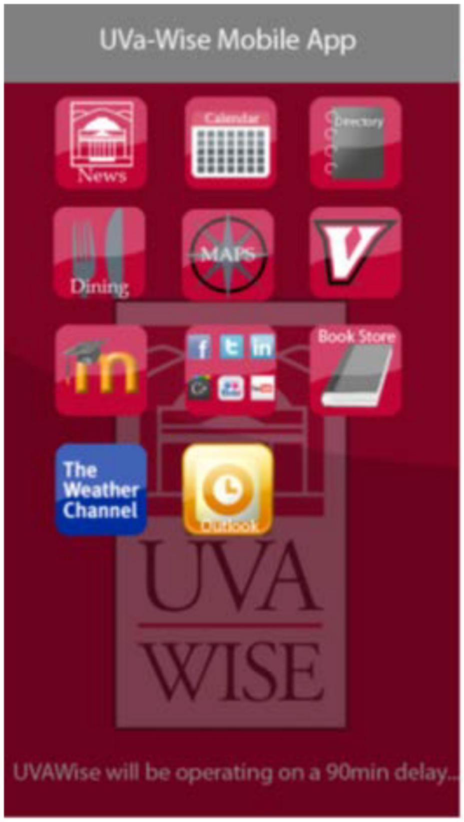


Figure 3 – Sequence Diagram

The bottom of the home page will also contain an information and alert crawler. Text will scroll across the bottom of the screen letting the user know the important messages without having to click any icons. Most icons will direct the user to a filter page to filter out the information that is displayed on each page. The pages that will have a filter page include the news, directory, dining, maps, sports, and alumni icons. The directory, news, and sports pages will contain filters as well as a search option that will be at the top of the page.

5 Code Construction

The students began the coding phase by establishing coding standards. They proceeded to code the project based on the class diagrams that had been developed and that were included in the software design document (SDD). They had also developed a general prototype to establish flow through the application. This first prototype was intended to contain a generic user interface, but function as a final product in the flow of the application. This prototype contained placeholder buttons and icons that were linked to empty screens to ensure that the application worked properly and that each major requirement had a screen that would contain the correct information and functionality. After this prototype passed preliminary tests and was reviewed, it was saved, base lined, and a new iteration was started. This new iteration of the prototype was intended to perform basic tasks and featured the finalized user interface so that it looked and felt like the final version should. As the prototypes were developed, each time a method was changed or added, a unit test was performed to ensure that the system worked correctly and was stable. The students approached the project with the philosophy of extensively testing each method to ensure that every part of the application worked on the most basic level.



UVAWise will be operating on a 90min delay...

Figure 4 – Application Home Screen

6 Configuration Management

A configuration management process was established to maintain the software and all supporting documents and information for the project. The students used Git [3], an open source tool, to manage changes to files and documentation for the project. They also devised a change control process, as shown in Figure 5, in order to evaluate change requests, whether they came from the client or a team member. Change requests were evaluated by the team, which consisted of discussing alternatives to the change and whether or not it was necessary. If deemed necessary and the best way to make the change, the change was implemented into the system. From there, the change was validated to ensure it was the right way to correct or improve the previous implementation. If the change was validated it would then be incorporated into the system. Change management was essential in deciding what the schedule could handle as new requirements were added to the system. Early on the decision was made to incorporate a reporting and analysis section to the system, and through the change management process the students were able to evaluate the functionalities worth and determine what it provided the system and customer.

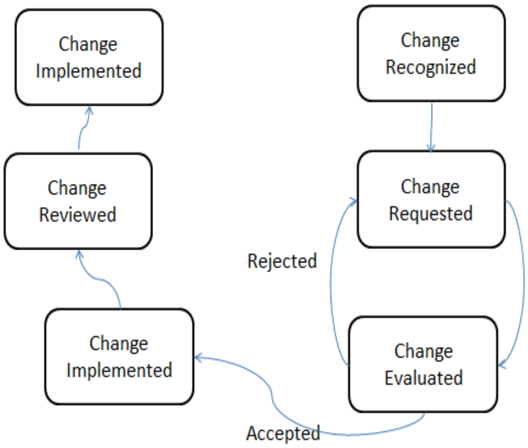


Figure 5 – Change Control Process

7 Conclusions

This paper presented the first half of a senior capstone project that focused primarily on gathering requirements and producing a suitable design. The code construction commenced in the first semester of the project, but will not be completed until the following semester. The students gained a great deal of experience by having to develop a product from the beginning by using the concepts they have learned in taking software engineering courses. Having a consistent process in place benefited the students by informing them what was expected. It would be beneficial for a university teaching software engineering to have a defined process and document templates that capstone projects could use. This could be similar to an organization developing its own software process to make it more consistent between various projects in the organization. The process may then be modified depending on the project type.

8 Acknowledgements

The authors would like to express a sincere appreciation to The Office of Media Relations and Communications at the University of Virginia's College at Wise for support of the capstone project presented in this paper.

9 References

[1] Introduction to Software Engineering Body of Knowledge, http://www.computer.org/portal/web/swebok/html/ch2, Last accessed on March 8, 2013.

[2] Karl Wiegers, *Software Requirements 2nd Edition*, Microsoft Press, 2009

[3] Git, http://git-scm.com/, Last accessed on March 8, 2013

[4] IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications, IEEE, 1998.

[5] IEEE Std 1016-1998, IEEE Recommended Practice for Software Design Descriptions, IEEE, 1998.

[6] Introduction to the Software Engineering Body of Knowledge, http://www.computer.org/portal/web/swebok/html/ch8, Last accessed on March 8, 2013

Code Debugging Tool to Help Teach Mainframe Programming Languages

Claudio Gonçalves Bernardo
MSc. Computer Engineer
Universidade Paulista
Brazilia – Brazil
claudiogbernardo@ig.com.br

Paulo Roberto Chineara Batuta
MBA Software Quality
Centro Universitário Araraquara
São Paulo - Brazil
pbatuta@hotmail.com

***Abstract* – The purpose of testing activity is to discover errors in software and for this objective to be achieved many activities must be performed. There are several types of test and while some are generated and focused on functional verification of a component and incorporating components in the structure of the program, others are applied to point traceability requirements. There are systems tests designed to validate part of a software component, which is a part of software solution greater. Its function is only to be a small part of a more complex reality. The different types of tests are complementary and in most cases are applicable to all software developed. All aim to develop programs with the fewest mistakes as possible, allowing the product to deliver the desired quality. The activity of preparing professionals in the use of language for Mainframe environment is difficult mainly because the programs need to test both the online environment as batch. This work presents a modern tool for debugging source code developed in Mainframe platforms. It presents the tool and its installation. Its use in a teaching unit of the programming language for Mainframe environment becomes essential as it allows the student to know the procedures of preparation and test execution step by step. This activity will prepare students better for becoming a member of a team developing applications large and it allows to accurately meet the requirements presented by Software Quality area of any institution, whose intention is to deliver the final product with minimum possible errors meeting all the requirements generated by the final customer in the shortest time.**

Keywords: Software Testing, Preparing for IT Industry, Debugging Tool, Mainframe, Innovation in classroom.

I. Introduction

Software development is an activity that involves several stages, including the testing phase. The test phase is assigned responsibility for finding the largest number of possible errors, avoiding that they are already found by the client when implanted [16]. Because it is a human activity, has a high probability of insertion errors. On average, 10 errors can be found for every 1000 lines of code (1 error per KLOC - Kilo Line of Code). The test software allows errors to be identified before being delivered to the customer, a situation that enables correction of the problem; because this fact allows the parties involved (customer and supplier) enter into agreement on the solution or consequences caused by error identified.

The activity of software testing can be considered as one of those contributing to the improvement of quality, since the software tends to have improvements compared to its previous state. Software quality must consider not only the occurrence of error but also the software serves the purpose perfectly expected by the client. It is important to check the quality of software at various stages of development, not only restricting the testing phase of the software.

An appropriate test is one that discovers the greatest possible number of faults in the software and when this software is designed to run in an large environment, also known as Mainframe, becomes an even greater difficulty, since this environment is complex and the characteristics of each facility are different. Universities see this difficulty as a major obstacle to the teaching of languages such as Cobol, PL/I, Easytrieve, Assembler and others, who did not die and will not die, as a single system mainframe can deliver thousands of virtual Linux platforms with a small fraction the dollar cost, power and space [20]. The preparation of students for the global IT industry becomes very difficult. [12] states that it is necessary that the professional has complete mastery of the scenario where the tests will be carried out and for this to happen should apply innovative uses of technology in the classroom, thus seeking to reduce the learning time with a closer view of the possible day-to-day industry produces software for Mainframe environments.

II. Software Test

The need of Software Test before delivery, whether Individual Test, System Test, Integration Test or Test Certification is a notorious issue in all organizations that develop IT solutions. To ensure that the product is delivered according to specified was necessary that the IT market suffer some facts of reality. According to research presented by [11], 50% of companies surveyed 25% or more of IT projects are completed late, 57% of companies in each two IT initiatives, generated only a positive business results. Companies that can accelerate the completion of projects and IT services acquire a significant profit margin, while the others were not capable of it are victims of ruthless competition.

The activity of Software Testing performed in a controlled and disciplined performance ensures superior to software product delivered, due to the reduced number of errors when it enters the production phase. In [14] researchers argue that the developer should use software agents for prioritizing test cases in order to maximize the number of faults found during the time available for testing with a particular order of priority. In [1] is presented research that provides a systematic review of the use of statistical tests to assess randomized algorithms in software engineering. Here the aim was to show that randomized algorithms are used in a significant percentage of projects and that in most cases do not appear in searches. There are many statistical tests based on different subjects and not always peaceful spot when and how these tests should be used.

In [13] researchers report cases of manual tests are time consuming and are prone to many errors. They argue that tests based searches can help in automating this process by reducing the time and effort to automatically generate test cases and complex data for relevant tests.

They claim that research has focused on programming languages and statistical data inputs for simple tests such as input of numbers, but this practice is not for dynamic programming languages that are widely used by developers.

A. Defects, Errors and Failures

To be presented some elements of the discipline of Software Testing is necessary to identify some concepts relevant to it. The first one is to interpret what constitutes a defect. This is inconsistent act committed by an individual trying to understand a piece of information, solve a problem or to use a method or tool. An instruction or incorrect command within a source code illustrates the concept categorically Defect.

Error is a concrete manifestation of the defect described in the preceding paragraph, when this happens in a software artifact. The difference between the value obtained and the expected value, representing any intermediate state incorrect or unexpected result in the execution of a program is a mistake. Failure is the operational behavior of the software different than expected by the user. It can be caused by several errors, but it is important to note that some errors throughout the lifetime of a software solution, can never cause failure.

The defects are considered part of the physical universe, which is the software application in a real and are caused by people, through the misuse of any technology. These can cause errors in the manifestation of a product, in building a software differently to what was specified. Errors generate faults, interpreted as unexpected behaviors in a software that directly affect the end user of the application, so you can undo all the generated worked to develop the same.

Some factors such as incorrect understanding of user needs, the lack of ability to deal with changing requirements and the discovery of serious problems in the final stages of the project lead to failures in software [16]. Faults can be considered as incorrect results regarding specification, obtained by using the software. The consequences of failure are faults inserted into software. The defects are characterized by being representations of human activity executed in error, and are representations of human error committed.

The defects can be classified according to the degree of severity of failures which cause, establishing criteria for evaluation. They are classified into:

- Smooth: failures that offends the aesthetic sense;
- Moderate: failure due to redundancy;
- Hater: faults that generate disagreement about the human standard;
- Disruption: failures that lead to no processing some legitimate transactions;
- Seriously: failure that causes loss of transactions;
- Very Seriously: dangerous failure results completely wrong;
- Record: whose fault occurrence becomes frequent;
- Intolerable: speech that corrupts a database;
- Catastrophic: generates a system shutdown;
- Infection: Failure generating shutdown on a system that propagates to other systems

The discovery of defects in the software presents a high cost to the project, increasing substantially as the project progresses through the phases of development. The later a defect is discovered the greater the cost to repair it.

According to [4], a failure can be caused by various reasons, such as the specification that may be incorrect, incomplete or unclear, it may contain requirements impossible to implement due to hardware limitations or software or even by sheer nonsense of requester, that generated the requirements. To ease the difficulty between the end user and the creator of the requirements was established requirements engineering. Another reason for the occurrence of a fault may be the existence of a database totally disorganized, featuring clearly the lack of preparation for mass screening appropriate to consider all possible ways that the conditions of business rules may lead. Finally, failures may also be the consequence of defects in deployed script prepared for the tests. The student of computer programming courses at a university or even training companies need to have the exact idea of the importance of testing for the software industry, need to understand how this industry suffers if requirements are generated so inaccurate as this inaccuracy directly affects the results that are expected of program testing to be performed.

B. Test Features

It is extremely important to identify the different features that the Software Testing presents. Not just a matter of nomenclature, but to ensure that all possibilities to interpret this discipline are reflected in software development. According to [5] some project leaders understand that Software Testing is an activity that should be handled only at the end of the development process and is perceived that the delay in detecting errors makes the delayed repair and costly. So spending on financial correction could be used for other project activities. The test becomes of great importance in the development, as it allows the hits are performed before implementation.

The set of principles, described below, serve as a guide to make the most effective software testing:

- All tests must be related to customer requirements;
- Tests should be planned long before it starts;
- The Paretto Principle applies to software testing. This principle implies that 80% of all errors discovered during the test tend likely to be related to 20% of all program components therefore must be isolated suspected components and test them closely
- Testing should begin in the individual components, towards the entire system;
- The full test is not possible;
- To be more effective the test should be conducted by someone other than the developed.

One of the important features and testability, which is the ability of the software, has to be tested. The Operability ensures that the better the system work better will be tested. The Controllability ensures that the more you can control the software more testing can be automated and optimized. The Observability is the ability to allow monitoring of the test by observing the inputs and outputs. What you see is what you test each input and generates an output distinct, facilitating the identification of nonconformities. The source code is available. Decomposability is characteristic of isolating parts of the system to test. Controlling test scope, one can quickly isolate problems and make further tests more rationally. Simplicity, which has the view that the simpler the faster tested. Even as we feature Understandability, where more information enables greater rationality to run the test, and to finalize the Stability because fewer changes cause less disruption in the software.

C. Steps of Test

At the time of student software's construction, whose pretends to be a software developer, aware of the number of observations, contributes to the implementation of a good test, which shows a high probability of finding errors. When preparing a test he should have as much information as possible about the software and know the points where this software can fail. A good test is not redundant, it should be very simple or very complex. In general, each case must be run separately, to avoid masking of errors.

It is necessary for the effective functioning of the activity of the test run are followed - in this order - the steps:

a) Planning, where are certain test cases are chosen and the data and determined the expected results;

b) Test Execution, that is the time they are established scenarios, test scripts and instrumentation programs;

c) Evaluation of the results to decide whether some changes should be made to the code tested. This step is performed to compare the results computed with the expected results and reports generated notes. Any institution responsible for the development of software must be able to establish a strategy for testing. This means defining what are the patterns, techniques, models and criteria to be used to allow software testing.

A good test strategy should:

- Contain the activities to be completed to the test;
- Establish criteria for evaluating the software;
- Know the types of test to be applied;
- Have adequate data to generate results conference;
- Start from the most basic level, progressing to reach the software as a whole, including integrations with other software;
- Have flexibility to allow creativity and customization;
- Allow monitoring and compliance test plans outlined.

D. Test Types

There are some types of tests that can be applied by the developer or by the customer and can be performed for each program, only the system or application across the organization, also the integration of this system with its other systems, with the aim of bringing up as much as possible to the reality of implementation in production, final phase in which the software is available for use in day-to-day [16].

The test types are:

a) Structural Test, which seeks to find bugs in the code, known as the program structure;

b) Functional Test, aiming to find discrepancies between what is expected to result versus the actual result presented;

c) Integration Test, which verifies the behavior of software during interfaces;

d) Regression Testing, which is focused on software maintenance, seeking to ensure that no new defects were introduced in new versions presented;

e) Validation Test, which identifies whether the software works according to customer expectation, if it was built meets user needs. This test is also known as Acceptance Test or Acceptance;

f) Alpha Test, where the user performs the test in the development environment, along with the developer. Is to bring the user to the testing environment and allow contact with this software developed for the developer to register the errors and problems encountered;

g) Beta Test, where the user receives the test version and report any problems found. Occurs on the desktop software user and the client reports problems to the developer to make the corrections;

h) Recovery Test, which is the system's ability to recover and return the processing failures within a pre-specified time. There are cases where a system must be fault-tolerant, being able to continue processing without stopping global operations;

i) Test Security, which verifies the software's ability to protect itself from invasion improper or illegal, harming or benefiting someone wrongly. This test attempts to verify that protection mechanisms built into the system will actually protect you from invasion;

j) Stress Test, which verifies the software's ability to react to situations and data processing quantities and extreme high frequency of use or application of abnormal volumes;

k) Performance Test, which checks how the software reacts during execution. It is designed to test the performance of the software when running in the context of an integrated system;

l) Test Unit, also known as Unit Testing, which aims to explore the smallest unit of the project, seeking to provoke failures caused by defects in logic and implementation for each module separately. The target universe of such testing are methods of objects or even small code snippets;

m) Integration Test, which aims to cause failures associated with the interfaces between programs when these are integrated in accordance with what was designed in the architectural phase of the project;

n) Test System, which evaluates the software for faults through the use of the same, as if an end user. Thus, the tests are run in the same environments, with the same conditions and with the same input data a user would use in their day-to-day handling of the software. This activity allows you to check if the product meets the customer's requirements;

o) Acceptance Testing, which are usually performed by a small group of end users of the system, simulating routine operations of the system to verify that their behavior is in accordance with the request;

p) Regression Testing, which is a strategy to reduce side effects. This type of test is applied to each new software version or every cycle, all the tests that have been applied in versions or test cycles previous system. This is a test procedure to be applied after the program has been modified since aims to ensure that no new defects were introduced during maintenance, checking that maintenance performed corrected the defect or functions according to the modification required;

q) Structural Testing or White-Box Testing. This is based on the internal architecture and the software is to ensure that all independent paths from one module to run at least once, and ensures that all logical decisions have been treated, also ensures that all cycles were performed within its boundaries and within their operating ranges; ensures that internal data structures were implemented to ensure its validity. Structural testing seeks to find factual errors in code;

r) Functional Test or Black Box, which serves to validate the expected results with the results. This test is performed based on the requirements of software obtained from the client and seeks to identify gaps in functionality, integrations and total software;

s) Integration Test, which serves to ensure that data is not lost during the interfaces between modules, or that one module does not have an effect on inappropriate or that another combination of sub-functions fail to generate the desired function;

There are testing environments, architectures and specialized applications, due to the wide variety of capabilities and software application areas become necessary to identify errors that other types of test ignore.

III. The Importance of Testing Unit and Use of a debug tool

Some people still wonder why to write Unit Test. The fact that should be considered relevant is that the effective use of Unit Testing considering its realization through appropriate testing tool can avoid basic programming errors and allows the coverage of code written without using mass data. A unit test should be able to examine the behavior of the code under the most varied conditions. The behavior of the code can be analyzed by the content of the parameters passed to it, i.e. it performs and what it returns if a condition is true or not. Furthermore, evidence through which debugged code is indeed important. With it, we can consider what is called code coverage. A program which shows its cover code by means of a test tool is guaranteed that were performed types of tests made by the developer.

A. Debug Tool for z/OS to Completion Test Unit and Coverage Code

The companies that develop IT solution to launch every year lots of tools to aid in development. For almost all activities of the phases of software development tools are ready for use. In Phase Test Software is no different.

One of the tools designed to debug individual tests is the Debug Tool for z/OS. z/OS [7] is the Operating System Mainframe environment. According to [8] Debug Tool for z/OS allows you to examine, monitor and control the execution of programs like C, C + +, COBOL and PL/I that are performed in the operating system z/OS. It is an interactive tool for source-level debugging for applications compiled in a variety of environments.

It supports batch processing and on-line, stored procedures in the database DB2, IMS, and UNIX System Services. It also supports a seamless debugging of applications in languages mixed in the same session and offers a set of commands that can be interpreted used to specify actions to be performed. It also offers the option of setting breakpoints in an application program, monitoring changes to variables and noting exceptions and conditions specified during program execution and counts how many times a statement or verb was processed in an application program. Its integration with other development tools enables developers to debug applications in an integrated environment.

Figure 1 shows a screen printing with the operation of the tool interface Debug Tool for z / OS with the application program.

Other features of the tool Debug Tool for z / OS are:

a. Level debugging code;

b. Multiple breakpoints;

c. Conditional;

d. Unconditional;

e. Debugging step by step;

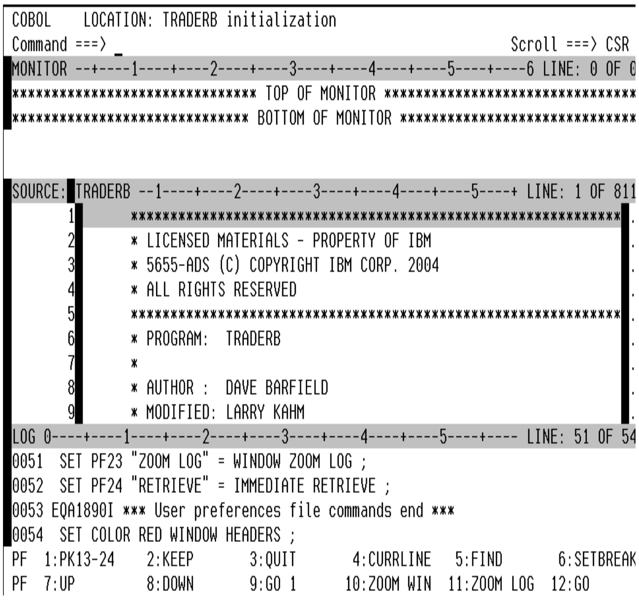
f. Dynamic changes;

g. Modifications of variables;

h. Entering commands;

i. Log of the commands used in the debugging session;

j. Cover tested code.



```
COBOL    LOCATION: TRADERB initialization
Command ===> _                                          Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 0 OF 0
****************************** TOP OF MONITOR ******************************
***************************** BOTTOM OF MONITOR ****************************


SOURCE: TRADERB --1----+----2----+----3----+----4----+----5----+ LINE: 1 OF 811
      1       **************************************************************.
      2       * LICENSED MATERIALS - PROPERTY OF IBM                        .
      3       * 5655-ADS (C) COPYRIGHT IBM CORP. 2004                       .
      4       * ALL RIGHTS RESERVED                                         .
      5       **************************************************************.
      6       * PROGRAM:  TRADERB                                           .
      7       *                                                             .
      8       * AUTHOR :  DAVE BARFIELD                                     .
      9       * MODIFIED: LARRY KAHM                                        .
LOG 0----+----1----+----2----+----3----+----4----+----5----+---- LINE: 51 OF 54
0051  SET PF23 "ZOOM LOG" = WINDOW ZOOM LOG ;
0052  SET PF24 "RETRIEVE" = IMMEDIATE RETRIEVE ;
0053 EQA1890I *** User preferences file commands end ***
0054  SET COLOR RED WINDOW HEADERS ;
PF  1:PK13-24    2:KEEP      3:QUIT      4:CURRLINE   5:FIND      6:SETBREAK
PF  7:UP         8:DOWN      9:GO 1     10:ZOOM WIN  11:ZOOM LOG  12:GO
```

Figure 1 - Interface with an Application Font: [10]

B. How Debug for z/OS evidence Coverage Code

The code coverage is evident in a practical manner, i.e., calculate the percentage of code coverage by dividing the number of lines tested by the total lines of the program multiplied by one hundred. Therefore a basic tool that is present by indicating which line of code was executed and how many times it was executed. Figure 2 shows how the code coverage as evidenced in the tool Debug Tool for z/OS. The code coverage is evident in a practical manner, i.e., when it terminates execution of white box testing, the tool executes a command Debug Tool and it immediately makes the calculation of percentage of coverage in your area and displays the log.

This enables the programmer to validate the quality and made his test. An area of quality can validate the test results both by the evidence of the data generated as well as the percentage of code coverage that was stored by the tool.

IV. Neeed Test of Code Source to Ensure Software Quality

The software development organizations have a need to test your source code as a prerequisite for ensuring the quality of software being delivered. An area of software development of any organization needs to create an Environment of Testing and Certification meeting ITIL recommendations [15][3] and the requirements of Basel 2 [2]. His necessity exists for improvement of software quality and reducing problems in production. The organizations intend to achieve this by improving the performance of existing applications in production as well as prevent new applications are poorly constructed. Most of these institutions create areas distributed in its Department of Information Technology, which usually are: Process, Methodology, Infrastructure, Systems and Production. The objectives of these areas are:

a) **Processes**

Reevaluate and set the new development cycle, contemplating the homologation environment and preserving the production environment;

Adapt the process of the development cycle while keeping the pole developer in distributed environment;

Draw the new path of developing software contemplating new instance of approval;

Set path emergency software development considering environmental approval;

Define types of development according to business need, i.e. define which systems or programs need to be approved;

Define indicators for a development, integration testing and homologation tests;

Ensuring traceability;

b) **Methodology**

Reset the types of tests for each instance of development;

Reset the deliverables of evidence validation of software testing;

Reassess criteria for which systems or programs need to be approved or not;

Develop training for the various areas involved in the new process.

c) **Infrastructure**

Draw the usage model of software tools that meet the requirements of new processes involving development environments, approval and production;

Training resources on the use of new software tools;

Differentiate libraries approval;

d) **Systems**

Make the activities included in the development process regarding the approval and development environments;

Use productivity tools and problem determination in development environments and approval;

e) **Production**

Validate evidence of acceptance of the approval;

Deploy the changes in the production environment;

Controlling the life cycle of software deployment predicting the back out.

To achieve these objectives it is necessary to perform the following tests:

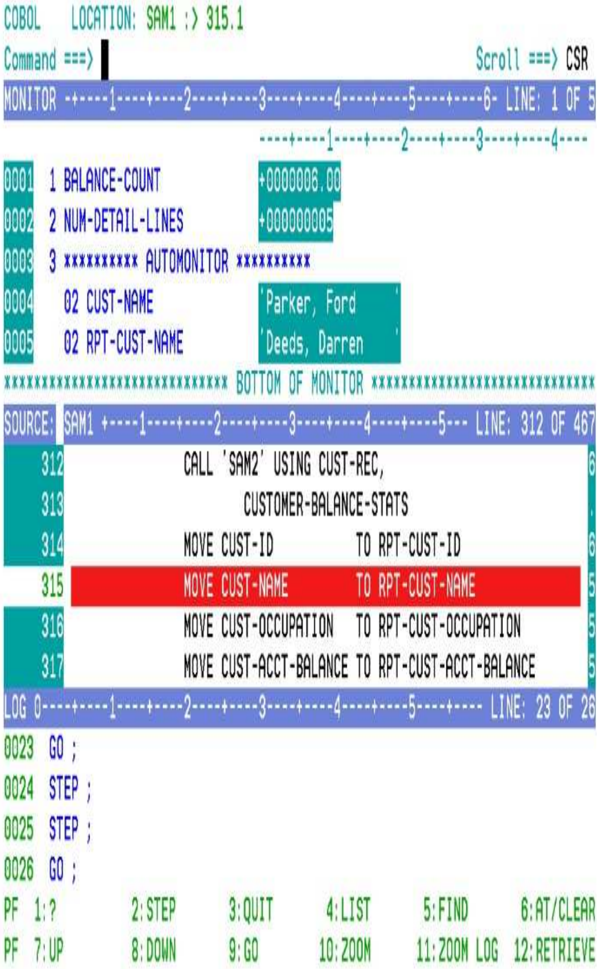
i) Unit - To be done by programmers; positive and negative; white box, with evidence of testing in the development environment, and an uncontrolled environment..

ii) Functional - Black Box - Regression - With test cases in test environment integrated, evidence also generating tests. This atmosphere should be a controlled environment.

iii) Performance - Stress and load being made in the homologation environment, generating evidence test, being a controlled environment.

V. Use of Testing Debugging Tool for Teaching in Mainframe Environment

Training courses in the area of Brazilian universities of Information Technology do not have computer labs in large scale, also known as Mainframe machines, and for this reason are unable to train future professionals in this environment. According to [17] the Information Technology industry has a demand for professionals is not answered, and this because of the deficiency that universities have and the low number of trained professionals in their own organizations.



```
COBOL    LOCATION: SAM1 :> 315.1
Command ===>                                          Scroll ===> CSR
MONITOR -+----1----+----2----+----3----+----4----+----5----+----6- LINE: 1 OF 5
                               ----+----1----+----2----+----3----+----4----
0001  1 BALANCE-COUNT          +0000006.00
0002  2 NUM-DETAIL-LINES       +000000005
0003  3 ********** AUTOMONITOR **********
0004    02 CUST-NAME           'Parker, Ford      '
0005    02 RPT-CUST-NAME       'Deeds, Darren     '
*************************** BOTTOM OF MONITOR ****************************
SOURCE: SAM1 +----1----+----2----+----3----+----4----+----5--- LINE: 312 OF 467
     312             CALL 'SAM2' USING CUST-REC,                              6
     313                  CUSTOMER-BALANCE-STATS                              .
     314             MOVE CUST-ID          TO RPT-CUST-ID                     6
     315             MOVE CUST-NAME        TO RPT-CUST-NAME                   5
     316             MOVE CUST-OCCUPATION  TO RPT-CUST-OCCUPATION             5
     317             MOVE CUST-ACCT-BALANCE TO RPT-CUST-ACCT-BALANCE          5
LOG 0----+----1----+----2----+----3----+----4----+----5----+---- LINE: 23 OF 26
0023  GO ;
0024  STEP ;
0025  STEP ;
0026  GO ;
PF  1:?         2:STEP      3:QUIT      4:LIST      5:FIND      6:AT/CLEAR
PF  7:UP        8:DOWN      9:GO       10:ZOOM     11:ZOOM LOG 12:RETRIEVE
```

I.

Figure 2 – Debug Tool and Code Coverage (Program Frequency Counting)

To attend a training scenario in the industry is necessary to implement an architecture that allows, in cases of Batch applications, access the tool Debug Tool via publishers that comes in tools TSO - Time Share Option and Roscoe. Figure 3 illustrates one solution that is typically drawn for Performing Batch.

To make a balance in his workload some organizations create two development environments and install them the same solution. Figure 4 shows how the architecture is a distribution in both development environments, and herein identified as DESA and DESB. The solution that is presented to execute online programs, also called transactional batch is similar to the solution, and the differential is the CICS environment [7]. Figure 5 shows an example of a solution.

To train students running tests in an environment similar to the industries, the tool Debug Tool can be performed via a workstation RDT - RDZ Unit Test [18], which has a unit testing environment. This station shown in Figure 6 is a development platform for System z running in a personal computer capable of execute operating system z/OS. This provides flexibility to run a customized environment because it facilitates developers to easily prototype releasing new applications using mainframe MIPS - Millions of Instructions per Second, which may be used to execute processes. This workstation reduces the dependencies because it is an individual unit. A university that can not afford to buy a Mainframe machine is able to buy this unit test, whose server can support about 60 or more clients.

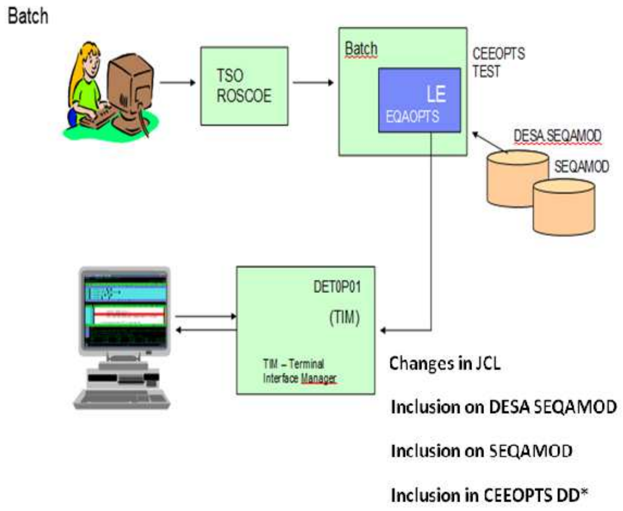


Figure 3 – Architecture of a Batch Solution Font: [6]

VI. Conclusion

This research presented a solution for debugging programs Mainframe and use your suggestion in the activity of preparing students in learning languages for Mainframe environment. Its use in an educational facility programming language becomes essential as it allows the student to know the procedures of preparation and execution of testing in this environment, step by step. This activity will prepare graduates to join the industry in development teams large applications. The innovative use of this technology in the classroom will meet the requirements presented by industry.

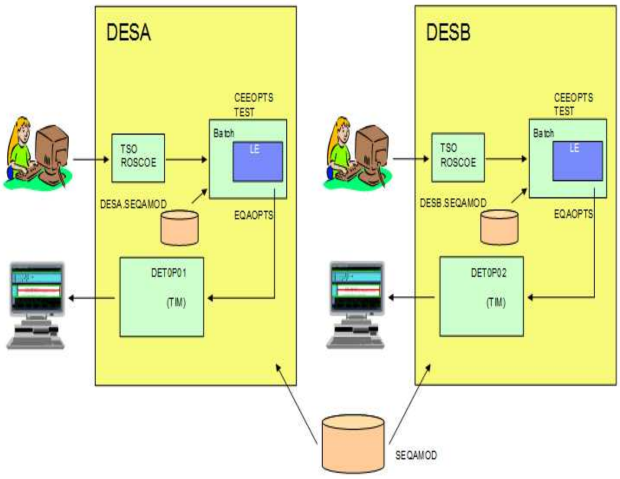


Figure 4 – Environment Distribution Font: [6]

Like all kinds of tests are complementary they are designed to make the software into the hands of the customer with the fewest mistakes and less time spent. The hard part is reaching the exact equation that is capable of delivering quality products and meeting customer expectations.

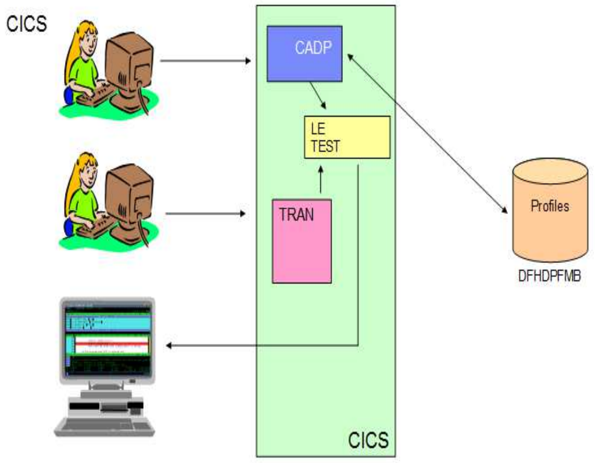


Figure 5 – Architecture of On-Line Solution (CICS) Font: [6]

This tool can be appointed as a new application in software testing, and may be part of the process of software development of the institution, allowing to accurately meet the requirements presented by the area of Software Quality, intending to deliver the final product with minimum possible error meeting all the requirements generated by the end customer.

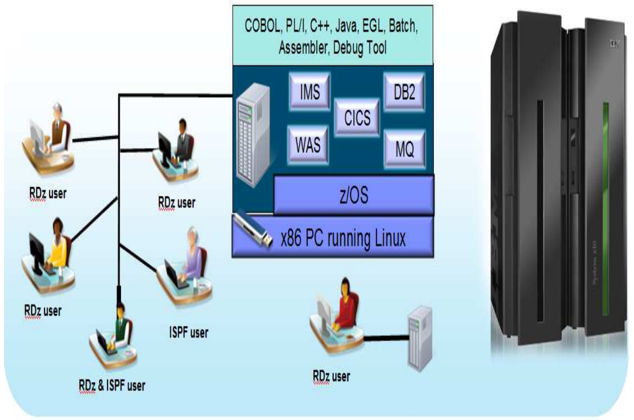


Figure 6 – Characteristics of RDT - RDZ Unit Test Font: [19]

It can be concluded from this research that the tool assists the university in teaching programming languages for mainframe environment because it reproduces the exact tests in the IT industry. When entering the labor market the newly graduated student will ease in adapting to a development team or tests, for work with the tool he used in his graduation with the same profile presented by industry because this tool can prepare them for IT Industry.

VII. Acknowledgments

Authors of this paper would like to acknowledge:

1 – FAPDF – Fundo de Apoio à Pesquisa do Distrito Federal – A Brazilian fund to support research located in Brasilia city, Distrito Federal state.

VIII. References

[1] ARCURI, A.; BRIAND, L. "A Practical Guide for Using Statistical Tests to Assess Randomized Algorithms in Software Engineering". ICSE '11 Proceedings of the 33rd International Conference on Software Engineering. ACM New York, NY, USA, 2011

[2] BANCO CENTRAL DO BRASIL. "Implementation of Basel II in Brazil" in portuguese. Brasilia, Brazil. Available in http://www.bc.gov.br/?BASILEIA2 Acess in 06/12/2012.

[3] BEST MANAGEMENT PRACTICE. "itSFM - An Introductory Overview of ITIL® V3 - Version 1.0". Berkshire, UK. Available in: http://www.best-management-practice.com/gempdf/itSMF_An_Introductory_Overview_of_ITIL_V3.pdf Acess in 12/06/2012.

[4] DEVMEDIA. "Introdution in Software Test" in portuguese. Available in: http://www.devmedia.com.br/artigo-engenharia-de-software-introducao-a-teste-de-software/8035. Acess in 03/26/2012.

[5] GOUVEIA, C.C. "Integration Test for Component Based Systems" in portuguese. Dissertation of Master's degree in Computer - Universidade Federal da Paraíba, Campina Grande, Paraíba, Brazil, 2004.

[6] HUGGLER, P. "Debug Tool in Banco do Brasil" in portuguese. Workshop for Banco do Brasil Employees. Brasília, Brazil, 03/08/2012.

[7] IBM. "CICS Transaction Server Glossary. CICS Transaction Server for z/OS V3.2". IBM Information Center, Boulder, Colorado, USA. September, 2010. http://publib.boulder.ibm.com/infocenter/cicsts/v3r2/index.jsp. Access in 06/02/2011.

[8] IBM. "Debug Tool for z/OS". IBM Corporation, Armonk, New York, USA. Available in: http://www-01.ibm.com/software/awdtools/debugtool/ Access in 05/05/2012.

[9] IBM. "IBM previews z/OS Version 1 Release 13 and z/OS Management Facility Version 1 Release 13". IBM Corporation, Armonk, New York, USA. February, 2011. Available in: http://www-03.ibm.com/systems/z/os/zos/. Access in 05/05/2012.

[10] IBM REDBOOK (2008), "IBM Application Development and Problem Determination Tools V7 for System z". IBM Corporation, Armonk, New York, USA. Available in: http://www.redbooks.ibm.com/redbooks/pdfs/sg247661.pdf

[11] IT WEB - "Delays in IT projects impacting corporate profitability" in portuguese. IT Web, São Paulo, Brazil. Available in http://itweb.com.br/28935/atrasos-em-projetos-de-ti-impactam-lucratividade-das-empresas/. Access in 06/12/2012.

[12] KALE, N.D. "The Key Principle of Testing – Testing is Context Dependent". Test Experience Magazine, March/2011. Available in: http://www.istqb.org/references/articles/istqb-related-articles.html. Access in 08/01/2012.

[13] MAIRHOFER, S.; FELDT R.; TORKAR, R. "Search-based software testing and test data generation for a dynamic programming language". GECCO '11 Proceedings of the 13th annual conference on Genetic and evolutionary computation. ACM New York, NY, USA, 2011.

[14] MALZ, C.; SOMMER K.; GÖHNER, P.; VOGEL-HEUSER, B. "Consideration of Human Factors for Prioritizing Test Cases for the Software System Test". EPCE'11 Proceedings of the 9th International Conference on Engineering Psychology and Cognitive Ergonomics. Springer-Verlag Berlin, Heidelberg, 2011.

[15] OFFICIAL ITIL® WEBSITE. "Information Technology Infraestructure Library". Buckinghamshire, UK. Available in: http://www.itil-officialsite.com/ Access in 06/12/2012.

[16] PRESSMAN, R. S. "Software Engineering", McGraw-Hill International Editions, Artmed, 2006.

[17] NATION CAREER SERVICE. "Strong demand for IT professionals – and it's predicted to rise ". Available in:

https://nationalcareersservice.direct.gov.uk/aboutus/newsarticles/Pages/News-DemandForITPro.aspx. Access in 03/09/2013.

[18] IBM. "Modern development tools for mainframe application development". Available in: http://www-01.ibm.com/software/rational/products/developer/systemz/. Access in 03/09/2013.

[19] IBM. "Rational System z Development and Testing Hub". Available in: https://www.ibm.com/developerworks/mydeveloperworks/groups/service/html/communityview?communityUuid=5d4610cf-76f1-46d9-806f-88f157367222. Access in 03/09/2013.

[20] O'Kane, K.C. "Basic IBM Mainframe Assembly Language Programming". Paperback, USA, 2011.

Game Programming for High School Mathematics Education

Josh Archer, Robert Johnson, and Nelson Rushton
Computer Science Department, Texas Tech University
(Josh.archer | robert.johnson | nelson.rushton) @ ttu.edu

Texas Tech University
Department of Computer Science
Box 43104
Lubbock, TX 79409-3104

Submitted to FECS '13

Abstract: A short, four-hour course was given to high school students in game programming. It was observed that students were self motivated in solving the problems, even when no grade was involved, and that they took home work even when not required to. We hypothesize that such a course, if offered for a full semester, would be popular with students and improve math achievment.

Key words: mathematics education, computer science education, computational thinking

This paper describes the methods and results of a short course in teaching game programming to high school students. The principal objective was to give students an engineering experience using the mathematics they were studying, or had studied, in other classes. By an *engineering experience* we mean that students would have to apply their scientific knowledge -- in that case, specifically, knowledge of mathematics -- to design an artifact that behaved in a given, clearly specified fashion.

Since the goal of the course is to create interest in, and deepen understanding of related mathematics curricula, we are creating a new game development environment with the goal of minimizing the overhead students incur in learning new learning technology. We hypothesize that by doing this the overhead can, surprisingly, be cut to almost zero, so that with well chosen exercises the game programming course practically becomes a course in applied mathematics.

Section 1 describes the exercises that were used. Section 2 describes the observed results, and Section 3 describes the general changes we will make as a consequence f these observed results when repeating the course or similar courses. Section 4 gives a functional programming language students will use in future versions of the course, known as LEDU. Section 5 describes a game programming environment based on LEDU. Section 6 describes conclusions and future work.

1. Exercises used previously

The course consisted of four class meetings of approximately 45 minutes each, and was given twice as a non-credit short course at New Deal High School in New Deal Texas, during the Spring semester of 2013. Students were given a game program that plays “five in a row”, or “gomoku” -- basically tic tac toe on a 20x20 board, where players take turns placing marks on the intersections of the grid (rather than in the cells), and the first player to get five in a row wins. The code was written in Python, and four of the functions were deliberately broken so that the game could not be played. The broken functions were as follows:

1 top(i): returns the top endpoint of the i'th horizontal line segment in the game grid.
2 bottom(i): returns the bottom endpoint of the i'th horizontal line segment in the game grid.
3 left(i): returns the left endpoint of the i'th horizontal line segment in the game grid.
4 right(i): returns the right endpoint of the i'th horizontal line segment in the game grid.
5 def point(i,j) -- returns the coordinates of the intersection of the i'th horizontal grid line with the j'th vertical grid line.
6 distance(x1,y1,x2,y2) -- the distance between the points.

The first four functions are needed to draw the game board. The fifth and sixth ones are needed to see if the player clicked within a set distance of a given intersection, because the player moves by clicking on intersections.

Interestingly, the final problem was not trivial for students *even though most of them knew the distance formula.* We conjecture there are two reasons for this. First, the students did not understand the distance formula well enough, or at a high enough level of abstraction, to use it in novel contexts. Second, their cue for applying that problem solving strategy had been taken from a limited set of examples, all in the context of *math class,* and it didn't occur to them that it could be used elsewhere. Both of these problems are worrisome barriers to students applying knowledge across contexts, and we believe that the introduction of game programming into the curriculum helps remedy both.

Importantly, we wrote a wrapper around the needed calls to Peter Zelle's graphics library that accomplished the following:

1 The y axis pointed up, as it did in students' math instruction -- as opposed to down as it naturally does in Zelle's library.
2 Points are written as ordered pairs, like students are used to (e.g., (23, 332))

In this way, we minimized the amount of Python students would have to learn to complete the program. It turned out better than expected: In our rough extimate, students spent about 85 of their time solving the mathematical problems needed to design the functions, and about 15% thinking about Python.

2. Observed results

The implementation of game programming in HS at New Deal is aimed at meeting very specific goals. The first goal is to allow the student to work in an environment that will model professional use. The second goal is to learn from an idea of living applications. The Third goal is for the student to learn from objective success criteria. What we hoped to see from the New Deal experiment was a manifestation of our ideas within student behavior.

Professional use is defined as allowing the student to work in an environment that mimics how they would work in the same environment out in industry. In industry an employee will be given a programming task, then they will endeavor to complete this task either on their own, or with a team. Very little supervision is provided in how to solve the tasks. If employers could readily solve the tasks themselves, then they would not need the employee. We establish this kind of environment by limiting directives to the student, and creating lessons in which a large amount of instruction is not needed. We also do not spend a large amount of time with any one student. When a student needs help. We limit the amount of help to focus on just the specific problem they have.

In application this was one of our biggest worries with the students at New Deal, how would the students react to solving problems with limited instruction. The results where pleasing. Students at first wanted to just dive into the work, and get it done as quickly as possible. Once thy discovered that this was not possible, they started to adapt to using more quality means of exploration. Some got out paper, and attempted to reason out ideas to solve the problems. Some reexamined the initial notes for help. Some formed groups, and bounced ideas off of each other. In all cases they had a pride in trying to figure out the solution without being told. Even when they wanted help, they did not want help beyond their specific problem. The difference from the classroom was in the quality of question. The class did not use questions of generality, “I don’t understand!” which would have been something I would here in a common math classroom. Questions became relevant and meaningful. For example, “So, if change the ith variable it will change where the segments are on the screen?”

The idea of living application is that the student is doing something that will have value to them. They are not trying to achieve the result because the teacher said it is so, they themselves see the problem as relevant. Since we are doing all of our lessons through the idea of creating games in a programming language, it can be supposed we are working at a high level of living application. The power of this was evident in a break out lesson where students used a mini program to write their initials. I do not know why a young person seeing there name written on a board or screen means so much to them (that might be an entire thesis in psychology), but it is highly motivational for young people across all educational settings. The students took great care to get this lesson right. Students created a competition between each other for quality of names produced. With Students of New Deal the concept of living application seemed to create a profound drive to get the problem right. As an experienced math teacher this becomes a time saver. I don’t have to coach as many students as I would in a normal class. This allowed for me to focus on students, who would have a hard time seeing the concepts.

Objective Success Criteria is a concept where the student identifies the truth in the assignment, and does not need reliance on validity from the teacher. Truth is discovered. In the students common mathematics classroom time is a factor. Teachers may not get to solidify a concept with examples of validity. The nature of the work with game programming is that the truth of the theory is concrete. For example, a student just does not study the properties of a rectangle; they discover the accuracy of the properties through creating a rectangle that is tangible. They see with their own eyes that formulas like distance work. From my experience as a math teacher this was a great deal of fun. Watching students create geometric objects from equations, and understand what they had done mathematically based on the output shown on the screen. Far different then the methods I had used in the classroom of keeping concepts isolated, and detached from use.

3. Proposed changes to exercises

The first lesson learned form the trial run of the course was that we need sets of graduated small lessons rather than a small number of large lessons. Essentially, a student's interest should inspire them through the end of the exercise -- which will not happen in some cases if the exercises take too long. Second. we learned that students are much more self motivated if we simply give them the problem and let them solve it, without too much guidance in the mathematics.

The tricky part is getting students to *spontaneously* apply mathematics to solve the problem -- as opposed to just "fiddling with it" until they get it right. The latter approach works with the easy warm-up problems, but is hopeless for the more difficult target problems. Our proposed solution for next time is to make extra effort to give warm up problems that cannot be done by guess-and-check or brute force.

4. LEDU

When using Python, students are unavoidably distracted by the infrastructure of the surrounding code, and sometimes have trouble with the interpreter (especially in case they change a piece of code that is outside the scope of the exercise). In order to keep the student's experience as pure as possible, we have designed an ultralight functional language specifically for the purpose of defining game functions.

LEDU data are symbols, rational numbers, truth values (`true` and `false`) and tuples. A *symbol* is in identifier preceded by a single quote. The *error* value is also a LEDU datum; but it cannot appear as a member of a tuple. Members of tuples may be symbols, rational numbers, truth values, lists, lists of lists, etc.The members of a list need not be of the same kind.

Tuples are written with their members separated by commas (if there are two or more), and enclosed in parentheses. Tuples can have 2 or more members.

Example tuples

- `(‘hello, ‘world, ‘x1)`
- `(4>2, ‘hello)`

The following are operators on their respective data

1 symbols: =

2 rational arithmetic: + - /1 -/2 `* / ^ floor ceiling intdiv mod < > <= >= =`

3 lists: `head tail cons = ++ size in _`

4 boolean: `and or ~`

In LED, any document can be used to write code. Anything not in *courier* font is considered to be documentation and is ignored by the compiler, anything written in *courier* is code. A few examples of LED definitions for a few common expressions follow:

If *n* is an integer and *i* is a positive integer, then *divisor*(*i*,*n*) means that *i* is a divisor of *n*.

```
   def divisor(i,n) iff n mod i
= 0
```

A prime number is a number greater than 1 whose only divisors are 1 and itself.

```
   def  prime(n)  iff  n>1  and
noDivisors(2,n-1,n)
```

If *i*, *j*, and *n* are positive integers, then *noDivisors*(*i,j,n*) means that *n* has no divisors in the interval {*i..j*}.

```
   def noDivisors(i,j,n) iff
     i>j or
     ~divisor(i,n)           and
noDivisors(i+1,j,n)
```

In the above examples, something *is* or *isn’t* what is being defined. The *iff* keyword is meant to denote a definition for a boolean function.

A point is a list of two numbers, thought of as the x and y coordinates of the point. If *p* is a point then *xval*(*p*) and *yval*(*p*) are its first and second coordinates, respectively.

```
   def xval(p) = p_1

   def yval(p) = p_2
```

If *p* and *q* are points with different *x* values, then *slope*(*p*,*q*) is the slope of the line that passes through *p* and *q.*

```
   let slope(p,q) =
       (yval(p)    -    yval(q))    /
(xval(q) - xval(p))
```

The following examples make use of conditional definitions, if “*owise*” is present, then it denotes the default. If it is not present then there is no default.

If *x* and *y* are numbers, *max*(*x,y*) is their maximum.

```
      let max(x,y) =
        x if x>=y,
        y owise
```

If *L* is a nonempty tuple of numbers, *tMax*(*L*) is the largest member of *L*.

```
      let tMax(L) =
        head(L) if size(L) = 1,
        max(head(L),tMax(tail(L)))
owise
```

5. LEDU Game Engine

The LEDU game engine will allow the creation of LEDU-based exercises that students can do without looking at the surrounding Python code. Once the exercise is created, the student can type only the target functions -- that is, the one s/he is designing -- into a text file. The student then places the file into a folder containing the game engine executable and the appriate Python code (all ideally placed by the teacher beforehand), and runs the game engine and selects the name of the exercise. If their LEDU code is correct then they get a playable game. In case of syntax errors the student receives error messages. We hypothesize that this will allow students to exercise skills they transfer directly from their math classe.s

6. Conclusion

The number of students that signed up for the course was larger than expected (due to the name "Game Programming"). Students quickly learned that game programming was not a game; but they also learned that it was enjoyable, and that mathematics had direct, enjoyable applications outside of the textbook. As future work we plan to offer a for-credit class meeting daily during Fall of 2013, and compare the enrolled student's math achievement to that of a control group having similar history but not enrolled in the class.

Our Experience Teaching After-School Programming to Parents and Their Children

Rudolf Pecinovský[1], Jiří Kofránek[2],
[1]University of Economics, Department of Information Technologies, Prague, Czech Republic
[2]Charles University in Prague, Laboratory of Biocybernetics, Prague, Czech Republic

Abstract - *Many years ago, Prague's sport clubs introduced courses of swimming, gymnastics, trekking as well as other physical training for parents and their children, in which parents and children are active participants who carry on the selected activities together. Compared to this thoroughly-explored problem, courses of programming for adults and children face a very different set of problems. We were inspired by the experience of after-school athletes and decided to examine how an interest group of after-school programming for parents and their children might operate. This paper summarizes a two-year experience with running such courses.*

Keywords: Education, Program Architecture, Object-Oriented Programming, Design Patterns, UML

1 Itroduction

The authors of this paper teach a wide range of students, from children attending interest groups, to regular lessons in middle schools and universities, and all the way up to running continuing education courses for professional programmers. They teach according to the time-tested methodology of *Architecture First* [14, 15] in all courses. But each kind of course has its unique needs and requires solving particular types of problems:

- Although children are immensely perceptive and acquire the presented items very quickly, they easily are drawn by their schoolmates from programming to playing computer games.
- University students usually divide quickly into two groups:
  - Those who have never programmed before, and need a slower start in which the basic programming constructions are presented first. However, as soon as they become acquainted with them, they may successfully proceed in the course, assuming their diligence and willingness to develop required programs.
  - Those who have already been taught programming consider the initial presentation of basic terms unnecessary, but don´t realize the subject matter differs significantly from what they learned in their school courses. Most often they bring a false idea from school that programming in an object language means object-oriented programming. They don´t realize that they learned to write only classically structured programs in prior courses, and that using classes does not mean object-oriented programming. These students usually have problems in the more advanced phases of the course, and only later realize that the subject matter presented at the beginning was not as self-evident as they thought. Thanks to their passivity they did not learn all that they should have learned, and do not know what they need at the moment.
- Professional programmers fight with the aforementioned problems of advanced students even more intensely. These programmers are sent to our courses by their employers, who discover that graduates know a number of frameworks and programming tools, but are not able to propose a good architecture for more extensive projects. Programmers with such training know to implement entered interfaces (designed by others), but they do not recognize where in their proposal they should define their own interface.

The reason these two groups of advanced programmers share common properties is well-characterized by a constructivist theory of teaching [7], which shows that new findings are grafted on to previously-adopted knowledge. If these new items are not in accord with the old ones, the students are not able to remember them properly. They subconsciously modify new information according to their existing experience, and therefore they place into their memory something slightly different than was presented to them. This was our chief reason for using the *Architecture First* methodology.

2 Lessons of programming for parents and their children

Analyzing the aforementioned problems, we had an idea to borrow the model of sport clubs. These are Czech after-school clubs that began offering courses of swimming, gymnastics, trekking and other training many years ago, and which are attended by both parents and children. Both are active participants, who carry out the given activity together. Therefore we opened the first experimental group of programming for parents and their children in the Technical Center of Children and Youth House, Prague. Our aim was:

- To clarify if this methodology can teach both children and parents. They have different needs: the children are pure beginners, while their parents are in most cases able to program a little bit, despite not being professional programmers.
- To analyze the difference in acceptance, when the same subject matter is presented to these two significantly different groups of students.
- To verify our presumption that in certain cases the children could help their parents interpret the first pieces of information gained in the initial phases of the course.
- To utilize the interest of parents to avoid the possibility that children could slide from programming to computer games, and to ensure that they would prepare for the following lessons with the necessary care.

3 The Architecture First methodology

The ***Architecture First*** methodology, which we use in our courses, comes out of the two following assumptions:

- From the fact that nowadays the major part of a programmer´s work is taken over by various code generators, and that the only area which has and will resist automation for a long time is the program´s architecture proposal.
- From the well-known pedagogical model of the Early Bird Pattern, which says [2, 3]: *"Organize the course so that the most important topics are taught first. Teach the most important material, the 'big ideas', first (and often). When this seems impossible, teach the most important material as early as possible."*

In other words: if you consider the art of proper architecture creation the basic knowledge with which students should leave their schools or programming courses, you have to teach it from the very beginning [16, 17]. Applying this methodology to programming, we present the material in our courses in four phases [15]:

1. In the first phase, the learner runs in an interactive mode in which all code is created by a generator included in the development environment.
2. In the second phase, the user is turned over to a text mode in which the students repeats subject matter of the first phase while learning to actually write the programs that were created by the generator in the first phase,.
3. Then, the students become acquainted with more demanding constructions in the third stage, which are behind the limits of the generator´s abilities. In this stage we still concentrate to architecture view of program and do not explain algorithmic construction but sequence of statements.
4. In the fourth stage, the students become acquainted with basic algorithmic constructions and learn to use them in their programs. In the final phase the students learn further important data structures and programming expressions.

The complete process is demonstrated in the graphic programs the students create. The students have at their disposal a very simple graphic library, and they program behavior for the devices illustrated in that library (they create elevators, cars going through complicated tracks, etc.). We verified that the students thus obtain very visual feedback when they see the constructions functioning.

3.1 Introduction to OOP in an interactive mode

In the first phase, we take advantage of the *BlueJ* development environment [1, 9]. We start from the very beginning with a non trivial project (see Figure 1) and we explain the basic object constructions as follows: objects, classes, interfaces, and interface inheriting. We explain their usage in a program proposal, and the students interact with these and many others in the interactive mode in which the students simulate one of the program´s objects. The students can immediately try everything what they propose.

Due to the fact that the student don´t deal with coding in the first stage, they are not distracted by syntactic rules of the particular language and can instead concentrate on architectonic principles. This enables us to demonstrate how such basic constructions—which the implementation of an interface is—operate immediately from the very beginning. We present a number of design patterns in the first phase of explanation, (such as *Utility class, Singletons, Null Objects, Enumerated Type, Servant,*

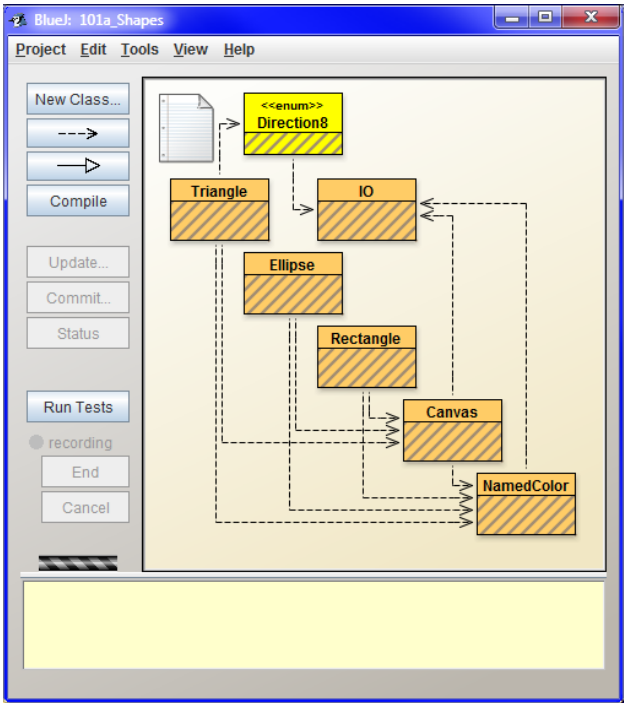


Figure 1: BlueJ with the starting project.

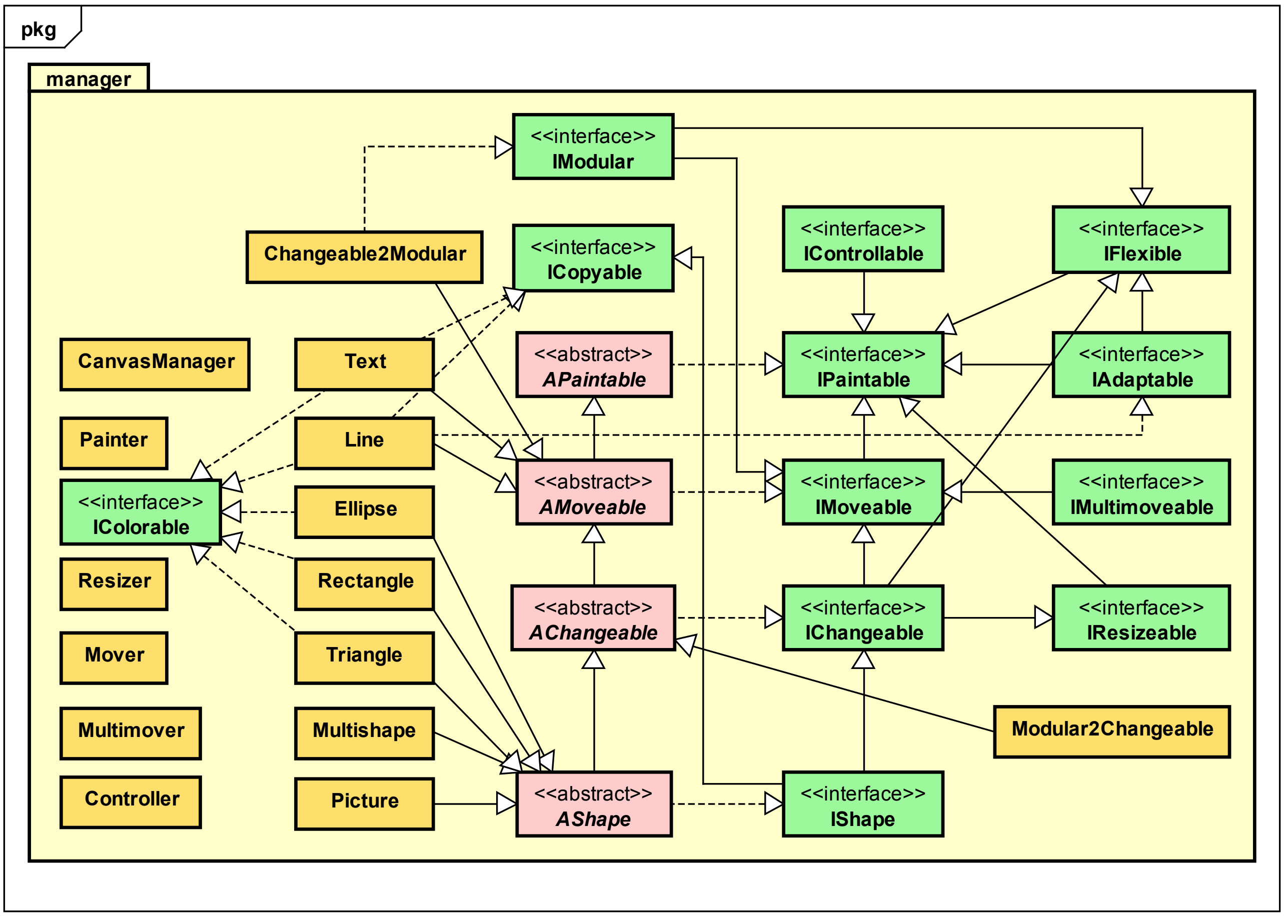


Figure 2: UML diagram of the project at the end of the first phase

Transfer Object - Crate, Mediators, Observers and so on) as well as certain architectural principles (programming against the interface, minimizing coupling, inversion of controls and similar concepts).

We frinish the explanation of the basic architectonic constructrions with a project roughly equivalent to the project at Figure 2.

3.2 Introduction to the syntax of the selected language

In the second phase of the couse, the students repeat the previous subject matter while learning to code the programs the development environment created in the first phase. Further, they learn how to use the syntactic rules and basic programming constructions of the selected language without being distracted with proposing a program. The program was already proposed in the first phase, and now they need only implement the rules of the first proposal.

The second phase finishes with an explanation of working with packages (name spaces), after which the students move to a professional development environment. For later phases, we used the *Java* language and the *NetBeans* development environment.

3.3 More complex programming constructions

In the third phase we leave the teaching environment and pass over to a professional environment which enables us to propose a more extensive project. Moreover, some of the students now become acquainted with some environments they will really use in their careers.

Further in this stage the students hear a deeper explanation of the architectural principles, interspersed with a presentation of a project using these programming constructions. And the introduction of design patterns continue: *State, Adapter* and *Decorator*.

In the third phase the students also encounter more complicated programming constructions, such as generic types, lambda-expressions, internal data types, collections and streams.

In this phase the students are programming more complex behavior: vehicles which are able to turn, or to drive along a zigzag track. They equip them with turn signals, and they also learn how to manage more cars at the same time. Yet I remind the reader that at this moment the students still don´t know algorithmic constructions like the conditional command and the cycle, because until now they have not needed them.

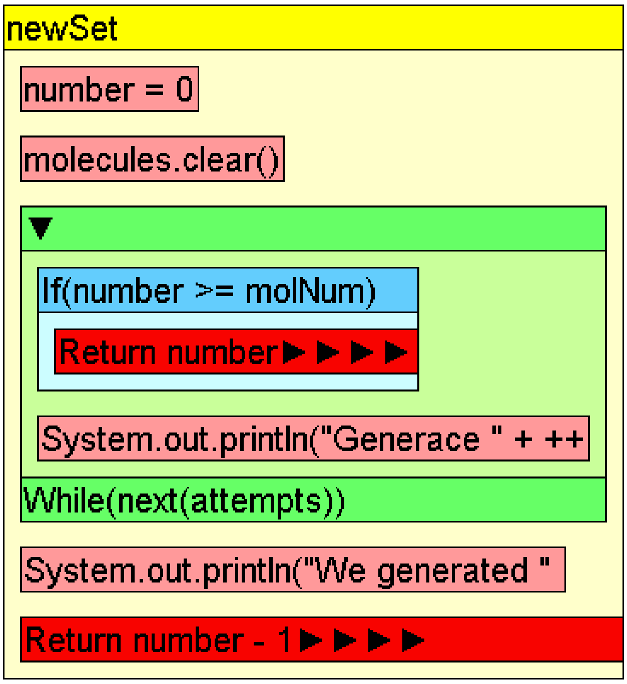
Until this moment, the students have made do with only interface, inheritance and their implementation. At the end of this phase, the students hear an explanation of inheritance, learning in which situations it can be used and, on the other hand, when they have to be careful with it.

3.4 Algorithmic constructions

The fourth phase introduces algorithmic constructions which the students might encounter in older programs: conditional commands and cycles. At this time they can have a look at how the problems they solved without algorithmic constructions can be programmed differently.

Until now, we have used class diagrams as a primary means for program proposals. We taught the students to think primarily at the problem level and descend into the code level only at the moment when it is really necessary.

When explaining the algorithmic constructions we leave the *Unified Modeling Language (UML)* diagrams behind, and demonstrate all constructions through *kopenograms* [8, 24] (see Figure 3). *Kopenograms* are a handy tool for clear graphical representation of the structure of algorithms, and they have been particularly useful in teaching programming classes. They are a convenient supplement to the *UML* diagrams used to represent algorithmic structures.



```
newSet
number = 0
molecules.clear()
▼
If(number >= molNum)
Return number ▶ ▶ ▶ ▶
System.out.println("Generace " + ++
While(next(attempts))
System.out.println("We generated "
Return number - 1 ▶ ▶ ▶ ▶
```

Figure 3: Kopenogram of a simple algorithm

3.5 Further data structures and program expressions

A number of useful data structures and programming expressions were not needed until this moment, but are necessary for future careers. These are presented in the final phase. These include: working with files, regular expressions, the basics of GUI applications, and some further useful items.

4 Results

So far, we have run two one-year terms. The courses were attended by children aged from 10 to 14 together with their parents. We didn´t offer a second year of teaching to the first class, because we wanted to reopen the first class and further examine our methodology. The couples continued studying according to the textbooks we provided, and from time to time they contacted us with a request for consultation.

During the one year course, we worked through the end of the third phase with the children and their parents. Eight pairs from the original twelve finished the course at the end of the first year, and ten couples out of twelve completed the course in the second year.

The course of teaching confirmed a lot of our assumptions, but also brought certain surprises.

4.1 Graphic interactive developing environment

The courses confirmed the utlity of the simple *BlueJ* interactive environment. This accorded with our previous experience, as well as experience of other authors [20], that it is useful to visualize relations among objects and classes through *UML* at the beginning, and only later pass over to a more complicated professional development environment. It is advantageous when the students get accustomed to using *UML* as a natural part of the problem analyses from the very beginning. We proved that using graphical tools for analysis and describing of the developed program is quite natural for children.

4.2 Speed of understanding

We verified that the same lessons can be used for both children and their parents. It is however important to avoid tedious lecturing, and instead stress the equivalencies between developed programs and a simulated reality. Students thus regard the objects in the program as natural equivalents of their patterns in the real world, not as an abstraction divorced from reality.

In several situations we experienced that the children really corrected their parents when they understood the subject matter incorrectly. The children explained the concepts in their own words, and the parents accepted their interpretation.

On the other hand, there were also children in the course (and surprisingly these were the older children) who always waited to see how their parents would understand the problem. They did not program anything independently and they only copied programs from their parents' computers

4.3 Unexpexted probles

There was also a surprise for us in these combined courses. When we taught courses attended only by children, we had no fundamental problems with homework. But these problems did occur in combined courses. The parents were not accustomed to doing homework, but instead learned how to justify the fact that they had not completed their homework. And of course, the children quickly adapted to this and the general speed of progress in the course considerably decreased.

5. Discussion

Special languages are often used when teaching the basics of programming for children—as are special development environments connected with a graphic user interface, in which you move or draw with a little turtle (the language *Logo* [11], [12], [19]) or you manage a robot's movements (the language *Karel* [13], [4]). And so on. These visual teaching tools are brilliant if you want to teach children algorithmization and the basics of structured and modular programming. These visualization tools are sometimes also used in introductory programming courses at universities [5]. But they are insufficient for teaching more complicated programming constructs and principles of object architecture. Usually the development environment of the particular programming language is used for further teaching.

At this point, the students must make a great effort to learn the syntax of the given programming language and to start working with the development environment, and only on the basis of this knowledge can they learn more complex concepts of object architecture using the design patterns for creating their programs. However, the biggest problem of novice programmers seems to be, not understanding architectural concepts, but rather learning to apply them [10].

One way to overcome the requirement of syntax knowledge is using visualizing tools which enable users to compose and test object programs visually, without coding them. These tools must be sufficiently simple and cannot discourage the beginners by their complexity. One such tool is the aforementioned *BlueJ* environment [23]. The development of *BlueJ* was started in 1999 by Michael Kölling and John Rosenberg at the Monash University, as a successor to the *BlueJ* system. The *BlueJ* system was an integrated system with its own programming language and environment. *BlueJ* implements the Blue environment design for the Java programming language. BlueJ is currently being maintained by a joint team at the University of Kent (Canterbury, England) and La Trobe University in Melbourne, Australia.

In March 2009, the BlueJ project became free and open source software, licensed under the GNU GPL with the class path exception.

Nearly one thousand schools and universities are now using the *BlueJ* environment [23]. A textbook on this environment was published in its fifth edition last year [1] and has been translated into six national languages.

Inspired by the success of *BlueJ*, Microsoft implemented a similar environment into the Visual Studio 2005 development environment under the name of *Object Test Bench* [25] in 2005. At the same time, Microsoft submitted a patent registration [6] concerning their *"facility for testing an object in an integrated development environment without providing source code or knowing semantics of a language"*.

The patent registration caused a stormy reaction among *BlueJ* authors as well as users, and in the end no patent was awarded. *Object Test Bench* was also a part of *Visual Studio 2008*; however, this tool has not appeared in later versions of *Visual Studio*. Microsoft explained the removal by saying that majority of professional programmers use more complicated visual tools that are included in *Visual Studio*, and *Object Test Bench* was determined more useful for academic circles and teaching of programming.

Nevertheless, we believe that it´s a pity *Object Test Bench* cannot be downloaded into the new *Visual Studio*, even as an optional supplement for teaching *C#*. According to our experience, it is advantageous when we have a visualizing tool with a relatively simple interface at the user's disposal for teaching programming. This is because we don´t have to bother students at the beginning with the quite complicated task of managing a development environment. Students in our courses start with working in the simple *BlueJ* environment, and only later pass over to the professional *NetBeans* development environment. Learning and managing a more complex development environment (as well as necessary knowledge of programming language syntax) is far easier when the student has proper habits from a simple object development tool.

Consistently exploiting the possibilities of the simple *BlueJ* environment enabled us to teach basic principles of architecture to students even before we started to explain the more detailed presentation of syntactic structures of the language. Especially with children (who are not burdened with any previous knowledge of programming language) we proved that object-oriented thinking when creating architecture is very natural. Children understood the basic concepts, and clarified their understanding through practical examples using the *BlueJ* interactive mode.

When composing the tasks, we often program a particular part of the task in advance (these parts use constructions not explained yet),. This enables us to solve even more complicated tasks, and the computer solution motivates our students. Thus we could differ from classic textbook tasks, where the subject matter is demonstrated in simple tasks whose main purpose is to illustrate certain explained characteristics of the language. The typical task of a programmer is usually not to propose the solution of a simple problem, but on the contrary, to solve a current, usually very complicated program, proposed by somebody else, by certain new function(s). Therefore, our attitude was much closer to real programmer´s practice. Besides that, in many examples we could better demonstrate not only the properties of separate language construct, but combining and using various language constructs. The importance of this aspect is stressed by Robins et al. [18], who recommended in their study that instructions should focus not only on learning the new language features, but also on combining and using these features. His study also suggested that programming strategies should receive more and more explicit attention in introductory programming courses.

Proceeding by short steps during the presentation proved to be very important, as did making an effort to see that the explanation and the programming tasks would followed naturally from the previous ones. As Winslow pointed out in *"Programming pedagogy"* [22], a good pedagogy requires the instructor to keep initial facts, models and rules simple, and only expand and refine them as the student gains more experience. Vihavainen [21] proposed a teaching methodology in which "small goals" are discussed as part of the teaching approach. "Small goals" are defined as the small parts that clearly set intermediate goals.

The result of gradually-expanding small goals was that the children succeeded in mastering quite a complicated course of object programming during one year, a course that can be compared in its content and range with some university courses.

6. Conclusions

The majority of programming courses begin with a great effort to explain the syntax of the selected programming language, as well to explain the meaning of separate language constructs. Only then do they proceed to an explanation of object architecture concepts and using design patterns. However, we are convinced that it is possible to turn this on its head, and start with an explanation of basic architectural concepts immediately, then teach the particularities of syntax throughout the whole course.

Our courses of programming for children with their parents confirmed our assumptions. They proved that when combined with suitable visualizing tools (such as BlueJ), it is possible and, and indeed better to teach object architecture from the very beginning of programming education.

7 References

[1] Barnes, D. J., Kölling, M. "*Objects first with Java: A practical introduction using Bluej*", 5th edition. Pearson Prentice Hall. ISBN 978-013-249266-9, 2012

[2] Bergin, J. "Fourteen Pedagogical Patterns"; *Proceedings of Fifth European Conference on Pattern Languages of Programs (EuroPLoP 2000)*, Irsee, Germany, 2000 [online] http://hillside.net/europlop/HillsideEurope/Papers/EuroPLoP2000/2000_Bergin_FourteenPedagogicalPatterns.pdf.

[3] Bergin, J. "*Pedagogical Patterns: Advice For Educators*". CreateSpace Independent Publishing Platform, ISBN 1 4791 7182 4, 2012

[4] Bergin, J., Stehlik, M., Roberts, J., Pattis, R. "*Karel J Robot: A gentle introduction to the art of object-oriented programming in Java*". Dream Songs Press, 2005.

[5] Dai, K., Zhao, Y., Chen, R. "Research and Practice on Constructing the Course of Programming Language"; In *Computer and Information Technology (CIT), 10th IEEE International Conference on Computer and Information Technology*, pp. 2033-2038, 2010.

[6] Goenka, G., Das, P. P., Unnikrishnan, U. "*U.S. Patent Application 11/255,066*", 2005

[7] Harasim, L. "*Learning Theory and Online Technologies*". Routledge, Taylor & Francis Group, 7625 Empire Drive, Florence, KY 41042, ISBN 978 0 415 99976 2, 2011

[8] Kofránek, J., Pecinovský, R., Novák, P. "Kopenograms – Graphical Language for Structured Algorithms"; *Proceedings of the International Conference on Foundation of Computer Science. WorldComp 2012*, Las Vegas. CSREA Press. ISBN 1 601 32211-9, pp. 90-96, 2012

[9] Kölling, M., Quig, B., Patterson, A., Rosenberg, J. (2003)."The BlueJ system and its pedagogy"; *Computer Science Education*, Vol. 13, No 4, pp. 249-268, 2003

[10] Lahtinen, E., Ala-Mutka, K., & Järvinen, H. M. "A study of the difficulties of novice programmers"; *ACM SIGCSE Bulletin*, Vol. 37, No. 3, pp. 14-18, 2005

[11] Papert, S. "Teaching Children Thinking"; *Programmed Learning and Educational Technology*, Vol. 9, No. 5, pp. 245-255, 1972

[12] Papert, S. "*Final report of the Brookline LOGO Project*". Massachusetts Institute of Technology, Artificial Intelligence Laboratory, 1979.

[13] Pattis, R. E. "*Karel the robot: a gentle introduction to the art of programming*". John Wiley & Sons, Inc., 1981

[14] Pecinovský, R. "Using the methodology Design Patterns First by prototype testing with a user"; Proceedings of IMEM, Spišská Kapitula, pp. 1-10, [online] http://edu.pecinovsky.cz/papers/2009_IMEM_Using_DPF_by_testing_with_a_user.pdf, 2009.

[15] Pecinovský, R. "Principles of the Methodology Architecture First"; *Objekty 2012 – Proceedings of the 17th international conference on object oriented technologies*, Prague, pp 1-5, [online] http://edu.pecinovsky.cz/papers/2012_OB_ArchitectureFirst.pdf, 2012.

[16] Pecinovský R. "*Java 7 – Textbook of object oriented architecture for beginners*"; (in Czech), Grada, Prague, 2012.

[17] Pecinovský R. "*Java 8 – Textbook of object oriented architecture for intermediates*"; (in Czech), Grada, Prague, 2013.

[18] Robins, A., Rountree, J., & Rountree, N. "*Learning and teaching programming: A review and discussion*"; Computer Science Education, Vol. 13, No.2, pp. 137-172, 2003.

[19] Singh, J. K. "*Cognitive Effects of Programming in Logo. A Review of Literature and Synthesis of Strate-*

gies for Research."; Journal of Research on Computing in Education, Vol. 25, No. 1, pp. 88-104, 1992.

[20] Van Haaster, K., Hagan, D. "Teaching and learning with BlueJ: An evaluation of a pedagogical tool"; In *Information Science+ Information Technology Education Joint Conference, Rockhampton, QLD, Australia*, pp. 455-470, 2004

[21] Vihavainen, A., Paksula, M., Luukkainen, M. "Extreme apprenticeship method in teaching programming for beginners"; In *Proceedings of the 42nd ACM technical symposium on Computer science education,* ACM, pp. 93-98, 2011.

[22] Winslow, L. E. "Programming pedagogy—a psychological overview". *ACM SIGCSE Bulletin, 1996,* Vol. 28, No 3, pp. 17-22, 1996.

[23] http://www.bluej.org

[24] http://www.kopenogram.org

[25] Microsoft MSDN Library. Object Test Bench [online] *http://msdn.microsoft.com/en-us/library/c3775d98(v=vs.80).aspx*

Acknowledgement

This paper describes the outcome of research that has been accomplished as part of research program funded by the Ministry of Industry and Trade of the Czech Republic. This research was funded by *grant number FR—TI3/869.* These results were obtained within a larger research project concerned with the methodology of teaching computer programming. This research has been supported by *ICZ Group (commercial partner)* and *University of Economics, Prague (academic partner).*

email to corresponding author: kofranek@gmail.com

Turtles, Robots, Sheep, Cats, Languages what is next to teach programming? A future developer's crisis?

F. Edgar Castillo-Barrera[1], **P. David Arjona-Villicana**[1],
Cesar A. Ramirez-Gamez[1], **F. Eloy Hernandez-Castro**[1], **and S. Masoud Sadjadi**[2]
[1]School of Engineering, Universidad Autónoma de San Luis Potosí, San Luis Potosí, México
[2]School of Computing and Information Sciences, Florida International University (FIU), Miami, USA

Abstract

This paper narrates the process that our Computer Science department has followed to drastically change the introductory programming courses. Traditionally, our students were learning to program using the C language, but because of the high failure rate it was decided to first teach basic programming skills using two programming support tools: Raptor and Scratch. Although the passing rate has increased, it has not been possible to determine if this new teaching strategy has eased our students to learn programming in **C** *or* **C++**. *We have also noticed that Raptor does not offer all the basic features required in the syllabus of our department, therefore an analysis of other programming support tools which may offer a richer set of characteristics is provided.*

1 Introduction

Programming courses and curricula have been evolving since they started and there seems to be no end to this process. The **BASIC** language was very popular as a teaching resource, but it lacked support for structured programming and a modular structure. Hence, Pascal became an answer to Basic's limitations. Nowadays, the most popular languages are **C**, **C++** and **Java**[7]. Our department is no different and until the fall of 2011, it used to introduce C as the first programming language. This produced a very high failure rate for the Introduction to Programming course.

Other programming tools offer the concept of a microworld, where a character performs actions based on instructions and within the environment or world defined by its creators. The first tools that used this concept were **Logo** turtle graphics and **Karel** the robot. However over time, it's very simple animated GUI ceased to be interesting to many students because technology provided better graphics and interfaces.

Still the main problem persists: most students do not develop a proper programming logic and cannot correctly employ control structures for programming [9]. New programming support tools with multimedia, like **Scratch** [20], have recently emerged. Unfortunately, Scratch introduces the students to programming based on the events paradigm, while the teaching objectives we seek to develop at our first programming course follow the linear and procedural paradigm.

We think that it is necessary to document the process we have followed to increase the passing rate of our students. This has required that we analyze different programming support tools and employ them in an introductory course. On the other hand, this process has not concluded and we will continue learning from these experiences in order to increase the chances of success for our next curriculum changes.

This paper is structured as follows: Section 2 shows the related work. Section 3 provides a description and analysis of the tools used at our department. Section 4 describes the experience using programming languages. Finally, Section 5 provides a discussion and conclusions of this work.

2 Related work

The literature review on how to teach programming skills to university students concurs that learning to program is difficult [12, 17]. Therefore, there have been many proposals for strategies and tools which could facilitate the learning process of this skill. One of such strategies has been to implement a pre CS1 course (CS0) in which students are familiarized with algorithmic and problem solving skills which could later be applied when trying to learn a standard programming language [8, 14], like **C**, **C++** or **Java**.

Computer science courses which try to develop programming skills without using a standard language usually employ a programming support tool. Most of these tools provide an easy mechanism to implement a program or algo-

Table 1. Analysis of tools used for learning and teaching programming

Tool	Subroutines	Parameters	Animated	Flowchart	Multimedia	Debugger	Statements for Graphics
KAREL	Yes	No	No	No	No	No	No
DFD	Yes	Yes	No	Yes	No	No	No
Raptor	Yes	Yes	No	Yes	No	No	Yes
Logo	Yes	No	No	No	No	No	Yes
Scratch	Yes	No	Yes	No	Yes	No	No
Byob	Yes	Yes	Yes	Yes	Yes	Yes	No
Jeroo	Yes	Yes	Yes	No	Yes	No	No
Greenfoot	Yes	Yes	Yes	No	Yes	Yes	No
PSInt	Yes	Yes	No	Yes	No	Yes	No
Alice	Yes	Yes	Yes	No	Yes	Yes	Yes

Table 2. Analysis of tools used for learning and teaching programming

Tool	Linear Arrays	Bidimensional Arrays	Variables	if	while	do-while	for	repeat until
KAREL	No	No	No	Yes	No	No	No	Yes
DFD	No	No	Yes	Yes	Yes	No	Yes	Yes
Raptor	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
Logo	Yes	No	Yes	Yes	Yes	No	No	Yes
Scratch	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes
Byob	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes
Jeroo	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes
Greenfoot	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
PSInt	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes
Alice	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No

rithm and a visual feedback that allows students to see the effects to the changes they make. In a SIGCSE panel session [13] the following tool classification was proposed:

- Narrative tools: help to create a customized story which in turn increases the student motivation and engagement, e.g. **Alice** and **Jeroo**.
- Visual programming tools: allow students to drag self-contained units of code and create their own programs without having to worry about the program syntax and coding errors, e.g. **Alice**, **Greenfoot** and **Karel**.
- Flow-model tools: implement a flow-model which shows student how a program should work and bypasses the need to write code e.g. **Raptor** and Iconic Programmer.
- Specialized realization: like **Lego Mindstorms**, allow students to physically experience the results of their implementations.

It is important to note that none of these tools will actually teach how to create a program in a standard language, like C or Java. Therefore, there is a gap between tools and languages. On the other hand, these tools encourage the development of the skills needed to learn a standard programming language. This means that, at least in theory, students that have successfully completed their CS0 course should be half-way to become novice programmers. Therefore, in order to evaluate the success of these tools it is necessary to wait until students have finished their equivalent CS1 course to evaluate if they have really learned to program.

Although there have been efforts to develop a standardized test to evaluate the programming proficiency of students [9], we are not aware if this test has been widely accepted by the academic community.

3 Tools

The use of tools with visual characters and animated objects using a language based on blocks are now more popular in the first course of programming. In fact, languages such as **BASIC** and **PASCAL** are no longer common in the first programming courses.

3.1 Using a simple character: Logo and Karel

Logo was created in 1967 at Bolt, Beranek and Newman (BBN). A Cambridge, Massachusetts research firm created by Wally Feurzeig and Seymour Papert [18]. Although successes have been reported in the use of microworld programming tools, the work done by Xinogalos [21] suggests the opposite. After Logo, the most used tool to teach programming in universities for first-year students was **Karel** the Robot, developed by Richard E.Pattis [11] [16] [19][1]. Another version of Karel, which focuses on Object Oriented Programming, was made by Xinogalos [22] and is called **objectKarel**. Based on the experience of other universities and the literature reviewed [17], we have decided not to start teaching to program using the object-oriented paradigm.

3.2 Flowcharts: Raptor and DFD

DFD was developed by the Smart group at University of Magdalena, Colombia [3]. This tool allows you to create flow charts with subroutine calls and execute them step by step. An even better tool also based on flow charts is **Raptor** [4], which incorporates the use of arrays, graphical mode, and a debugger. It also allows to create class diagrams in UML.

3.3 PSeInt: Pseudocode, Flowchart and more

PSeInt is a tool to learn the logic of computer programming based on pseudocode. Through the use of a simple and limited pseudocode-language and an intuitive user interface in Spanish. The students can begin to understand basic concepts of control structures, supported by a modular and a procedural programming paradigm. In addition, this tools allows the use of flowcharts, functions and procedures with parameters, and multidimensional arrays are also supported.

Although in our evaluation of tools PSeInt ranked as one of the best, it still lacks some desirable features (see Tables 1 and 2): It is only available in Spanish and does not support multimedia or animated features, which are attractive to students.

3.4 Multimedia: Scratch, Byop, Alice and Greenfoot

Visual programming tools, such as **Scratch**, **Byop**, **Alice**, **Greenfoot**, have been developed to attract children and young people to learn programming. They usually allow the user to develop their own games and provide friendly interfaces.

3.4.1 Scratch

Scratch [15] is defined as a programming language by its creators. It was developed at MIT by the Lifelong Kindergarten Group. Scratch allows building games, simple apps and animations, using blocks classified by their functions with colors, therefore the programmer does not have to learn commands, keywords or a specific syntax. Therefore this is a tool which tries to develop skills to program in an Object Oriented Language. Scratch is intended for 8 to 16 years-olds but it is being used in introductory computer science classes at some universities [15]. We selected Scratch as one of the tools for teaching programming because its graphical interface motivates our students to practice and develop their own applications.

The topics cover in our first programming course are selection statement, loops, functions and one-dimensional and two-dimensional arrays. We have been working with this tool for four terms, and in the paragraphs below we describe the advantages and disadvantages we have found.

Scratch has two different selection statements: the *if* statement and the *if - else* statement. Their logic is the same as C language, and nested selection statements are allowed. However, there is not a block to represent the switch command or multiple selections.

Scratch implements four blocks that represent loops, two of them are infinite loops, there is a while loop with a conditional at the beginning, and there is a structure similar to a *for* loop: it occurs a specific number of times but it does not have a variable to control the iterations. Finally, there is no block to represent the *do-while* loop.

One of the biggest problems using Scratch is the absence of functions, procedures or subroutines. Scratch works based on events and objects, therefore Scratch is not a sequential programming language such as C. Scratch allows sending messages from one to several objects.

Scratch has lists insted of arrays, which are different. Lists can have elements of different types and there are blocks that represent functions like orderly insertion, deletion of elements in a specific position, addition of elements in a specific position and replacement of elements. Those functions are not predefined for arrays in languages such as C. Two-dimensional arrays do not exist, neither the option

[1] A new version of Karel, called **Critters** [2], has been developed by Anderson and McLoughlin.

to define list of lists. There are other points that cause confusion at the moment of learning how to program in Scratch.

- The program can have many entry points, which means a program can start with an object, or two at the same time, or maybe the same object with two different actions.
- Scratch support threads that are a great programming tool, but having access of these when you start programming makes the learning process more complex.
- There are few functions and operators, e.g. the $>=$ and $<=$ operators do not exist.
- There are not rules for naming at the moment to create a variable or a message.

On the other hand, **Scratch** is pleasing for students and teachers because:

- Creating animations is fast and easy.
- Allows to import images and sounds.
- It allows to develop creativity.
- There is a web site where students, teachers and developers can find material, share ideas and projects.
- There are also some complements for Scratch, such as the PicoBoard which is a device with sensors that allow the computer to interact with variables in the external environment.

3.4.2 Byob

Byob (Build Your Own Blocks) is an extension of Scratch. We have not used it in class yet, but it provides the opportunity to solve three important issues found in Scratch.

- Provides the ability to create blocks by the programmer, which means the student can implement functions. In fact, these blocks can receive parameters. The problem with these blocks is that the external variables are recognized even though they are not declared inside the block.
- Multidimensional lists are supported, but programming them using blocks is not an easy task.
- Byob has two extra blocks for managing ASCII code, which allows the use of strings.

Some disadvantages of Scratch can be overcome by Byob, but first it needs to correct some details and increase the existing documentation.

3.4.3 Alice

Alice seeks to teach programming by employing a 3D graphical interface. The objective for students is to program and animate 3-D objects (e.g. people, animals, and vehicles) in a virtual world. Alice is a programming environment designed to teach computer programming to children. It uses a similar approach to Scratch by dragging and dropping instructions without requiring a text editor, but it differs in that you can see the generated code in Java. In addition to the basic language components such as variables, arrays, loops and conditionals, it also allows the use of functions and parameter passing. Although it is based on **Java**, it is not required to know the concepts of object-oriented paradigm for its use.

Because Alice is based on Java, and since the objective of our department is to teach our students to use the **C** language, we have decided not to use Alice in our courses. Because there might be a transition problem from Java to C.

3.4.4 Greenfoot

In the concept of microworlds, **Greenfoot** is an environment for teaching object oriented programming skills. It employs the **Java** language and it is a simplified programming environment for developing Java applications. It has a set of classes that enable the development of programs directly without starting from scratch. In addition, all the Java language features such as variables, loops, conditionals, classes and objects are available. Although **Greenfoot** is designed to be a programming environment for beginners, we use this tool to teach Object Oriented Programming in our later programming courses. In our experience **Greenfoot** has proven to be successful for this pourpose. In fact, it is designed to be used as a programming environment for beginners. There is a tendency to use Scratch or Alice afterwards.

4 Languages: PASCAL, C, C++ and Java

C, **C++** and **Java** are still the more popular languages in the software industry [7], which is why most universities employ these languages for their introduction to programming courses. Our experience have shown us that the use of C as a first language to learn how to program is difficult for our students.

4.1 PASCAL

As a result of the deficiencies of **BASIC** (the most popular language in the mid-1970s), **PASCAL** was considered by many to be a good option as a first language to learn programming. **PASCAL** is an elegant and complete language under the structured paradigm. However concerns were identified when developing the skills of programming logic. This led to the conclusion that it was necessary to develop this skill before learning the language. Dr. Richard Patys developed Karel robot language, associated with the concept of microworlds [21]. But based on our analysis of the concepts needed to learn the student **KAREL** found that did not meet all the requirements, so we proceeded to look for other programs. Moreover its graphical interface is no longer attractive to students with the development of graphics and multimedia.

4.2 C

The first language used in our Department, for the bachelor of Informatic and Computer Engineering, was PASCAL. After that the academic staff decided to use the **C** language for their first programming course. It is considered the most complete language for programming in a structured way and with other characteristics that allow its use in other subsequent courses (operating systems, compilers, graphs, numerical analysis, etc.)

In previous years, we have tried to improve our *Introduction to Programming with C* course by employing structures that would allow to define graphical and three-dimensional objects. We believed that this structures would develop the logical skills needed for programming, and at the same time provide an interesting environment for our students. Moreover, we would encourage students to develop their own video games. Unfortunately, the percentage of failure among these students was still very high. This led us to create a previous course where the only focus is on learning the concepts of structured and modular programming, looking for programming activities that are simple and attractive to students. We have found that it is very difficult to find a tool that includes all the loops in the C language: *while*, *do-while* and *for*.

A tool called C-Sheep, which is based on C and Karel and was developed by Anderson and McLoughlin [1], is another option that we are going to analyse for its possible use.

4.3 C++ and JAVA

For years it has been discussed whether to teach programming using an object-oriented language like **C++** or **Java**. A more complete discussion and review was made by Robins et al. [17]. Based on the experience reported from several universities and considering that the foundation of the object-oriented paradigm is structured programming, and also to the fact that this paradigm seeks to facilitate the programming of complex problems which is not the focus of instruction for a first programming course, we have concluded that it should not be the language for teaching an introductory course to programming.

5 Discussion and Conclusions

Dijkstra [5] suggested the use of mathematics as a way to teach programming, but our experience has shown low interest in young people to learn math, so it is not considerated a good choice.

Based on the experience gained and the consulted literature, we have not found definite proofs that using a visual tool, such as Scratch, is better than using a scripting language, a language of diagrams such as Raptor, or a programming language such as C. However, based on our analysis, we suggest that a tool should be created with the characteristics mentioned in Table 2, or to complement the PSEint tool. However in the case of PSEint, it might be better to be translated into English, to add further control structures required in our syllabus, and also it might be necessary to add a lively visual atmosphere such as Scratch that could motivate students. With these changes we believe that the number of failed students might decrease. Nevertheless, we need to evaluate in a later course if students are really learning how to program.

Students without experience in programming computers require tools that motivate them to learn and to acquire programming skills. In this era of technology, programming applications for cell phones, social networks, chats, web pages, tablets and video games are sometimes the most important concerns of young people.

This article has made the analysis of the results obtained using software tools to teach a structured programming as first course in Introduction to Computer Programming. In past years the course taught using the **C** programming language had a passing rate of around 30%. Once we employed tools based on **microworlds** and **flowcharts** the statistics showed an increase in the passing rate of around 20%. However we believe it is necessary to build a tool with the features mentioned above in order to furhter increase the passing rate, and that could lead to a proper learning of computer programming. We want to emphasize that passing a course of introduction to programming using a tool does not guarantee that the student knows how to program and it is not enough to acquire the required programming skills. In our second programming course based on C language, only 18% passed the course in December 2012.

Students failing in introductory programming courses are also more likely to drop their degree altogether. This, together with the fact that there is a decrease in applications to join in a bachelor of Computer Engineering [6][10], makes the problem of teaching programming even more important. With advances in modern technology, students demand programming tools for learning, in which they can apply to their interests, e.g. social networks, cell phones applications, web pages and video games [2]. Thus these applications could be used in introductory programming courses as a way to motivate and introduce students to the core computer science courses.

6 Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. OISE-0730065.

References

[1] E. F. Anderson and L. McLoughlin. C-sheep: Controlling entities in a 3d virtual world as a tool for computer science education. 2006.

[2] E. F. Anderson and L. McLoughlin. Do robots dream of virtual sheep: Rediscovering the" karel the robot" paradigm for the" plug&play generation". 2006.

[3] C. N. Cardenas, F. and D. Eduardo. Universidad del magdalena, santa marta, colombia. Grupo Smart, 1998.

[4] M. C. Carlisle, T. A. Wilson, J. W. Humphries, and S. M. Hadfield. Raptor: a visual programming environment for teaching algorithmic problem solving. In *ACM SIGCSE Bulletin*, volume 37, pages 176–180. ACM, 2005.

[5] E. W. Dijkstra et al. On the cruelty of really teaching computing science. *Communications of the ACM*, 32(12):1398–1404, 1989.

[6] S. Guia, S. Campus, and S. Talento. Fomentar el ingreso y la permanencia en carreras de ti.

[7] R. S. King. The top 10 programming languages. Technical report, IEEE Spectrum.

[8] M. Klassen. Visual approach for teaching programming concepts. In *9th International Conference on Engineering Education*. INEER, Jul 2006.

[9] M. McCracken, V. Almstrum, D. Diaz, M. Guzdial, D. Hagan, Y. B.-D. Kolikant, C. Laxer, L. Thomas, I. Utting, and T. Wilusz. A multi-national, multi-institutional study of assessment of programming skills of first-year cs students. *SIGCSE Bulletin*, 33(4):125–180, Dec 2001.

[10] B. Parhami. Puzzling problems in computer engineering. *Computer*, 42(3):26–29, 2009.

[11] R. E. Pattis. *Karel the Robot: A Gentle Introduction to the Art of Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1981.

[12] A. Pears, S. Seidman, L. Malmi, L. Mannila, E. Adams, J. Bennedsen, M. Devlin, and J. Paterson. A survey of literature on the teaching of introductory programming. *SIGCSE Bulletin*, 39(4):204–223, Dec 2007.

[13] K. Powers, P. Gross, S. Cooper, M. McNally, K. J. Goldman, V. Proulx, and M. Carlisle. Tools for teaching introductory programming: What works? In *Proceedings of the 37th SIGCSE technical symposium on computer science education*, pages 560–561, Mar 2006. This was a panel session, not really an article.

[14] D. Reed. Rethinking cs0 with javascript. *SIGCSE Bulletin*, 33(1):100–104, Feb 2001.

[15] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, B. Silverman, et al. Scratch: programming for all. *Communications of the ACM*, 52(11):60–67, 2009.

[16] E. Roberts. Karel the robot learns java. *Department of Computer Science Stanford University*, 2005.

[17] A. Robins, J. Rountree, and N. Rountree. Learning and teaching programming: A review and discussion. *Computer Science Education*, 13(2):137–172, 2003.

[18] C. J. Solomon and S. Papert. A case study of a young child doing turtle graphics in logo. In *Proceedings of the June 7-10, 1976, national computer conference and exposition*, pages 1049–1056. ACM, 1976.

[19] R. H. Untch. Teaching programming using the karel the robot paradigm realized with a conventional language. *On-line at: http://www. mtsu. edu/untch/karel/karel90. pdf*, 1990.

[20] I. Utting, S. Cooper, M. Kölling, J. Maloney, and M. Resnick. Alice, greenfoot, and scratch–a discussion. *ACM Transactions on Computing Education (TOCE)*, 10(4):17, 2010.

[21] S. Xinogalos. An evaluation of knowledge transfer from microworld programming to conventional programming. *Journal of Educational Computing Research*, 47(3):251–277, 2012.

[22] S. Xinogalos, M. Satratzemi, and V. Dagdilelis. An objects-first approach to teaching object orientation based on objectkarel. In *Proceedings of the 5th WSEAS International Conference on Education and Educational Technology (EDU'06)*, page 93, 2007.

SESSION

DEGREE PROGRAMS, CURRICULUM, AND COURSE DEVELOPMENT + RELATED ISSUES

Chair(s)

TBA

Requiring a Systems Analysis and Design Course for a Computer Science Major

David V Beard[1], Kevin R Parker[1,2], Thomas A Ottaway[1,2], William F Davey[3], and Corey D Schou[1,2]
[1]Computer Science, Idaho State University, Pocatello Idaho, USA
[2]Computer Information Systems, Idaho State University, Pocatello Idaho, USA
[3]School of Business information Technology and Logistics, RMIT University, Melbourne, Australia

Abstract - *A systems analysis and design course can be used successfully to replace the traditional introduction to software engineering course in a computer science curriculum. The systems analysis and design course can provide an improved design experience as students and faculty stay focused on learning analysis and design, rather than programming. Such a course also can enhance computer science students' knowledge of project management, cost analysis, and business process flow analysis. Missing essential computer science knowledge outcomes must, however, be included in other required CS courses. Such a substitution all but requires that the instructor has significant software development and computer science backgrounds.*

Keywords: computer science curriculum, software engineering, systems analysis and design.

1 Introduction

For the last several years, Idaho State University's computer science (CS) department has required their students to take a systems analysis and design course from the computer information systems department (CIS) in place of the traditional introduction to software engineering course [e.g., 1]. While some of the course objectives typically covered in a software engineering course have had to be covered in other required computer science courses, use of the systems analysis and design course has allowed our computer science department to provide a general purpose major that meets ACM and other curricula standards while providing CS students with a broader analysis and design experience.

There are a number of circumstances that have allowed Idaho State University's (ISU) computer science department to successfully replace a software engineering (SE) course with systems analysis and design (SA&D) that might not be present at other universities. First, ISU's CS curriculum is a general purpose major focused on the development of large complex computer systems; many of their majors take jobs building complex business applications, so much of the material covered in SA&D courses, that is not part of the ACM 2013 Tier 1 or tier2 [2] required objectives, is still highly useful to CS students. In fact, a number of ISU's CS students choose to complete an MBA after finishing an undergraduate major in computer science and working for several years in industry. Second, ISU's CIS major is fairly technically oriented with a strong emphasis on IT technical skills, security considerations, systems analysis and design, data base design including entity-relationship diagrams [3] and some development of business applications. Third, ISU's CIS faculty has extensive software applications development and/or computer science backgrounds.

Perhaps most importantly, ISU's CS and CIS departments work well together, have considerable mutual respect, and thus have been able to cooperate in the details of curricula and course outcomes. Further, each has developed clear written definitions of what CS and CIS are for Idaho State University, and they have made these distinctions clear to faculty, students, administration, and prospective employers. Defining and maintaining these distinctions is academically and politically essential.

The purpose of this paper is to discuss how a systems analysis and design course can be used successfully to replace the traditional introduction to software engineering course in a computer science curriculum. We first detail what CS and CIS majors are at Idaho State University, and discuss some of the similarities and differences between an introduction to software engineering course, and a systems analysis and design course. We also describe what knowledge outcomes need to be incorporated elsewhere in the CS curriculum to ensure complete coverage of software engineering. Finally, we summarize our initial experiences and feedback in including SA&D in the CS curriculum.

2 Background

2.1 Computer Information Systems

Computer information systems (and its variants management information systems and information systems) evolved over the years from the referent disciplines of behavioral science, management theory, and computer science. Although the roots of information systems (IS) are firmly anchored in computer science, the instruction of IS courses must transcend the technological aspect, and encompass the organizational and behavioral issues that are integral in the business world. Every course must not only instruct the student in the application of the pertinent software package, but must also emphasize the importance of such tools in a corporate environment. Information systems can be traced back to the systems analysts who worked in the typical data processing departments of the late 1950s. Systems analysts elucidated the needs of a business situation, and developed algorithmic solutions – typically represented by flow charts – as well as managed the software development process. Systems analysts were aided by "programmers" who translated those flow charts into languages such as COBOL, keypunched the code, and resolved any syntax issues. CIS curricula included typically two courses in COBOL focused on file management and reports, as well as on the skills needed to analyze business situations and manage the process of developing solutions.

However, this initial CIS model quickly blurred. By the mid-1970s, computing departments were hiring "programmer analysts" who did both systems analysis and programming. Relational databases [4] became widespread by the mid-1980s [5] and by the late-1990s, web-based database applications had replaced most COBOL business systems requiring a far higher level of technical expertise including web applications, database management systems [6], networks, computer operations, and computer security. Two programming courses were no longer sufficient to do much development. Some CIS departments increased their technical course requirements while others dropped programming all together.

ISU's computer information systems major has somewhat more of a development emphasis than many IS programs, with two required programming/development classes as well as some programming required in a database class. The CIS major is part of a business school, so CIS students also take accounting, marketing, statistics, project management, and business law.

2.2 Computer Science

Computer science is primarily concerned with the development of complex software and to some extent hardware. Computer science evolved out of the systems programmers' role in the data processing departments of the late 1950s and in operating systems development by computer manufacturers. Computer science systems programmers wrote and optimized asynchronous operating systems, tuned disk drivers, developed data structures, wrote custom algorithms, etc., often in assembler and FORTRAN. Computer scientists also had the math skills to handle more scientific-oriented situations such as operations research algorithms and early simulation modeling with a focus on efficiency, effectiveness, and fault freeness.

Computer science curricula educate students in abstract reasoning about software programs and, above all, develop their ability to implement large software systems. ACM 1978 recommended curriculum [7] was based on this understanding and included CS-I, CS-II, computer organization, data structures, operating systems, and compilers. CS majors typically completed math courses at least as far as differential and integral calculus, discrete math, and linear algebra. The ACM 2008 update [8] and ACM 2013 [2] curricula guidelines now urge the inclusion of information assurance and cyber security knowledge outcomes.

Idaho State University's computer science (CS) department has embraced this traditional systems programmer notion of CS by developing a general purpose curriculum focused on implementing large, complex, high-performance, secure, asynchronous systems that require complex algorithms and intricate data structures including network, operating system, compiler, graphics and simulation packages. The curriculum incorporates 30 credit hours of math and science including differential and integral calculus, linear algebra, discrete math, and statistics. However, while mathematical formalism is essential, it is a means to an end, not an end in itself. The major is well above the ACM/IEEE 2013 curriculum minimum (Tier 1 + 90% Tier 2) [1] while being less than 1/2 of the total credit hours for the undergraduate bachelor of science degree. This allows students to add a second major in math or a business minor and eventually, an MBA in a 5th year.

The computer science department's focus on large systems development is due – to some extent – to its faculty members who have extensive industrial experience that includes writing hundreds of thousands of lines of code while developing tools for automotive engineers, radiologists, pilots, etc.

2.3 Fundamental distinction between CIS and CS

While computer science is concerned with the optimal design and implementation of software, computer information systems concentrates on the most efficient use of the finished package. For example, a database course in a computer science curriculum includes the internal organization of the database management system examining such theoretical issues as the most efficient algorithms for locking mechanisms and deadlock detection. A course by the same name in CIS generally examines the importance of organizing the data so that, from a user perspective, database access will be most efficient.

This fundamental distinction is reflected in the expected skills of the computer science and computer information systems students: The CS student has to exhibit a considerable gift for abstract reasoning, but is not expected to interact successfully with as large a variety of application domains. CIS students have a strong understanding of the management, marketing, and accounting facets of an organization, allowing them to work more effectively in an organizational setting. In addition, interpersonal skills and communication skills are more strongly emphasized in the CIS major. CIS students are also better versed in assessing the needs of the end-users, and are therefore more capable of designing software systems that satisfy those needs.

3 Software engineering

The ACM 2013 curriculum [2] describes software engineering as follows:

> Software engineering is the discipline concerned with the application of theory, knowledge, and practice to effectively and efficiently build reliable software systems that satisfy the requirements of customers and users. This discipline is applicable to small, medium, and large-scale systems. It encompasses all phases of the lifecycle of a software system, including requirements elicitation, analysis and specification; design; construction; verification and validation; deployment; and operation and maintenance. Whether small or large, following a traditional disciplined development process, an agile approach, or some other method, software engineering is concerned with the best way to build good software systems. Software engineering uses engineering methods, processes, techniques, and measurements. It benefits from the use of tools for managing software development, for analyzing and modeling software artifacts, for assessing and controlling quality, and for ensuring a disciplined, controlled approach to software evolution and reuse.

Tomayko [9] identifies three periods in the history of software engineering education: the era of single free-standing courses (prior to 1978), the early graduate programs (1978-88), and the rapid spread of graduate programs influenced by the Software Engineering Institute's efforts (since 1988). By the 1970s a number of computer scientists had realized that software complexity was not just a matter of algorithms and data structures, but that software could become overwhelmingly complex due to its sheer size; even straightforward code becomes overwhelmingly complex when it grows to perhaps 300,000 lines of code and 6000 subroutines [e.g. 10]. No software engineering course was mentioned in the ACM 1968 [11] or 1978 curricula, though ACM 1978 recommended CS curriculum consider an optional "Software design and development" course. In 1978, Spencer and Grout [12] saw the need for including a SA&D course in a CS curriculum under the name "software engineering: large systems design". Constantine's [13] classic "Structured design" stressed the need for systems analysis and design to include a software implementation component so students could see the results of their design, and correct problems.

Table 1 –ACM 2013 tier-1 and tier-2 course contact hours per outcome [2]

Software Engineering Outcome Category:	**Tier1**	**Tier 2**
Software Process	2	1
Project Management		2
Tools & Environments		2
Requirements Engineering	1	3
Software Design	3	5
Construction		2
Verification & Validation		3
Software Evolution		1
Reliability		1

Even by the late 1980s the importance of SA&D and SE had not been completely embraced. In 1988, Poole and Callihan [14] argued for the inclusion of a SA&D course in the CS curriculum suggesting that many graduates have "a merely superficial acquaintance with

…SA&D" and that "many CS majors graduate under the misapprehension that SA&D is a career path that has little or nothing to do with computer science".

The ACM/IEEE 2001 CS curriculum detailed a series of software engineering courses such as software engineering, software process improvement, and advanced software development [1]. The ACM/IEEE 2013 curriculum guideline no longer stipulates particular courses but now provides a list of tier-1 and tier-2 software engineering "knowledge outcomes," most of which should be covered somewhere in a computer science curriculum (Table 1).

3.1 Is the initial software engineering course better without programming?

Both Constantine [13] and the ACM/IEEE 2013 curriculum [2] urged that software engineering for computer scientists be taught in conjunction with systems development. However, over the years, the authors have discovered a number of problems with this traditional software engineering pedagogical approach. First, in our experience, CS design courses that involve writing code, quickly devolve into only a small amount of internal design and analysis, with the bulk of the course becoming a conventional coding class. We suspect that computer scientists like writing code but perhaps are not as enamored with what might be perceived as the tedium of analysis and design.

Second, feedback from employers indicate that many of ISU's CS students did not really understand testing procedures, maintenance, version control, task analysis, data flow diagrams, project management tools such as PERT (program evaluation and review technique) and other techniques for developing large software systems. CS students were well equipped for entry level programming jobs but had difficulty moving into team leader and higher level management positions.

4 Systems analysis and design

Systems Analysis has been defined as follows:

> …the specification of what the system needs to do to meet the requirements of end users. In … systems-design … such specifications are converted to a hierarchy of charts that define the data required and the processes to be carried out on the data so that they can be expressed as instructions of a computer program. Many information systems are implemented with generic software, rather than with such custom-built programs" [15].

Pool and Callihan [14] argue that software engineering is a subset of systems analysis and design. They state that: "SA&D concerns itself with methodologies to manage the development and maintenance of computer-based systems in general, including hardware, software, people, plant, and the interfaces that link all the components together into a functionally harmonious whole. Software engineering focuses on software development within the already given context of the larger system [14]. One definition of software engineering includes the terms "analysis" and "design" [15].

ISU's systems analysis and design course – taken by both CS and CIS students – is somewhat unique as all of the CIS students have had at least one development course and the course is taught by a CIS professor with a PhD in information systems, a MS in computer science, and extensive industrial experience building applications and kernel level code. The course outcomes listed in Table 2 have been developed in conjunction with CS faculty.

Table 2: ISU Systems Analysis and Design Outcomes

On completion of the course, systems analysis and design students will be able to do the following:
Explain the fundamental concepts of systems analysis and design.
List and explain the important development methodologies for complex systems.
Demonstrate an awareness of the complexities of requirements determination
Demonstrate the analytical skills required to examine a situation, to understand thoroughly the factors involved, to recognize any problems, and to derive potential solutions
Analyze system requirements and specify system processes and data flows, express requirements in use cases, design user interfaces, and develop a systems proposal
Employ appropriate systems design tools such as structure charts, process specifications, UML, use cases, and dialog flow designers to design a system and its user interface
List and explain the fundamental concepts behind the implementation, testing, conversion, and maintenance of a system.

4.1 Software engineering v. systems analysis and design

As can be seen from the above descriptions of software engineering and systems analysis and design, there is considerable overlap between these courses. Both deal with analyzing requirements, budget, user needs, external and internal design, project management, testing, deployment, training, and maintenance.

However, there are three key aspects stipulated in the ACM/IEEE 2013 software engineering knowledge area that are not covered in ISU's SA&D course; these aspects have to be covered elsewhere in the CS department's curriculum in order for computer science to be able to use the SA&D course instead of a CS SE course.

First, CS students focus on developing systems software rather than simply application software, so they need to understand how to design systems with parallel and real-time aspects; the design approaches that work with typical business applications may not work well with other types of software. Second, CS students need exposure to considerably more object-oriented design than can be taught in a SA&D course with CIS students. Third, computer scientists feel that software project management is best learned as part of a team building a fairly large software package. This is because students need to be able to see the correspondence between their internal designs, resulting software, and project outcomes before than can learn to improve their design skills. The ACM/IEEE 2013 curriculum states: "students learn best at the application level … by participating in a project". While the systems analysis and design course involves a semester team project made up of a series of deliverables, it emphasizes the analysis and design of a system and stops short of implementation.

5 Systems analysis and design in CS

Based on the preceding analysis, it was decided that instead of a conventional computer science introductory software engineering course, ISU's CS curriculum would require the CIS systems analysis and design course. CS students are expected to have first completed CS-I, CS-II, computer organization, and advanced English class to insure they have sufficient writing ability. The CS-I course, requires structure charts [13] for many of the assignments. CS students also take a third programming class that focuses on advanced object oriented design and development and includes a final project that requires extending an existing system of about 25 classes with new features and an additional 15 classes or so. Finally, there is a capstone course titled "advanced software engineering and senior project" where they are required to analyze, design, develop, and test an extensive system as part of a team project.

The SA&D course involves a team analysis and design project with a series of scheduled deliverables. Students choose their own teams, with a peer evaluation at midterm that functions as an early warning system – and a second peer evaluation that is factored into the final individual grade. Peer evaluations involve each team member submitting a peer evaluation form assessing the contributions of each team member with regard to the percentage contributed by each team member toward the successful completion of all phases of the project.

While ISU's systems analysis and design course does not cover all the software engineering knowledge outcomes in the ACM/IEEE 2013 curriculum document, we feel it falls within the broader, historic tradition of much of what is covered in software engineering courses, and combined with other required CS courses, provides a solid software engineering background as well as a more extensive SA&D experience.

6 Experience and feedback

We now have had CS students taking our SA&D course for several years. Feedback from CS students is positive. One group ended up with a mix of CIS and CS students that reported working well together. The fact that the course focused on analysis and design techniques unknown to either CIS or CS students and did not involve any coding put both groups of students on a somewhat even footing. One student commented, "We didn't segregate jobs based on major either, because we were all learning the same material for the first time."

Having the SA&D course taught by someone with a MS in computer science as well as a PhD in CIS and extensive real-world software development experience was mentioned as significant. The software engineering knowledge outcomes incorporated into a number of required CS courses as well as requiring the SA&D course seems to have provided a solid practical SE background for the students.

Our experience has shown that while many CS graduates enter the workforce as systems software developers, the bulk are hired by businesses to build applications software. For the latter group, experiencing a systems analysis and design course has proven invaluable, providing knowledge of not only software development techniques, but also familiarity with the business context inherent in a systems analysis and design class. This makes such students better prepared to excel in the workforce. CS majors have gained exposure to concepts like business rules, varied stakeholders, and business requirements elicitation. They have learned the importance of accounting and financial data, and gained a better awareness of the importance of written and oral communication. CS students have also be exposed to students with a business background, and, if a live project is used, exposure to an actual business. CS students have learned that there can be considerable differences between business application software – which provides the bulk of employment for CS graduates – and the more traditional CS systems software.

Perhaps most importantly, they also have become aware of how critical application software can be to the success of a business, as compared to system software that may be used across a variety of businesses. Based on this, a number of CS students have decided to start taking business courses in addition to their CS majors and some graduates are in the process of completing their MBA.

While this single case is based on a university with a unique history and with a faculty with a specific skill set, the outcomes reflect a broader question: Is the strict division between CS and IS relevant to the modern graduate? It has been shown that very little of the pure CS program is lost through the substitution that has been tried. Undergraduates value their exposure to a broader curriculum and contact with both business students and faculty. At least in our alumni community the tasks of the recent graduate are not those of the traditional CS graduate but require a better knowledge of the context of the systems being developed. We are sure that this change has been beneficial to our students and we suspect that this alternative would be of benefit elsewhere.

7 References

[1] ACM/IEEE Computing 190 curricula 2001 final report. http://www.acm.org/sigcse/cc2001.

[2] ACM/IEEE-CS Joint task force on computing curricula 2013 strawman draft.

[3] Chen PPS The entity-relationship model – towards a unified view of data ACM TODS 1,1 9-36 Mar 1976.

[4] Codd, EF "A relational model of data for large shared data banks". 13, 6 377–387 1970.

[5] Bagui S, Earp R Database design using entity-relationship diagrams, Second Edition, Boca Raton, FL: Taylor & Francis 2012.

[6] Ramesh B, Pries-Heje J, Baskerville R Internet software engineering: a different class of process. Annals of Software Engineering 14 169–195 2002.

[7] Austing R H et al., Curriculum ’78: Recommendations for the undergraduate program in computer science – a report of the ACM curriculum committee on computer science, Comm. ACM 22,3 147-166 Mar 1979.

[8] ACM/IEEE-CS Joint Interim review task force. 2008. Computer Science Curriculum 195 2008: An Interim Revision of CS 2001.

[9] Tomayko JE Forging a discipline: An outline history of software engineering education, Annals of Software Engineering 6, 3-18 1998.

[10] Brooks FP. The mythical man-month Reading, MA: Addison-Wesley Publishing, 1975.

[11] Atchison WF, ACM curriculum committee on computer science. Curriculum 68: recommendations for academic programs in computer science. *Comm. ACM* 11,3, 151-197, Mar, 1968.

[12] Spencer JW, Grout JC Systems analysis and design in a computer science curriculum, SIGCSE Bulletin, 24-27 1978.

[13] Yourdon E, Constantine LL Structured design: fundamentals of a discipline of computer program and System Design Yourdon Press, NE 2ed 1978

[14] Poole BJ, Callihan HD Systems analysis and design: an orphan course about to find a home SIGCSE bulletin 20,2 54-64 Jun 1988.

[15] Encyclopedia Britannica. Encyclopedia britannica online academic ddition. Encyclopedia Britannica Inc., web, Mar 2013.

[16] Shoorman ML The teaching of software engineering SIGCSE 83, 15,1 66-71 Feb 1983.

Including Computer Design in Early Hardware Study in Computer Science

Hassan Farhat
University of Nebraska at Omaha, Omaha, NE, USA

Abstract

Based on the ACM/IEEE-CS Computer Science recommendation, a typical curriculum may incorporate one or two classes in the hardware track computer organization, and computer architecture. With restrictions on the number of credit hours, a detailed study of computer design may have to be skipped. Instead, following digital design, the study may focus on the organizational level of the computer (single and multiple-core).

In this paper we introduce a simple computer design that can be included in the curricula and studied in the later part of a first course in hardware design or early part of a second course on computer architecture. The work is related to earlier work on multiple-cycle and single-cycle computer design. Here, however, we present the design as a single- cycle design in Multisim. Unlike the single-cycle design done in Altera, the single-cycle design in Multisim can be introduced earlier in the topics coverage. In addition, the proposed single-cycle design includes program entry using switch inputs instead of memory initialization files.

The contribution of the paper is instructional in nature. By looking at the design of a single-cycle computer, instead of multiple-cycles in Multisim, the students can be introduced to a somewhat set of advanced topics. This can be done in a later part of a course on digital design or early part of a course on computer architecture. As a result, the later discussion on multi-core processors can be presented with the students having better understanding of a single-core processor design.

1 Introduction

The study of digital circuits and central processing units design is a topic of interest in computer engineering as well as computer science. The topic is covered at many levels. It can be covered at the actual chip design level [1] as well at the digital logic level [2] to [7], computer organization [8,9] level and computer architecture [10] to [12] level.

In the study of computer hardware, the ACM recommendations[1] is to cover a minimum of 36 contact hours in the field of digital design, computer organization and computer architecture. The number of contact hours is less than the number of contact hours provided in a sequence of two courses, each is a semester 3-credit hour course. As a result, many topics on actual digital design, computer organization and computer architecture may be covered briefly.

We feel that in a curriculum on hardware track, it is beneficial that students study the design of a simple computer. Among the computer design package used in the study of digital design are two packages Multisim [13] and Altera [14]. To meet the constraints on the number of topics that are covered and the need to study computer design, we proposed a simple single cycle computer design in the Altera package [15]. Similar work on more complex multiple- cycle design was done in Multisim. In this paper we show the design of the simpler single-cycle computer in Multisim. The Multisim package is suited to digital design and can be used to introduce the simple design in a first course in the hardware track. We illustrate by looking at a sample user interface.

Fig. 1 shows a simple design in the Multisim. The same user graphics interface is used to: a) complete the design, and b) simulate its function. There is no need to change the interface window (for simulation for example) or download the design to a design board. Instead, simulation can be performed through the use of switches as inputs (for a given switch, its value is controlled by a key press on a keyboard) and the use indicators for output responses.

The circuit is a realization of a Moore finite state machine that outputs a one whenever the binary input entered so far is divisible by 5.

Once the design is completed, the user can click on the upper left button to start simulation. There is no need to generate simulation waveforms and initiate compilation. Instead, buttons on the left are used to simulate the input and the clock. In the figure, by pressing the x key on the

[1] The number of hours were reduced in CS2013, ACM/IEEE-CS Joint Task Force document

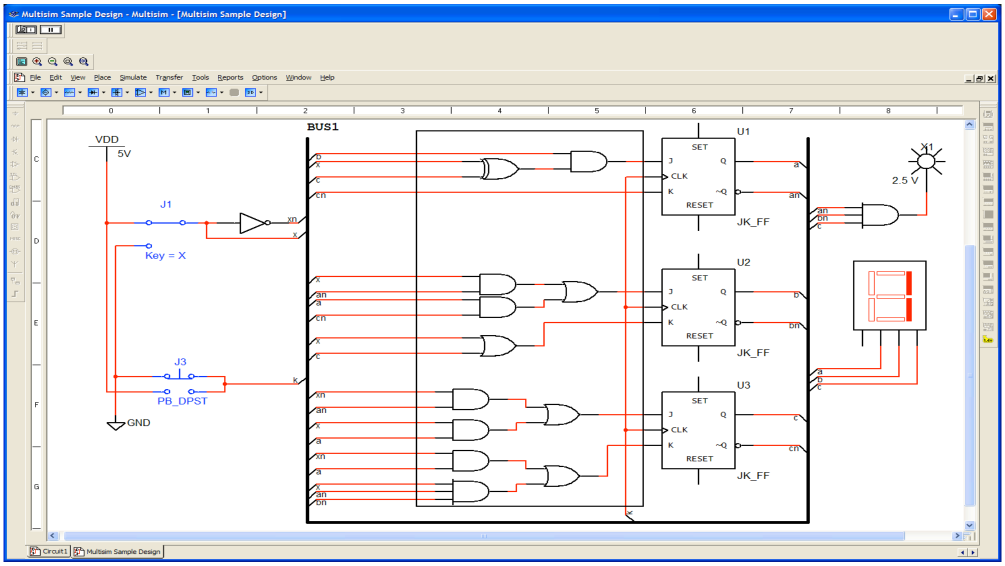

Multisim Sample Design - Multisim - [Multisim Sample Design]
File Edit View Place Simulate Transfer Tools Reports Options Window Help
BUS1
VDD
5V
J1
Key = X
J3
PB_DPST
GND
U1
U2
U3
SET
CLK
RESET
JK_FF
X1
2.5 V
Circuit1
Multisim Sample Design

Figure 1: Multisim user interface

keyboard we can adjust the input to 0 or 1. To read (sample) the x value, we simulate the clock input by pressing the keyboard key corresponding to the pushbutton on the lower left part of the circuit. The circuit output response is seen on the virtual indicators shown on the right. While packages such as Altera may be suited for advanced designs, Multisim can provide fast easy to test designs at a lower level. We use this package to design the single-cycle computer as proposed in [15].

Unlike [15], we use switches to input both instructions and data to the simple computer. We are also able to modify instruction memory contents while the computer is running and, as a result, mix both modes of instruction entry as well as instruction execution. This gives students better understanding of the computer functionality.

The paper is organized as follows. In section 2, we review the instruction set of the single-cycle computer. In section 3 we look at the design procedure. Section 4 contains the proposed design in Multisim and explains the input/output functionality. The conclusions are given in section 5.

2. The Instruction Set of the Single-Cycle Computer

To keep the instruction set simple, we: 1) choose an accumulator-based architecture, 2) have one addressing mode (direct addressing), 3) choose an 8-bit instruction size, and 4) select a minimal instruction set. The set has seven instructions and is chosen to conform to instruction set completeness. A 4-bit opcode field is chosen for possible expansion.

A small memory is chosen, 256 words. The rationale is to complete design process without having to keep track of all the needed details when compared to larger instruction sets. The complete instruction set of the computer, [15], is given in Table 1.

Table 1

Instruction	Meaning	Assigned Opcode
LDA XX	AC ← M[XX]	0XX
STA XX	M[XX] ← AC	1XX
ADD XX	AC ← AC+M[XX]	2XX
NAND XX	AC ← NOT (AC AND M[XX])	3XX
BUN XX	PC ← XX	4XX
SKZ XX	If AC = 0 then PC ← PC + 1	5XX
INC	AC ←AC + 1	6XX

Expanding the functionality of the computer can be accomplished by adding macros that perform logic and arithmetic operations. We give these as exercises. Example: To form "NOT A" we write two assembly level instructions "LDA A, NAND A". Similar exercises are given to expand the functionality of the computer. The instructions given above form a small subset from a popular earlier AC-based instruction set architecture (PDP 8).

Since the design is a single-cycle design, we need to complete instruction execution during the same cycle. Since the instruction needs to be fetched from memory, the instruction fetch and instruction execution must be completed in the same cycle. We accomplish this by having two memories, the instruction memory and the data memory. The presence of the data memory eliminates the need to for a memory buffer register. A schematic of the simple design is shown in Fig. 2.

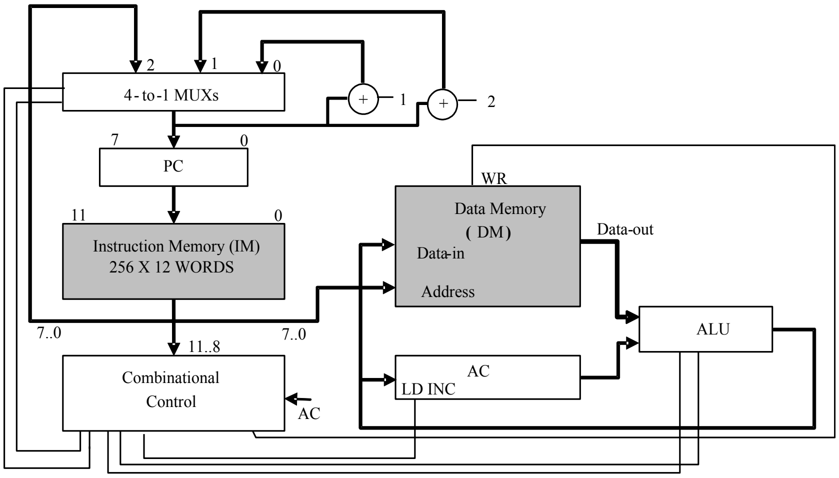


Figure 2: Schematic of the single-cycle computer

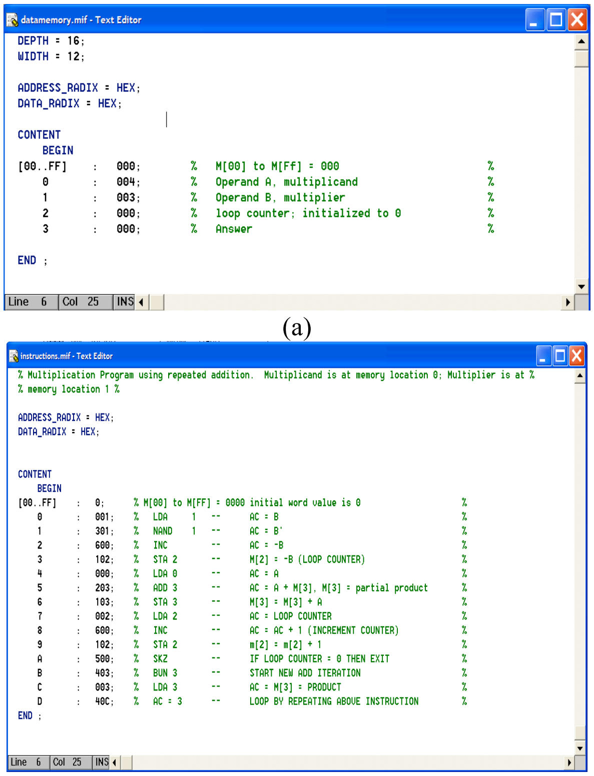
3 The Computer Design

The internal computer design follows similar steps as used in the design using the Altera package. Here, however, we need to enter data into the computer memory manually. In Altera, the memory modules can be initialized to specific values using ".mif" files. As a result, the input interface is absent. In this section we briefly review the design steps.

Since the computer design is a single cycle, the design of the control unit of the computer is combinational in nature. Due to the size of the instruction set, the simplest realization of the control unit is as a ROM. The ROM outputs the needed signal values as control word. The control word contains several fields corresponding to the different units of the computer. Table 2 (next page) shows the control word values based on the type of instruction of the computer.

In the table, the PC_{MUX} field determines the next program counter value. Computer designs assume the fetch of next instruction occurs sequentially. The sequential mode is changed by branch instructions (if statements, loop statements, etc.). In the instruction set the two instructions that affect the program counter contents are the branch unconditional (BUN x) instruction and the increment and skip if zero (ISZ) instruction. The multiplexer input to the program counter chooses among the three options.

Once the design is completed in order to test functionality, we initialize the memory to specific instructions and data using memory initialization files as shown in Fig. 3 (a) and (b).



```
DEPTH = 16;
WIDTH = 12;

ADDRESS_RADIX = HEX;
DATA_RADIX = HEX;

CONTENT
   BEGIN
[00..FF]    :   000;      %   M[00] to M[Ff] = 000                  %
   0        :   004;      %   Operand A, multiplicand               %
   1        :   003;      %   Operand B, multiplier                 %
   2        :   000;      %   loop counter; initialized to 0        %
   3        :   000;      %   Answer                                %

END ;
```

(a)

```
% Multiplication Program using repeated addition.  Multiplicand is at memory location 0; Multiplier is at %
% memory location 1 %

ADDRESS_RADIX = HEX;
DATA_RADIX = HEX;


CONTENT
   BEGIN
[00..FF]    :   0;      % M[00] to M[FF] = 0000 initial word value is 0          %
   0        :   001;    %   LDA    1   --      AC = B                            %
   1        :   301;    %   NAND   1   --      AC = B'                           %
   2        :   600;    %   INC        --      AC = -B                           %
   3        :   102;    %   STA 2      --      M[2] = -B (LOOP COUNTER)          %
   4        :   000;    %   LDA 0      --      AC = A                            %
   5        :   203;    %   ADD 3      --      AC = A + M[3], M[3] = partial product      %
   6        :   103;    %   STA 3      --      M[3] = M[3] + A                   %
   7        :   002;    %   LDA 2      --      AC = LOOP COUNTER                 %
   8        :   600;    %   INC        --      AC = AC + 1 (INCREMENT COUNTER)   %
   9        :   102;    %   STA 2      --      m[2] = m[2] + 1                   %
   A        :   500;    %   SKZ        --      IF LOOP COUNTER = 0 THEN EXIT     %
   B        :   403;    %   BUN 3      --      START NEW ADD ITERATION           %
   C        :   003;    %   LDA 3      --      AC = M[3] = PRODUCT               %
   D        :   40C;    %   AC = 3     --      LOOP BY REPEATING ABOVE INSTRUCTION      %
END ;
```

(b)

Figure 3: (a) Initial data memory contents, (b) initial instruction memory contents

Table 2

	Control Address								Control Word Contents				
Instr.	AC_{11}	AC_{10}	……	AC_0	IM_{11}	IM_{10}	IM_9	IM_8	PC_{MUX}	ALU	AC_L	AC_{INC}	WR
LDA X	x	x	……	X	0	0	0	0	00	00	1	0	0
STA X	x	x	……	X	0	0	0	1	00	01	0	0	1
ADD X	x	x	……	X	0	0	1	0	00	10	1	0	0
NAND X	x	x	……	X	0	0	1	1	00	11	1	0	0
BUN X	x	x	……	X	0	1	0	0	10	Xx	0	0	0
SKZ	0	0	……	0	0	1	0	1	01	Xx	0	0	0
SKZ	1	x	……	X	0	1	0	1	00	Xx	0	0	0
SKZ	x	1	……	X	0	1	0	1	00	Xx	0	0	0
……	……	……	1	……	0	1	0	1	00	Xx	0	0	0
SKZ	x	x	……	1	0	1	0	1	00	Xx	0	0	0
INC	x	x	……	X	0	1	1	0	00	Xx	0	1	0
Not used	x	x	……	X	X	1	1	1	xx	Xx	x	x	x

4 The Multisim design

To design the computer using Multisim, we could follow similar steps as presented in section 3. In earlier work we used VHDL for the output requirements and used the Altera package capabilities to initialize memory using ".mif" files. Here unlike previous work, however, we work at the schematic capture level for internal computer design, for the inputs, and for the outputs. Our objective is to move the discussion to the schematic capture level in digital logic. We will then need to determine the input requirements to the modeled computer. The input design requirements include determining: a) access to the instruction and data memories, b) the control unit, and c) the needed registers for interactive program entry and run.

We first give the block diagram of the design as realized in Multisim. We then discuss the added functionality to the design with sample program entry and runs instructions. Fig. 4 (next page) shows the completed design.

As can be seen from the figure, the single-cycle computer is enclosed in block diagram labeled "SINGLE CYCLE BLOCK" the input to the computer are the set of switches seen on the left side of the design. The outputs are shown using seven-segment-displays and an indicator. The computer design is completed so as to directly control memory inputs for both data and instruction and for writing. In addition, the design is completed so as to continually display (read) the contents of the memory components and other needed registers. This simplifies debugging and allows the user immediate access to the datapath aspects of the computer. While the program is running, the design can, in addition, to single-step instruction execution, the design can alter specific instructions in the program and measure the effects of the changes.

We briefly illustrate the functionality of the computer. We start with the output part. The address and contents of the data memory are displayed using the seven-segments displays labeled, DPC and DM respectively. Similarly the address and contents of the instruction memory are labeled IMPC and IM. The arithmetic logic unit contents and the AC contents are displayed on the right side of the figure. The figure contains one additional output corresponding to a status bit of AC (AC = 0).

For the input part, the computer is placed in one of two modes: data/instruction entry mode, and program run mode. The switch labeled W determines the mode. For the instruction entry mode we: a) set W to 1; b) adjust the instruction value using the keys labeled 0 through 7 (generate an 8-bit word); c) set the IMPC value using the pushbutton C_IMCLK; and d) write to instruction memory using the I_IMWR pushbutton. For data memory write we repeat the same procedure as above for parts (a) and (b). For parts (c) and (d) we, respectively, choose the K_DMCLK and D_DMWR.

In program run mode we: a) set W to 0; b) clear the instruction program counter (start at instruction zero); and (c) press the pushbutton labeled Key = Space to clock the circuit (this corresponds to pressing the spacebar on the keyboard). Following each spacebar key press, the computer advances to the next instruction in the program. If desired, to check the contents of data memory at any point in run mode, we simply set the W value to 1 and advance the data memory program counter (without writing data) to the proper data memory word. Similar procedure can be applied to read instruction memory.

6 Conclusion

In this paper we looked at the design of a simple interactive computer as an instructional tool. Unlike previous work, the computer design is presented in Multisim and is a single-cycle computer. The computer design is completed at the gate and register level. This avoids including hardware description languages such as VHDL used in the output interface as done in earlier work. In addition, the inclusion of the input interface, gives the user additional

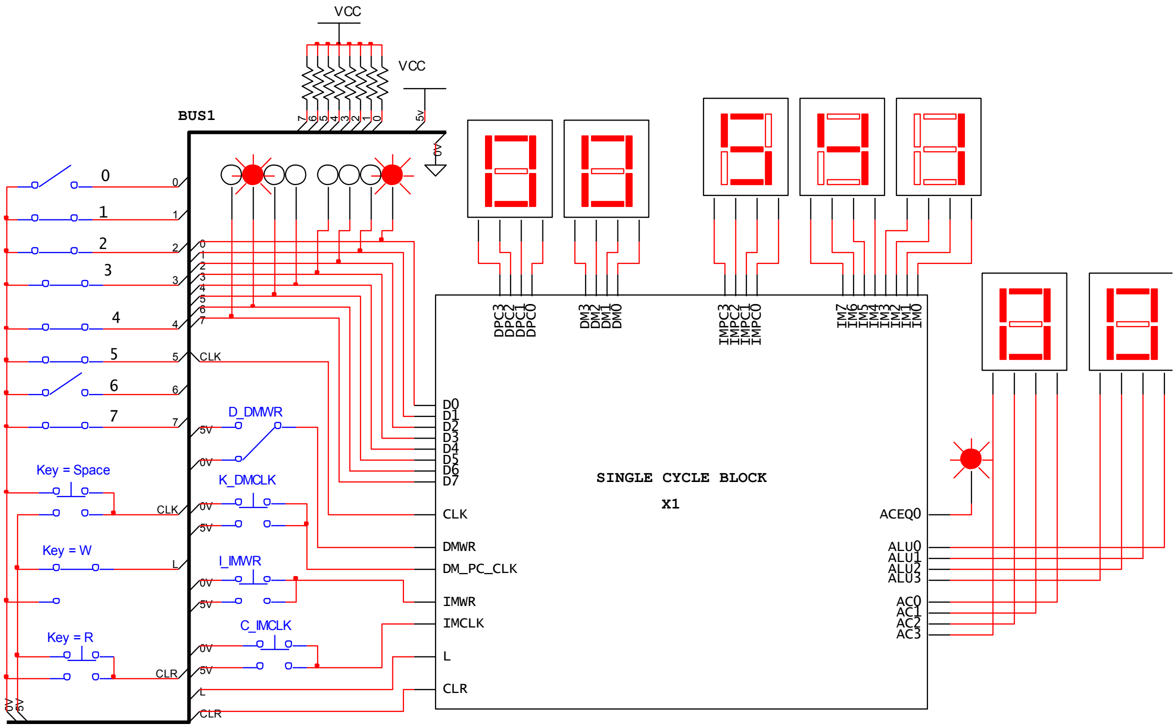


Figure 4: Block diagram of the single-cycle computer in Multisim

flexibility where data and instruction memory can be modified during program run. This avoids the need to use memory initialization files as done in Altera.

Due to constraints on the number of classes that are required in the hardware track in computer science, students may not be able to experiment with designs at this level. The work is intended so as to include the topic in the later part of a course on digital design and computer organization. Alternatively the topic can be included at the early part of a second course in computer architecture. Understanding the details of the design helps the students in better understanding of advanced topics on multi-core processors.

References

[1] N. Weste and D. Harris. "CMOS VLSI Design: A Circuits and Systems Perspective 4th edition". Addison Wesley, 2011.

[2] S. Brown and Z. Vranesic. "Fundamentals of Digital Logic with VHDL Design with CD-ROM, 3rd edition". McGraw-Hill, 2009.

[3] T. Floyd. "Digital Fundamentals: A System Approach". Prentice Hall, 2013.

[4] W. Kleitz. "Digital Electronics A Practical Approach with VHDL, 9th edition". Prentice Hall, 2012.

[5] M. Mano and M. Cilette. "Digital Design, 5th edition". Prentice Hall, 2013.

[6] R. Tocci, N. Widmer and G. Moss. "Digital Systems Principles and Applications, 11th edition". Prentice Hall, 2011.

[7] J. Wakerly. "Digital Design Principles and Practices 4th edition". Prentice Hall, 2006.

[8] D. Patterson and J. Hennessy, "Computer Organization and Design. Revised 4th edition". Morgan Kaufmann, 2011.

[9] W. Stallings. "Computer Organization and Architecture. 9th Edition", Prentice Hall, 2013.

[10] D. Harris and S. Harris. "Digital Design and computer Architecture". 2nd edition, Morgan Kaufman, 2013.

[11] J. Hayes. "Computer Architecture and Organization 3rd edition". McGraw-Hill, 1998.

[12] J. Hennessy and D. Patterson. "Computer Architecture: A Quantitative Approach 5th Edition". Morgan Kaufmann, 2011.

[13] Altera, www.altera.com.

[14] Electronics Workbench. www.electronicsworkbench.com.

[15] H. Farhat. "A Simple Computer Design with Monitor Interface: Integrating Hardware and Software in Early Research in Computer Science". Proc. ICSNC, 218-222, 2008.

Cloud Computing in the Curriculum: A Current Perspective

Kathy J. Liszka, Chien-Chung Chan, Michael L. Collard,
Drew Guarnera, and Nicholas Bordo

Department of Computer Science
The University of Akron
Akron, Ohio 44325-4003
{liszka, chan, collard}@uakron.edu
{dt.guarnera, bordo.nicholas}@gmail.com

Abstract

Cloud computing is one of the most disruptive technologies for the near future, changing the way we think about things and the way we do business. We designed a project-driven course that pairs students and engineers from local industry to work together as early as possible in a mentoring hierarchy. Students learned the fundamental principles, skills, and paradigms of cloud computing by developing a project for solving industry-oriented problems. This paper describes the course goals, content, a sample student project and outcomes of this ongoing venture with a local company.

Keywords: cloud computing, project-based curriculum, REST, web services

1. Introduction

Cloud computing is a nebulous term that refers to using the Internet to store data, run programs, and much more. However, it has yet to significantly infiltrate the college curriculum. Students from The University of Akron and software engineers from Canton-based Diebold Inc., leaders in ATMs and security systems, are engaging in an experimental series of three courses in cloud computing to learn how industry can maximize cyberspace resources. A group of selected computer science students are receiving specialized training to help industry. This paper details the content and approach taken in the first of these courses.

The objective is to pair engineers from industry with students in a project-based course sequence that will prepare them to contribute from the first day on the job. Some were selected for coop opportunities at Diebold as a result of their participation in these courses. This serves as a mechanism to attract and retain top-tier talent at Diebold. If this model is successful, we hope to continue the relationship with Diebold and to establish similar relationships with other industry partners.

The course is project-based but the student population is divided into two different levels of expertise. Roughly half of the class (excluding the engineers) has experience with web-based programming through our curriculum and were well into their junior year or first semester seniors. The other students have finished at least two semesters of the core programming sequence but did not have any web-based programming skills. The teams were composed of one Diebold engineer, one more experienced, upper-level student and one less-experienced student. This created a natural mentorship relationship. The engineer's duty is to give guidance on the project as a whole. The more senior-level student acted as mentor for the less-experienced student in the technical realm.

The paper is organized as follows. Section two describes the lecture content used in the first course. This is intended to help other faculty members design their own course. Section three details the projects assigned to teams. In the fourth section, we give a brief summary of one of the actual projects completed in the first course and draw conclusions in the final section.

2. Course Overview

With the difference in experience among the students in the class in web-based programming, the course started with overview topics. Indeed, the lectures were basic introductions to areas that students need some minimal competency with and awareness of how they fit into an overall project. This was followed by more complex areas so that teams could successfully create a web service application. The course topics are briefly described.

2.1 Introduction to Cloud Applications

This material covers a brief history of cloud computing and introduces the major terms "*software/platform/infrastructure*- as a service". The terms and definitions are taken from The NIST Definition of Cloud Computing [1]. The different types of clouds (public, community, private, and hybrid) are presented here. The lecture is intended to set the stage for the evolution of computing, moving from where something is done to on how it is done.

2.2 HTML5and CSS

HTML markup and CSS are covered in one lecture including an introduction to the newer features offered with HTML5. Tutorials from w3schools are introduced for students to delve deeper into any particular topic that they are interested in using for their first project [2]. Emphasis is on client-side computing as a starting point although the features of HTML5 that require server-side support are demonstrated. Semantics and the structure of the documents (DOM) are introduced [3].

2.3 JavaScript

JavaScript is covered quickly, as all students in the class have at least two semesters of programming experience in either C++ or Java. The second project incorporates adding dynamic functionality to the web page, but the page still only supports client side computing.

2.4 Server Side Web Applications

Next, we discuss server-side web components and the web application lifecycle. We go through process for creating, deploying, and executing a web application. They get an overview of the Java EE (J2EE) platform [4], and then work with installing their own Tomcat servers. Java Server Faces (JSF) is an interface framework used to build web applications. Java Server Pages allow code snippets to be inserted directly into a document [5]. Both of these technologies are presented and installed in a hands-on lab. Live examples are demonstrated and worked with by teams.

2.5 JDBC

Most students in the class do not have a background in database management, so a brief introduction to SQL is given followed by coverage and a lab on the JDBC API. This is used to provide data access to Java applications, in particular the web services used for the team projects.

In a hands on lab, students start with adding JDBC drivers for the databases MySQL and MS SQL to Eclipse IDE. They learn how to establish connections to a database and the MS SQL 2005 server. Then they create a database table and run simple queries, then show the query results on a web page. The hands-on labs are to ensure a starting minimal skill set going into each stage of their project.

2.6 Data Interchange Formats

The easiest way to decouple a system is to use a data format to exchange data, rather than use direct calls. In general, decoupled systems are more flexible, easier to develop, modify, and enhance, and generally easier to evolve. Students learn about comma-separated values (CSV), field-value pairs, and block structured formats. Emphasis is placed on XML [7] and JavaScript Object Notation (JSON) [8].

2.7 RESTful Web Services

Representational State Transfer (REST) is a popular web service architectural design model [9]. Team projects evolved through use of JavaScript, JSON, Java Server Pages, Java beans and SQL to create a RESTful web service.

2.8 Google App Engine

The final topic covered was the Google App Engine (GAE) [10]. This choice was made in part because it is a free resource and also due to the interest by students. They downloaded and installed the GAE SDK Eclipse plugin, created a simple GAE Guestbook application from the tutorial and ported their app to GAE [11]. This proved to be the most challenging part of the course. All teams were able to deploy locally, but only one team successfully deployed in Google.

3. Implementation

In this section, we present the project descriptions, objectives and constraints. At the end of the first five projects, teams gave a formal 15 minute presentation with the exception of the last one. Part 6 comprised the final exam and was given to the entire faculty team as well as the industry sponsor. A written report was submitted for each team as well as a brief individual report by each student describing their individual contribution and summarizing what they learned.

3.1 Store Front

You are a mom and pop business with a product to sell. You need to develop a web presence to enhance your business.

The objective of the first project was to create a series of web pages using HTML5 and CSS. The requirements follow.

- Use HTML5 to design a web page for a business decided up by the team.

- Introduce the business (i.e., a picture of the building or logo, address, description of product line -- whatever is appropriate) on the main web page.

- Include links to at least two other pages that display products with information.

- Use CSS to create a uniform look and feel for the web pages.

- Incorporate two of the following: a short video, audio, canvas, SVG, or geo location.

A potential project is a pizza shop chain with multiple locations. One page would be pictures/maps/addresses of the various locations. A second page could be the lunch/dinner menu and a third the takeout menu. A coffee shop may incorporate one page or caffeine products, another with promotional (t-shirts with the logo, different cup designs). Other ideas are an online Halloween shop; an online hardware store; an online customized t-shirt store (athletic teams for example); or an online used book exchange.

3.2 Window Display

Use your Store Front to put life into your web site.

The objective of the second project was to add client side, dynamic functionality. At this point, the students knew some JavaScript and how to access the DOM. They couldn’t connect with a database yet, but they could leave hooks (unfinished, nonfunctional pieces). Now they can make their web site more interesting. The students are encouraged to spend time at the w3schools tutorials [2]. They weren’t limited by lecture content, but they were limited to client-side technology for this project so that they stayed focused.

3.3 Taking Orders

You’re getting ready to launch your website. You need to start taking customer orders.

The objective of the third project was to convert their window-display application into a web application deployed on Tomcat. They needed to provide an order form for their store. Orders were to be stored on the server side in an external text file. Each order was to be assigned a unique order number for identification purpose. They were allowed to add other fields for an order such as customer name, date, time, items, and so forth. They also needed to provide a feature allowing customers to review their orders and make changes. We encouraged them to consider implementing a shopping cart for customers to place orders.

3.4 Processing Orders

Time to make money!

The purpose of this project was to create a data store for their orders. Based on the third project, they used a database for storing information for orders in order to provide further business intelligence analysis. Teams created a simple relational database for storing orders for the store. MySQL was introduced as an integral part of this release. They also needed to design with the model-view-controller (MVC) software pattern in mind [12].

3.5 Going Google!

Go crazy, go Google!

The final working portion of their project was to port their store to the Google App Engine (GAE). They ported their web sites to a local GAE development environment using the Eclipse IDE. Once that was running with good results, it was optional to set up a Google Account and deploy there.

3.6 Stockholder Meeting

Time to report to your funders.

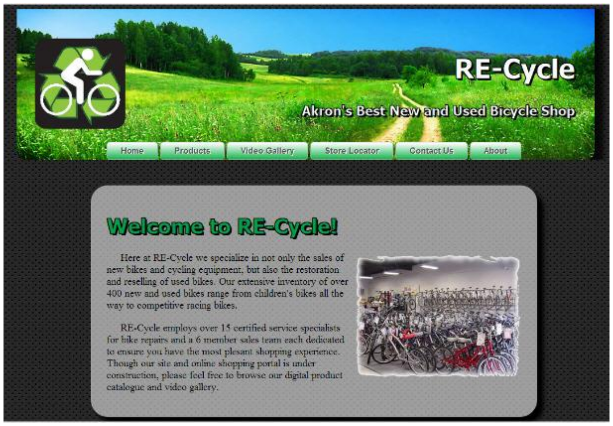
In lieu of a final exam, teams gave a formal presentation to executives at Diebold who funded this project. Students summarized their accomplishments and showed the progression through the technologies learned and used. Engineers wrote an evaluation of the team as a whole and on each student member. Their input was part of each student's grade. It also gave us feedback on what worked well for them in the class and where improvements could be made. The engineer evaluations were overwhelmingly positive.

4. The RE-Cycle Project

There were 14 students and 6 engineers enrolled in the course that were divided into 6 teams, each one led by an engineer serving as a mentor. The degree of freedom given to the class made each project completely unique. We selected one project to present here. The title of it is RE-Cycle, a new and used online bicycle store. The student team members were Nicholas Bordo and Drew Guarnera, two of the authors on this paper. This team reflected the standard team assignments of the course with one of the students a having just completed a Computer Science II course, and the other a first-semester senior. As with all of the teams, the students did all of the coding. The Diebold engineers served in advisor/mentor roles. All of the teams did an excellent job, but this particular team seemed to set the standard for the class.

In the initial phase, the team gave their web page a Web 2.0 appearance while maintaining visual consistency and clarity. They established a basic formatting template across all of the pages including future ones to be created. Figure 1 shows the main page for RE-Cycle. Students were encouraged to investigate and select their own editors. This team found Sublime Text 2 [13] useful. It allowed a certain degree of auto-completion which helped ensure the correct properties were being used and it offered suggestions based on the style elements they were working with. HTML5 features included were video playback and geo-location.

Using JavaScript in phase two, the team replaced the static navigation menu with an attractive drop-down menu from an off-the-shelf package. A lesson learned is that it took more work than the team expected to make it work the way they wanted. They applied a standard geo-location script to show the location of their store in relation to global positioning coordinates on a Google map. They also added scripts to display Google driving directions. The final part of this phase was to add simple text input validation on a dummy contact submission form to demonstrate accessing the DOM using JavaScript and changing their HTML markup dynamically based on valid/invalid user input.



Welcome to RE-Cycle!

Here at RE-Cycle we specialize in not only the sales of new bikes and cycling equipment, but also the restoration and reselling of used bikes. Our extensive inventory of over 400 new and used bikes range from children's bikes all the way to competitive racing bikes.

RE-Cycle employs over 15 certified service specialists for bike repairs and a 6 member sales team each dedicated to ensure you have the most plesant shopping experience. Though our site and online shopping portal is under construction, please feel free to browse our digital product catalogue and video gallery.

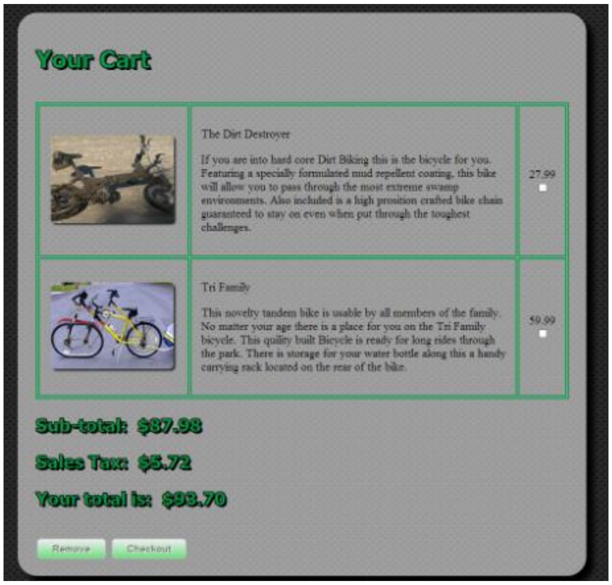
Figure 1. Main page of RE-Cycle for phase 1.

The third phase marked an evolution for the project. They migrated their site to a local Tomcat server and began the process of taking orders, as in Figure 2. Taking a "learn as you go" attitude, they configured their server and integrated the new platform into their workflow. They switched to the full Eclipse IDE [14] and began converting their pages into JSP pages.

The other innovation was establishing a central control to handle get and post requests separately to keep code more manageable. The general purpose of the controller was to determine what the request was, gather the necessary information needed to process it, pass the data to a bean for further handling, and setup the data to be dispatched to the client. The beans would then act as models and handle the business logic and specifics of the site while the JSP pages provided a view for the user to see the representations of the data, all while the controller transported the data back and forth between the models and the views.

This team was very careful to follow a simple Model View Controller (MVC) architecture that they would later describe as one of the most helpful aspects of the project. The compartmentalization of all the tasks into specific zones of responsibility made a significant difference when it came to expanding functionality, finding and resolving bugs, and connecting client side and server side features.

JSON files were used for storing product and cart data. Product information was read into a JSON file using Google's Gson [15] (a JSON parsing and serialization library). Another JSON file was created when the user assembled a cart. They used JSTL to connect the pages and the data needed to display from the server.



Your Cart

	The Dirt Destroyer If you are into hard core Dirt Biking this is the bicycle for you. Featuring a specially formulated mud repellent coating, this bike will allow you to pass through the most extreme swamp environments. Also included is a high prosition crafted bike chain guaranteed to stay on even when put through the toughest challenges.	27.99 □
	Tri Family This novelty tandem bike is usable by all members of the family. No matter your age there is a place for you on the Tri Family bicycle. This quality built Bicycle is ready for long rides through the park. There is storage for your water bottle along this a handy carrying rack located on the rear of the bike.	59.99 □

Sub-total: $87.98

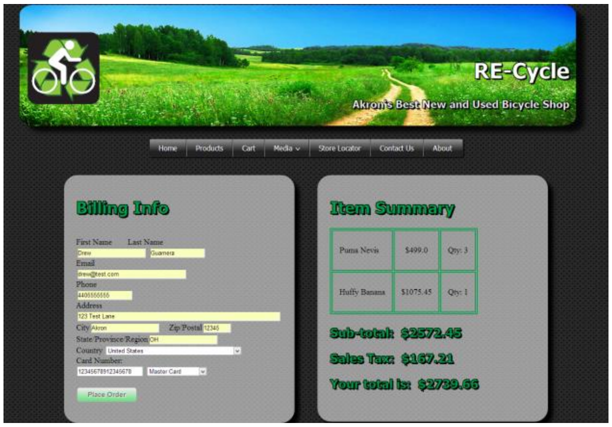
Sales Tax: $5.72

Your total is: $93.70

Figure 2. Sample order page of RE-Cycle for phase 2.

Until phase four, we did not allow relational databases as an option intentionally. There was already a steep learning curve for roughly 1/3 of the class so we were clear in requirements for each project phase. In the fourth component, students used MySQL [16]. They used a database manager to handle the details of database connectivity and operations as well as centralizing all of the potential problems spots with that external interaction to assist in any necessary debugging. With this setup, Java beans only need to make a request for either data processing or a request for the data itself, and the database manager would in turn connect to MySQL and perform the appropriate actions.

The data was designed to be stored in the cart using the user's session id in anticipation of letting users create accounts and login. It would allow a more natural transition to associating the cart with the user currently logged in which would be tracked via session data. With the backend code complete, they connected the modified back end to the new billing page and cart pages. As an added feature, the team took the initiative to add an email system to notify customers of their purchases and demonstrated it live in their presentation. Figure 3 shows an iteration from this phase.



Billing Info

First Name Drew

Last Name Guarnera

Email drew@test.com

Phone 4405555555

Address 123 Test Lane

City Akron

Zip/Postal 12345

State/Province/Region OH

Country United States

Card Number: 12345678912345678 Master Card

Item Summary

Puma Nevis	$499.0	Qty: 3
Huffy Banana	$1075.45	Qty: 1

Sub-total: $2572.45

Sales Tax: $167.21

Your total is: $2739.66

Figure 3. Order information for RE-Cycle from phase 4.

The final phase of the project required porting the storefront to a local instance of the Google App Engine. This configuration setup was by far the most difficult among all the phases. Indeed, all of the teams encountered many errors and configuration problems before getting it to work with GAE.

Problems from the migration were just beginning. Once they ran the application, they received an error message saying that using an updatable record set for inserts, updates, or deletes with the GAE MySQL driver was not yet implemented. This meant going back and rewriting all of those methods in the database manager using the ugly SQL command statements they had tried to avoid. Another casualty of the move to GAE was the order email notification system. GAE would not allow the application to open the necessary socket network connections to send out the email, presumably for security reasons. Ultimately a lack of any good solution and time caused them to disable the functionality.

5. Outcomes

Part of the process of presenting their semester progress for each assigned project was feedback from other teams. The engineers led part of the discussions with questions and feedback. By the second project presentation, students started asking questions and giving advice based on their experience with different libraries and utilities that they explored and experimented with. It was very gratifying to watch the level of participation grow as the semester progressed. One of the objectives of the course sequence was to improve communication skills in a team environment, similar to what they might experience in the real workplace in a design or code review. This definitely started happening early in the process due to the participation and encouragement of the engineers.

Common feedback from the students was appreciation of the freedom given in the choice of projects and tools to enhance the experience. They covered a number of web development languages and tools, learned about the underlying technologies that support those languages and about web projects in general. We covered MVC, the RESTful web design pattern, basic concepts on web security risks, and data formats with their respective strengths and weaknesses. The format of the projects allowed students to take an open-ended approach toward accomplishing goals that they set as a team. Research, experimentation and mistakes are the best teacher. Mentoring by the Diebold engineers helped keep the students stay on track and receive insight into how projects are done in industry.

6. Acknowledgements

The authors would like to thank Diebold, Inc., Natalie Gainer, Jamie Meek, and Balaji Devarasetty for their support and for funding this project.

7. References

[1] The NIST Definition of Cloud Computing, Available: http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

[2] HTML and CSS tutorials, Available: http://www.w3schools.com/

[3] Semantics, Structure, and APIs of HTML Documents, Available: http://www.w3.org/TR/html5/dom.html

[4] The Java EE 6 Tutorial, Available:

http://docs.oracle.com/javaee/6/tutorial/doc/

[5] V. Chopra, S. Li, R. Jones, J. Eaves, and J. Bell, *Beginning Server Pages*, Wrox publisher, 2005.

[6] M. Fisher, J. Ellis, and J. Bruce, JDBC API Tutorial and Reference, 3rd ed., Prentice Hall, 2003.

[7] Introduction to XML, Available: http://www.w3schools.com/xml/xml_whatis.asp

[8] D. Crockford, JSON, Available: http://tools.ietf.org/html/rfc4627

[9] J. Webber, S. Parastatidis, and I. Robinson, *REST in Practice*, O'Reilly Media, 2010.

[10] D. Sanderson, Programming Google App Engine, O'Reilly Media, 2012.

[11] Google App Engine, Getting Started, Available: https://developers.google.com/appengine/docs/java/gettingstarted/

[12] M. Potel, "MVP: Model-View-Presenter. The Taligent programming model for C++ and Java." Taligent/IBM White Paper, Taligent, Inc., 1996, Available: http://www.wildcrest.com/Potel/Portfolio/mvp.pdf

[13] Sublime Text, Available: http://www.sublimetext.com/

[14] Eclipse IDE for Java Developers, Available: http://www.eclipse.org/downloads/moreinfo/java.php

[15] Google Gson, Available: https://code.google.com/p/google-gson/

[16] MySQL, Available: http://www.mysql.com/

A Minor in Computational Statistics

Xiaohui Zhong and Kevin Daimi
Department of Mathematics, Computer Science and Software Engineering
University of Detroit Mercy,
4001 McNichols Road, Detroit, MI 48221
{zhongk, daimikj}@udmercy.edu

Abstract— **Computational statistics methods enable statistical techniques, analysis, and interpretations through various computational techniques and processes. It helps in solving statistical problems both numerically and symbolically and automates various distributions and statistical procedures and inferences. The field of computational statistics is appealing to both statisticians and computer scientists. To contribute to the preparation of future computational statisticians, this paper proposes a minor in computational statistics. Curriculum design, objectives, outcomes, and assessment for this minor will be discussed in details.**

Index Terms— **Computational Statistics, Minor Design, Objectives, Outcomes, Assessment**

I. INTRODUCTION

Statistics is traced back to early history of human being. First census in human history was recorded in Numbers of the Bible [1]. The first two sample tests occurred in Daniel 1 in the Bible when it was proven that the children's appearances were as fair as those who ate the King's food. Statistics is the science of collection, organization, presentation, analysis, and interpretation of data [18, 26]. A statistician is a valuable asset in every industry at the design, data collection, analysis and conclusion, production, and post-production stages. The input of the statistician is essential for the success of every endeavor across science, engineering, industries, government agencies, and business [2, 4, 5, 6, 9, 12, 14, 17, 22, 23, 25, 26]. To summarize, there is no single field in which statistics is not applied in some way. Statistics is so important that even governments have branches named Bureau of Statistics or similar titles.

Recently, a growing challenge in the application of statistics is the data becoming extremely large. Traditional issues of data visualization and analysis are as challenging as computing is because the data can have complex structure with numerous sources of errors and variability. Big and massive data are everywhere. Such data need processing, analysis, and interpretation. This fact inspired the Obama administration to announce the Big Data Research and Development Initiative [13, 14, 15]. More important, the data are dynamic and constantly growing with information fed from internet. Data require modern computation techniques and database management skill to handle, process, and analyze.

Computation is an information technology based process relying on algorithms, heuristics, or models to obtain faster and accurate results (outputs). The harder the problem becomes the reliance on IT technology increases. Therefore, computation is closely related to computer science.

Computer scientists need statistics in many areas including data mining, data compression, speech recognition, vision and image analysis, network modeling, stochastic algorithms, artificial intelligence, machine learning, software quality, software reliability, and security. On the other hand, statisticians need computing skills to analyze and interpret large amount of data in a timely fashion with high accuracy before drawing conclusions. With large datasets, deriving various statistical models and carrying out various statistical analyses would be very time consuming, if not impossible, without the use of computers.

Computational Statistics is a branch of mathematical sciences concerned with efficient methods for obtaining numerical solutions to statistically formulated problems. Applied and computational mathematics and statistics use modeling, simulation, and data analysis to produce solutions to real-world problems [8, 22]. These methods are used in a wide variety of areas in business, engineering, natural sciences, medical science and social sciences [2, 9, 10, 12, 17, 25].

Computation plays a central role in modern statistics and machine learning [8, 11, 12, 13, 14, 25]. Statistical software is the key link between statistical methods and their application in practice [11, 22]. Software that establishes this link may be realized as a number of tools including tools for large-scale computation, database systems, distributed systems, embedded systems, Internet-based systems, and archiving and documentation.

Often, students or researchers trained in statistics or other fields requiring the use of statistics as tools do not have the necessary computing or data management skills [2, 4, 6, 12, 17]. On the other hand, there are demands and requests for developing links between computational thinking, statistics and other sciences [6, 11, 13, 14, 23], and for developing the skills for efficiently handing and analyzing very large sets of data [2, 12, 15, 25, 26]. *The Chronicle of Higher Education* reported that computational science is one of the top five up-and-coming majors [21]. Many government agencies and private sector also expressed interest in hiring students with statistics background [10]. Students who graduate with a traditional bachelor's degree in science, engineering, business or social science lack the skills required to solve modern-day science problems that are computing and data intensive. A minor in Computational Statistics offers students a solid preparation for careers in the fields they are trained at.

Hence, an attempt to establish a minor in computational statistics is necessary to allow students the opportunity of being exposed to the joint evolution of computational and statistical methods and techniques at the undergraduate education. To our best knowledge, no university offers a minor of such kind. The closest ones are either minors in statistics or minors in computational and data science.

II. DETAILED DESIGN OF THE MINOR

A. Objectives

The objectives of this minor are career and professional achievements oriented. This minor is uniquely situated within the department of Mathematics, Computer Science and Software Engineering. Thus, faculties in both disciplines can efficiently advise students, and the required courses are effectively arranged without conflict. The objectives of this minor include the following:

- Apply fundamental concepts, methodology and techniques of probability and statistics.
- Efficiently handle data collection, analysis and interpretation of the results.
- Apply modern computational methods for statistics, including methods for simulation, estimation and visualization of statistical data, to real world problems.
- Deploy computation as a tool for discovery in data analysis.
- Pursue professional practice of computational statistics in technical or further study on quantitative methods in business, economics and other related fields.

B. Outcomes of the Minor

Outcomes illustrate what graduates are anticipated to know or be able to do by the time of graduation. The outcomes anticipated in this minor are aligned with the recommendations from guidelines for statistics minor programs adapted by the American Statistics Association [3, 4, 24]. Upon completion of the computational statistics minor, students will be able to:

- Understand computational and statistical techniques and methods.
- Design and conduct computational statistics experiments, as well as to analyze and interpret data.
- Use the techniques, skills, and modern tools necessary for computational statistics.
- Identify, formulate, and solve computational statistics problems.
- Present their work orally and in writing to people in non-statistical fields.
- Collaborate with others on team projects.
- Recognize the need for engaging in life-long learning.

C. Course requirements and pre-requisites

The computer courses included in this minor are carefully selected to ensure the needed computational skills are provided. The Computational Statistics minor requires seven courses, three of which are statistics related courses, and the other four are computer science/software engineering courses. The proposed curriculum includes core courses in statistics, fundamental techniques in computing, as well as practical courses handling data with statistical software. While being consistent with guidelines recommended in [3, 4, 24], such combination of courses offer an optimal blend to students to experience the full benefits of this new emerging minor.

TABLE I. Minor Courses

Course #	Name	Credits	Prerequisite
CSSE2430	Data Structure	3	CSSE1720
MTH4270	Applied Probability and Statistics	3	MTH2410
CSSE4012 CSSE4013	Computational Statistics with (Lab)	3 1	CSSE2430 MTH4270
MTH4570	Regression Modeling	3	CSSE4012
MTH4370	Applied Time Series	3	CSSE4012
CSSE 4590	Computer Simulation and Modeling	3	CSSE4012
MTH 4690 Or CSSE 4950	Mathematics Seminar or Senior Design Project	3	

Table I lists the required courses in the order that students should follow to obtain their minor. The curriculum includes existing statistics areas within mathematics courses and computer science/software engineering courses. The concentration is on computing, data structures and statistics. Even though mathematical statistics courses are not required [3, 4, 24], all the mathematics courses require students to complete the calculus sequence (Calculus I, II, and III), which allow them to have basic mastery of mathematical concepts. Probability and statistics are introduced to students in MTH 4270 with exposures to applications on science and engineering. Further applications and advanced topics in statistics will be studied in the subsequent courses MTH 4570, Regression Analysis and MTH4370, Applied Time Series. The other components of the minor will cover conventional topics of data structure, and computational statistics. Prior to taking these computing courses, students are required to have basic programming skills, which are covered in CSSE 1710 and 1720 introduction to C++ programming I, and II. Industry experience in programming or coursework in a different programming language could satisfy this requirement. The computational statistics topics are evenly distributed among all five of the seven required courses, MTH4270, MTH4570, MTH4370, CSSE4012/ CSSE4013, and CSSE4590. The mathematics courses provide a broad exposure to data analysis skills powered by the techniques learned from the computer science /software engineering courses. The focus is on developing data analysis skill with modern computational tools to manipulate large sets of data. Through these courses, students will have the opportunity to apply their knowledge to projects involving large amount of data and earn the skills and knowledge that are often appreciated by modern workforce. Even though computational statistics is an interdisciplinary field applicable in almost all fields, this particular curriculum has been carefully chosen to emphasize the optimal combination of quantitative reasoning and computational competence. Graduates equipped with such skills will be highly valued by the industries, business, and any workplace where quantitative decision skills are needed. This curriculum will appeal to mathematics majors, computer science/software engineering majors, and other science/engineering majors because at least three of these courses overlap with the requirements of their majors or can be counted as the electives toward their majors. Students from other departments, especially in the electrical engineering and mechanical engineering who have taken basic programming courses, probability and statistics will find this minor very attractive.

D. Assessments

Assessment of the computational statistics minor employs standard practices in statistics and software engineering programs. The Minor assessment will be based on:

a. Homework, tests and exams

In-class exams, quizzes, homework, and projects will be used to measure student comprehension of the class material. Such assessment will be adopted in each individual course to ensure that the course is achieving its learning outcomes.

b. Comprehensive exam

A comprehensive exam, which covers all the topics of the minor, will be given to students during week 13 of their last semester. This exam is required for all students completing the minor.

c. Capstone project

In the capstone seminar/project, students will work on projects concentrating on statistical model building, computational manipulation and analysis of real world large datasets, or statistical simulation of real world problems. The performance in these projects will reflect how well students developed their skills throughout the minor curriculum. Students will also present their projects to other students and faculty as part of their communication skills improvement.

d. Surveys

An Exit Survey will be offered to students completing the program to solicit their feedback on the minor and on how to improve it.

An Alumni Survey will be used to discover how well our graduates feel they were prepared for their current position, where computational statistics is needed.

An Employer Survey will be prepared to obtain the feedback of employers on how well our graduates are prepared for their positions.

Samples of exams, projects, and other assignments will be collected over a period of time and over range of classes to evaluate the outcomes and syllabi content. The department assessment committee consisting of faculty members from mathematics and software engineering will process assessment data. It is the responsibility of the minor advisors to coordinate the input of all participating faculty to facilitate any modification of the minor.

III. COURSE DESCRIPTION

CSSE2430 - Data Structures

Review of object oriented principles, standard data structures, Big-O notation, NP completeness, stacks, queues, generic data types, dynamic memory, recursion, linked lists, circular lists, doubly linked lists, trees, binary search trees, heaps, graphics, sorting algorithms, searching algorithms, object-oriented language implementation of data structures and algorithms.

MTH4270-Applied Probability and Statistics

Introductory probability theory, elements of sampling and descriptive statistics, sampling distributions, estimations and hypothesis testing, regression, correlation analysis, and ANOVA.

CSSE4012 - Computational Statistics

Optimization, Combinatorial Optimization, EM Optimization Methods, Simulation and Monte Carlo Integration, Markov Chain Monte Carlo (MCMC), Advanced Topics in MCMC including Adaptive MCMC, Reversible Jump MCMC, and Auxiliary variable methods, Bootstrapping, Nonparametric Density Estimation, Bivariate Smoothing, Multivariate Smoothing.

CSSE-4013- Computational Statistics Lab (1 credit)

Lab session to accompany with CSSE4012 to introduce the tools used to implement computational statistics including SPSS, SAS, MATLAB, MAPLE, and R. It will be taken concurrently with CSSE4012.

MTH4570- Regression Modeling (3 credits)

A study of various regression modeling: multiple regressions, weighted least squares, logistic regression, log-linear models, analysis of variance, model diagnostics and selection. Emphasis on practical interaction of mathematics, basic statistics, and regress ion analysis is developed and applied to a real world setting using examples drawn from a variety of fields.

MTH4370- Applied Time Series

This course introduces the fundamental concepts, estimations, and hypothesis testing of discrete time series models. The models will be developed using the autoregressive and moving average processes through numerous examples in business and finance.

CSSE 4590 - Computer Simulation and Modeling

Basic simulation modeling such as in single-server queuing systems, inventory systems, parallel and distributed simulation, continuous and discrete simulation, and Monte Carlo simulation. Modeling of complex systems such as time-shared computer model, multi-teller bank with jockeying, differential equation systems (difference, ordinary, partial), and job-shop models. Discussion of simulation software, review of basic probability and statistics needed in simulation, selecting input probability distributions, and building valid simulation models. Simulation programs will be written and studied.

MTH 4690 – Mathematics seminar

Students will prepare and present paper(s) related to their primary area of interest. This course should be taken by mathematics majors in their final year of study.

CSSE 4950 – Senior Design Project

This is a team-oriented project course in which teams design, implement, test, and document a software system. The instructor will either offer or solicit a selection of projects. Projects may include students from other disciplines. Students should take this course in their final year of study.

IV. CURRICULUM DISCUSSION

The minor of computational statistics starts with a sequence of calculus as prerequisites, with basic programming, data structures, and algorithm courses being offered concurrently. Once students complete the calculus sequence, they are ready to take MTH4270 to gain skills in the fundamental methodology of probability and statistics. Students will learn all the necessary concepts such as parameter estimates, hypothesis testing, linear regressions, and ANOVA. While MTH4270 is prerequisite for the other two statistics courses required for the minor, it is a required course for mathematics, computer science and software engineering, and all engineering majors. CSSE 2430 is also a required course for all computer science and software engineering majors. Students should take CSSE2430 before taking other CSSE courses required for this minor. Hence, such curricular design minimizes students' time and money waste in case they choose not to pursue the minor further. It is recommended that students take CSSE4012/4013, Computational Statistics plus lab, prior to taking the other two mathematics courses so that students can utilize these tools immediately within the context of these math courses. The remaining two mathematics courses will deepen and broaden students' knowledge in the discipline of statistical methods. The work in time series will be rewarding for many application in the business and financial management area, while the regression modeling is extremely useful in data mining and engineering. CSSE-4012 is an essential course, which enables many statistical techniques and methods to be implemented using computational methods and techniques to achieve faster and accurate analysis and interpretation of data. CSSE 4590 is a typical application area course involving various statistical ideas and concepts to solve problems via simulation modeling software when deterministic approach is unavailable. The mathematics courses supported by the two CSSE courses, computational statistics and computer simulation and modeling, will empower the students with rich knowledge of statistical applications and solid computational skills.

All these courses will offer both supervised and independent team projects to promote teamwork as well as individual competence. Finally, the minor concludes with the capstone/seminar course listed as Mathematics Seminar or Senior Design Project. In this course, students will work on real world data to develop either models, or software tools, to solve some real world problems. Such practice offers students the freedom to be creative, apply their skills, and demonstrate their knowledge. Computational skills, which are often required to solve real-world problems, will be developed continuously throughout the curriculum. This course also enhances teamwork, programming skills, data analysis skills, and proficiency with professional tools [19]. Upon completion, students will present their work at the department colloquium, which is an open forum for mathematics, computer science, and software engineering faculty and students to exchange ideas, research progress, and new developments. Such opportunity for practicing communication and interaction is well emphasized in literature [2, 3, 10].

The main goal of the minor is to prepare students for career opportunities in industry and government, as well as the private sector in such fields as business, banking, insurance, health, and for further graduate studies. The program has a good balance of basic methodology, applications and data analysis, as well as carefully selected sequence of courses from computer science, and software engineering. This interdisciplinary approach is meant to make the program flexible, and give students a broad education to improve their chances of employment. Graduates of the program are well-qualified statisticians with good knowledge in computing, information systems, management and marketing.

V. FUTURE IMPROVEMENTS

The current curriculum puts an emphasis on computation, handling of large data, in addition to the essential methodology of statistics. The computational part of this minor distinguishes it from other minors in statistics. It covers fundamental mathematics, statistical methods, computer modeling, and data management. Even though these courses briefly cover all of the related topics, discipline specific courses for statistics elective would be beneficial for the students who want to work in this area [3, 4, 24]. For future minor improvements, experiment design, analysis of experiment, or quality control may be recommended to students from engineering majors. Numerical analysis could be suitable for students in the software development area. Courses in sampling theory or categorical data analysis would be appropriate for students in the social or behavior science. For biology, health science, or pre-medical science students, courses in survival analysis or multivariate analysis should be valuable. For economic and business students, a course in factor analysis or stochastic analysis will be recommended. Finally, for students in mathematics major, a course in mathematical statistics is strongly encouraged.

VI. CONCLUSION

The Computational Statistics Minor has a unique balance of courses both in statistics and computer science/software engineering. These courses include programming in C++ and Java, Data Structures, and Computational Statistics and Tools. It is housed in the department of Mathematics, computer science and software engineering. Even though a "statistics" course can be offered in different departments such as business, psychology or economy, only a department with faculties having both academic and industrial experience in statistics and computer science can offer such a minor with an effective curriculum. Such curriculum was designed to give high priority to intellectual exploration and attention to the process of learning in an interdisciplinary environment.

The collaborative minor in Computational Statistics is a project-led program. All courses are based on practical projects and lab assignments with real world data. It prepares students for employments with core skills in statistics and computing. This skills set is considered valuable and effective by the industries and many other marketplaces [10, 20]. It also provides students with the foundational statistics and computer science courses to prepare students for a Master degree in computational statistics, provided further requirements are met.

With such strong curriculum and marketable features, it is expected to attract students from many disciplines, who can apply the knowledge and skills to their own disciplines. This well-designed program will infuse the work force with "data specialists" equipped with greatly needed skills— computational statistics analytical skills. By doing so, we hope to contribute to "expand the workforce needed to develop and use the new technologies," one of the goals of the Big Data initiative [15]. It is also hoped that this minor can offer a model for other institutions, which do not have the full resources to offer a major in statistics.

REFERENCES

[1] The Bible, New International Version, Available: http://www.biblegateway.com/passage/?search=Numbers+1&version=NIV

[2] M. Boyce. "Statistics as Viewed by Biologists," *Journal of Agricultural, Biological, and Environmental Statistics*, Vol. 7, No. 3, pp. 306-312, 2002.

[3] G. Bryce, R. Gould, W. Notz, and R. Peck, "Curriculum Guidelines for Bachelor of Science Degrees in Statistical Science," *The American Statistician*, Vol. 55, No. 1, pp. 7-13, 2001.

[4] A. Cannon, B. Hartlaub, R. Lock, W. Notz, and M. Parker "Guidelines for Undergraduate Minors and Concentrations in Statistical Science," *Journal of Statistics Education*, Vol. 10, No. 2, 2002.

[5] K. Daimi, G. Grabowski, and K. Snyder, "A Collaborative Bioinformatics Minor Design," *in Proc. The 2009 International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS'09)*, Las Vegas, 2009, pp. 341–345.

[6] J. Friedman, "The Role of Statistics in the Data Revolution?" *International Statistical Review / Revue Internationale de Statistique*, Vol. 69, No. 1, pp. 5-10, 2001.

[7] W. Gilks, S. Richardson, and D. Spiegelhalter, *Markov Chain Monte Carlo in Practice*, Chapman and Hall, London, 1996.

[8] J. Givens, and J. Hoeting, *Computational Statistics,* Wiley, New Jersey, 2005.

[9] D. Hilderbrand, and T. Love, "Statistics Education and the Making Statistics More Effective in Schools of Business Conferences," *The American Statistician*, Vol. 56, No. 2, pp. 107- 120, 2002.

[10] R. Hogg, M. Ritter, and R. Starbuck, "Advice from Prospective Employers on Training BS Statisticians," A paper prepared as part of the Undergraduate Statistics Education Initiative of the American Statistical Association, 2000, Available: www.amstat.org/education/pdfs/usei_BIG.pdf.

[11] V. Kiernan, "Sophisticated Software Is Reshaping the Way Scientist Use Statistics," *The Chronicle of Higher Education-Technology*-, January 8, 1999.

[12] Lambert, D. "What Use Is Statistics for Massive Data?" *Lecture Notes-Monograph Series- Crossing Boundaries: Statistical Essays in Honor of Jack Hall*-, Vol. 43, pp. 217-228, 2004.

[13] K. Lange, *Numerical Analysis for Statisticians*, Statistics and Computing Series, Springer-Verlag, New York, 2010.

[14] N. Lazar, "Multiplicity Control in Large Data Sets Presents New Challenges and Opportunities," *Chance*, Vol. 25, No. 2, pp. 37-40, 2012.

[15] N. Lazar, "Big Data Hits the Big Time," *Chance*, Vol. 25, No. 3, pp. 47-49, 2012.

[16] N. Lazar, “Data. Data, Everywhere …,” *Chance*, Vol. 26, No. 1, pp. 33-37, 2013.

[17] K. Mangan, “The Need for Statistical training in Medicine,” The *Chronicle of Higher Education-Archives-*, September 25, 2006. Available: http://chronicle.com/article/The-need-for-statistical/118043/.

[18] V. Mantzapolis, and X. Zhong, *Probability and Statistics*, Kendall Hunt Publishing Company, Dubuque, 2010.

[19] S. Morozov, K. Daimi, and Y. Wu, “A Minor in Game Development,” *in Proc. The 2012 International Conference on Frontiers in Education*: *Computer Science and Computer Engineering (FECS’12)*, Las Vegas, 2012, pp. 22-27.

[20] M. O’Fallen, “Undergraduate Education and Communication,”*Amsta News*, No. 277, pp. 2-3, July 2000.

[21] R. Read, “America’s Top Job: Software Engineer,” *The Chronicle of Higher Education*, April 23, 2006, Available: http://chronicle.com/blogs/wiredcampus/americas-top-job-software-engineer/2156.

[22] C. Rose, “Computational Statistics”, in *International Encyclopedia of Statistical Science*, M. Lovric, Ed., Springer-Verlag, 2011, Available: www.tri.org.au/Rose_Computational_Statistics.pdf.

[23] J. Spencer, and P. Tobias, “Statistics in the Semiconductor Industry: A Competitive Necessity,” *The American Statistician*, Vol. 49, No. 3, pp. 245-249, 1995.

[24] T. Tarpey, C. Acuna, G. Cobb, and R. De Veaux, “Curriculum Guidelines for Bachelor of Arts Degrees in Statistical Science,” Journal of Statistics Education, Vol. 10, 2, 2002. Available: http://www.amstat.org/education/pdfs/BA-curriculum.pdf

[25] J. Wing, “Computational Thinking and Thinking about Computing,” *Philosophical Transactions: Mathematical, Physical and Engineering Sciences*, Vol. 366, No. 1881, pp. 3717-3725, 2008.

[26] X. Zhong, and K. Daimi, “Alternative Two Sample Tests in Bioinformatics,” *in Proc. The 2012 International Conference on Bioinformatics and Computational Biology (BIOCOMP’12)*, Las Vegas, pp. 345-352, 2012.

A Writing Intensive, Oral Presentation, Ethics Course in Computer Science

Laurie J. Patterson
Department of Computer Science, University of North Carolina Wilmington, Wilmington NC, USA

Abstract - *This paper covers how one institution addresses the need to prepare computer science students to write well, present well, and understand the ethical issues associated with technology, all the same time.*

Keywords: Curriculum, ethics, writing intensive.

1 Introduction

Many colleges and academic institutions expect students to master writing in their curriculum. These courses may be a requirement for their broad-based basic studies, or for courses mandated by accrediting organizations (such as the Southern Accrediting for Colleges and Schools, SACS) or for the department (such as ABET).

This paper discusses one version of a writing intensive/ ethics course at the University of North Carolina Wilmington (UNCW).

2 Historical Perspective

UNCW recently underwent a change in their basic studies or general education (genEd) courses. Students were initially required to take 39 credit hours of courses in areas such as English, Physical Education, Sociology, Life and Physical Sciences, and Mathematics. Two course, that were required, were composition courses; both of which were offered through the Department of English.

Currently, however, the genEd plan extends the types and focus of courses across all departments, attempting to build on relationships and themes between topics and departments. One such area is writing. A third writing course is now required, one of the three courses must be taken within the student's major [4]. It is believed that a student learns to write for their profession within the profession. Two of the university's stated goals of their genEd program specifically address writing:

5. Goal 5. Effectively express meaningful ideas in speech and writing (Thoughtful Expression);

6. Goal 6. Demonstrate basic proficiency in speaking, listening, writing and reading in a language in addition to English (Foreign Language);

Table 1 provides a listing of the UNCW components of a writing intensive course.

The writing intensive requirement coincided with the recommendations of the ACM Computing Curricula 2001 Computer Science (CC2001) [1]. The CC2001 Task Force used guidelines in the development of the curriculum:

10. "CC2001 must include professional practice as an integral component of the undergraduate curriculum. These practices encompass a wide range of activites [sic] including management, ethics and values, written and oral communication, working as part of a team, and remaining current in a rapidly changing discipline. We further endorse the position articulated in the CC1991 report that "mastery of the discipline includes not only an understanding of basic subject matter, but also an understanding of the applicability of the concepts to real-world problems" [1].

11. The CC2002 Task Force also confirms UNCW's position that writing for the profession occurs within the profession. Chapter 9, Completing the Curriculum, contains many subsections of curricula recommendations. Subsection 9.1.4, Communications skills, focuses on the computer scientist's ability to "communicate effectively" and the students need to "sharpen their oral and writing skills" [1].

To meet both UNCW's genEd requirement and the ACM recommendation, the UNCW Department of Computer Science modified their ethics course to include both the writing intensive requirement and the oral communication requirement of all UNCW students (see Goal 5, above). The course description for the writing intensive course reads as follows:

> "Ethical and professional issues arising from the impact of computer science and related technologies on society. Topics include ethical issues, obligations of professional practice, privacy and security, intellectual property, work and health issues, and the impact of emerging technologies. Students give both oral and written presentations and participate in the discussion of case studies."

TABLE I
COMMON STUDENT LEARNING OUTCOMES FOR WRITING INTENSIVE COURSES

The student will:

WI 1.	Locate appropriate sources of information to support written arguments. [Foundational Knowledge]
WI 2.	Evaluate and use evidence to generalize, explain, and interpret content. [Information Literacy; Critical Thinking]
WI 3.	Demonstrate an understanding of the ethical use and citation of the ideas of others used as supporting material in written work. [Inquiry; Information Literacy; Thoughtful Expression]
WI 4.	Demonstrate the ability to write critically, using the conventions of the discipline covered in the course. [Critical Thinking; Thoughtful Expression]
WI 5.	Analyze and evaluate the claims, arguments, and theories presented course materials using appropriate methods (such as logical analysis and the identification of fallacies). [Foundational Knowledge; Information Literacy; Critical Thinking; Thoughtful Expression].

3 Current Course

This course has typically been taught in both fall and spring semesters. One instructor teaches the course in the fall; and another teaches it during the spring. This paper focuses on the approach by the spring instructor.

Course Learning Outcomes (CLOs) are consistent across both semesters as is the general breakdown of assignment weights for the final grade. CLOs for the course are listed below. Values in brackets indicate which component of UNCW's writing intensive requirements the CLO aligns with:

1) Identify ethical issues as they impact computer science and related disciplines; [WI 2]

2) Differentiate between the main ethical theories and be able to use the ethical theories in evaluating the ethical issues impacting computer science and related disciplines. [WI 5] [WI 2]

3) Discuss ethical issues in writing, using appropriate reference to the established Code of Ethics of the professional society relevant to that student's field (ACM, IEEE, etc.), and apply professional codes of ethics to analyze and resolve ethical questions. [WI 5] [WI 2]

4) Demonstrate the ability to write within the computer science discipline including writing one or more research papers that demonstrate the students grasp of ethical issues, display a clear understanding of how the ideas of other persons may be properly cited and used in written documents, and illustrate use of popular formats for presenting published papers in computer science. [WI 3] [WI4]

5) Prepare and present information on a technical topic, in a professional manner.

6) Identify and locate appropriate sources of information to support decisions and written ideas. [WI 1]

7) Analyze and evaluate arguments using rules of logic and be able to formulate effective arguments based on sound premises. [WI 2] [WI 4] [WI 5]

TABLE II
MAPPING STUDENT LEARNING OUTCOMES TO COURSE CONTENT

Course Learning Objectives/ Outcomes	Research Project Written	Small Written Assign	Research Project Present	Ethics Topic Present	Resume LinkedIn Interview	Midterm	Final
	1	2	3	4	7	8	9
1	◆	◆	◆	◆	◆	◆	◆
2	◆	◆				◆	◆
3	◆	◆					
4	◆	◆			◆	◆	◆
5		◆	◆	◆	◆	◆	◆
6	◆	◆					
7	◆	◆				◆	◆

4 Course Format

Following is a brief description of each of the grading sections.

4.1 Small Writing Assignments

The current offering of the writing intensive course includes four small writing assignments. Each of the assignments allows for a second submission; however, only the first two assignments average the score between the two grades. The second two assignments, students can accept the grade they have received, or submit it a second time for a higher grade.

Topics for the four assignments are:

1) One page summary of a chapter from the textbook. Rewrite required for a second submission.

2) One page essay on the effect of technology on the student's life. Rewrite required for a second submission.

3) Three-page paper on a technology-based scenario. Rewrite for a better grade accepted.

4) Position paper response to an ethical question posed by the textbook. Rewrite for a better grade accepted.

TABLE III
GRADING WEIGHTS FOR ASSIGNMENTS

COURSE ITEM	GRADE PERCENT
Small writing assignments (5 or 6)	20%
Presentation of an Ethical Issue	10%

Resume & Interview & LinkedIn	10%
Formal Written Research Paper	20%
Formal Research Paper Presentation	10%
Midterm Examination	10%
Final Examination	10%
Class Participation	10%

4.2 Presentation of an Ethical Issue

This course is also designated as fulfilling the requirement that UNCW students must be able to present their research and/or themselves in a public forum. The ability to make an oral presentation is also part of the ACM curricula recommendations:

> "The program of each computer science student must include numerous occasions for improving writing and practicing oral communication in a way that emphasizes both speaking and active listening skills.
>
> At a minimum, a computer science curriculum should require:
>
> - Course work that emphasizes the mechanics and process of writing
> - At least one formal oral presentation to a group
> - The opportunity to critique at least one oral presentation" [1].

Students are provided with an technology ethical issue. They create a PowerPoint presentation and make a ten-minute presentation on the topic. The presentation is followed by five minutes of question and answer.

Students are scored on six areas:

1) Organization
2) Subject Knowledge
3) Graphics
4) Mechnics
5) Eye Contact
6) Elocution

All students are expected to attend the presentations and to score the presenter. Their grades are averaged and then provide 50% of the grade for this assignment. The instructor provides the remaining 50%. Once the scoring has occurred, the instructor returns the assessment sheets (with names removed) to the presenter. Students are told to note the suggestions relating to their ethics presentation when preparing for their research presentation.

This course is also designated as fulfilling the requirement that UNCW students must be able to present their research and/or themselves in a public forum. The ability to make an oral presentation is also part of the ACM curricula recommendations.

4.3 Resume & Interview & LinkedIn

Subsection 9.1.6, The complementary curriculum, of the CC2001 in part addresses the need for inclusion of this portion of the course. It is recommended that the computer science curriculum "encourage" students to develop other skills:

> "… skills … such as the ability to write an effective résumé" [1].

Students work with a member of the university's career center to write a professional résumé. The students then attend at least one career fair and present their resumes to prospective employers.

In preparation for the next step in their communication with employers is to participate in a mock interview. UNCW utilizes InterviewStream to conduct these interviews. The instructor for the class creates a set of questions that are then presented via a computer. The computer has a video camera and microphone that captures the video and audio responses of the students. Students get multiple chances to use Interview-Stream. The resulting video is available for the students to view as well as the instructor.

Preparing students to present themselves into the professional world is also a concern for ACM. The CC2001 states that the professional sectors in which a student would work benefits from "students who have experience with the realities of professional work" and students "need to understand the importance of professional conduct on the job" [1].

An additional step in the UNCW course, for students to be able to connect to the work sector and see what professionalism in computer science means is accomplished through the third portion of this part of the class: LinkedIn. Again, working with the university's career services center, students receive professional "headshots" for use in the LinkedIn profile. A discussion of why a professional picture is needed is held and students have the opportunity to see several that are not "professional." A discussion follows on how a company might view this prospective employee. A follow-up to the picture discussion occurs in the midterm (see the section on "Midterm Exam."

Additional discussion occurs about "connecting" to prospective companies that an individual might wish to work at. As they connect to companies, organizations, and professionals in the field, the students learn not to use the standard “I'd like to add you to my professional network on LinkedIn” [3] on their invitation to connect. A discussion is held on how to grab/get/hold a prospective employer’s or colleague’s attention through mentioning a previous meeting or connection.

Until a student submits a request with a professional headshot and a professionally framed request to the instructor, a grade is not assigned.

4.4 Formal Written Research Paper

During past offerings of the course, the standard “find-a-research-topic-and-write-on-it” paper was assigned. Students would then write a paper that may not have actually been “research” based and more in line with what may have been expected in their beginning writing courses.

In the most recent offering of the course, the instructor modeled nine grant call for proposals (CFP) based on several actual NSF grant proposals. Students were instructed to think of a problem within their field and to brainstorm possible solutions to that problem. Most options provided to the students dealt primarily with Science, Mathematics, Engineering, and Technology (STEM) topics since the students, themselves, had experience in education in those topics. A few other options allowed them to utilize classroom experience in databases and networking. Topics included:

1) Undergraduate Education in STEM

2) Computational and Data-Enabled Science and Engineering

3) Ethics Education in Science and Engineering

4) Science Across Virtual Institutes

5) STEM Talent Expansion Program

6) Innovative Technology for Students and Teachers

7) Advanced Technological Education

8) Science, Technology, and Society

9) Networking Technology and Systems

The total assignment for the pseudo-grant writing was divided into five segments that covered the eight items required in a grant proposal writing as defined by Coley and Scheinberg. Coley and Scheinberg’s eight items are:

1) Cover letter, title page, and abstract. A generic template was designed for the students to start putting their grant components in to it.

2) A needs or problem statement.

3) The description of the project.

4) A plan for how their success (or failure) of their project will be evaluated.

5) A budget for equipment, software, and personnel.

6) Their strengths and capabilities. This step included a copy of their finished resume as well as paragraphs describing what their strengths were and how they would benefit their project.

7) Whether or not they will need future funding.

8) A letter of support from someone describing the strengths of their project. For most students, they contacted one of their former professors to get additional help on how to fine-tune their project. This faculty member then wrote a letter of support [2].

These eight items were spread out over five separate assignments to reduce the daunting nature of writing a grant. The first assignment included the cover letter and title page from item one and the student’s capabilities from item six. Assignment two included the needs statement from item two and their letter of support from item eight. The third assignment was the project description from item three. The abstract from item one and the evaluation plan from item four constituted the fourth assignment. The last segment included the budget from item five and future funding plans from item seven.

As students write each section, they get feedback and a second chance to rewrite that assignment for a better grade. The initial submissions are not specifically formatted; however, the sixth and final submission includes all the previous sections into the instructor created template.

By the final submission point, students have essentially written a grant proposal with each section potentially checked for professional writing at least one time.

4.5 Formal Research Paper Presentation

In preparation for the final research presentation, students are subjected to two additional oral presentations. These presentations are “lightning talks.” The presentations are scheduled for 10 minutes with 20 slides (including one with the title of the presentation and the name of the presenter). Each slide automatically advances to the next slide after 31 seconds. The presenter must occupy the entire amount of time the slides is presented.

Topics are selected by students, combined, and then picked by a student. The student has 5 days to prepare the presentation and submit it to the instructor. The instructor randomly puts the presentation together. Students are only notified it is their turn to present when their name appears on the screen. The instructor makes the first presentation. For the second presentation, the same PowerPoints are used, but a name is randomly drawn at the time of the presentation.

The purpose of using the lightning talks is to take the pressure of making a presentation off the student. The students are not scored by anyone, including the instructor. The object is to focus on professionalism and thinking "on one's feet."

The final research presentation occurs during the student's final exam time. Exam times at UNCW occupy a three hour time slot and those hours are considered part of the required teaching hours of the course.

Students will present their research project in a setting that is similar to both the lightning talk and the ethics presentation. Presenters will be scored by both the instructor and the other students in the same manner as the ethics presentation (see IV.B), but will be severely limited in time.

4.6 Midterm Exam

For the midterm exam, students are given a "take-home" exam that is due 48 hours after it is handed to the students (thus ensuring receipt of the exam). A scenario is presented to the student that has a contemporary ethical twist to it. Students are then required to present a series of premises and the conclusion to which the premises lead. The remaining two pages are then dedicated to online research and legal issues which relate to the ethical dilemma presented.

The most recent exam presented a scenario in which a company was considering hiring a woman, but through data-mining of images on the Internet found a picture of her taken at a protest of their company. There was no additional evidence to support the company's claim that she might be a subversive. Students were asked to take a position and support their position with research to back up their premises and conclusions.

4.7 Final Exam

The final paper is, again, a position-based writing assignment. The focus on this four-page paper is to find a technology-based ethical issue that they feel passionately about…such as: broadening participation, intellectual property (either protecting or not), or the digital divide.

To assist in developing their position on the topic, students are required to:

1) Page1:
Describe in a paragraph the issue that they feel strongly about. The topic must be a different topic than their final paper. Provide premises and a conclusion to support their statement.

2) Page2:
In the top half, provide premises and a conclusion to support the opposite position.

3) Pages2-3:
For the remaining 1.5 pages, students must provide support for the opposite argument.

4) Page4:
The last page is dedicated to references. Students are required to use the reference format from the IEEE template.)

4.8 Class Participation

For the initial portion of the class, when it is only the instructor or the representative from UNCW's career center presenting, students must sign-in that they have attended the course. Once student presentations begin, it is the scoring sheet that each student submits that serves as the notification of attendance.

The number of attendances are tallied and divided by the number of times the course met. This result serves as class participation grade.

5 Findings and Recommendations

Because of the upswing in computer science enrollments, this course is frequently full. Semester control sizes are capped at 30, which does not cover the demand for the course. The control size, however, means that a great deal of writing must be evaluated by the instructor (in addition to any other courses that must be graded as well). Because of the grading pressure, each assignment has a recommended due date. Students are aware that each assignment must be done, but they are allowed to submit when they are ready. This removes some immediate grading pressure from the instructor.

The amount of grading, however, remains daunting. Because of this, the instructor has worked with other departments on campus to utilize their graduate students. Graduate assistantships are expected to be provided to students in English writing and technical writing to assist (and be a resource to the students) in providing feedback to the students and perform some of the grading. Without this assistance, providing this level of writing support to the growing number of computer science students would be impossible. With the increased demand on assessment by administrators, the level of support provided to the students must continue.

6 References

[1] Association for Computing Machinery. (December 15, 2001). Computing Curricula 2001 Computer Science, Final Report. [Online]. Available: http://www.acm.org/education/curric_vols/cc2001.pdf

S. M. Coley and C.A. Scheinberg, Proposal Writing. Newbury Park, CA: SAGE Publications, Inc., 1990.

[2] LinkedIn.com. (2013). [Online]. Available: http://www.linkedin.com//

[3] University of North Carolina Wilmington. (2012). University Studies: Writing Intensive Course. [Online]. Available: http://uncw.edu/universitystudies/writing intensive.html

[4] Wikipedia. (February 26, 2013). “Lightning talk.” [Online]. Available: http://en.wikipedia.org/wiki/Lightning_talk

Web-enhanced design of university curricula

Elisabeth Bacon
School of Computing and Mathematical
University of Greenwich
Greenwich, London, SE10 9LS, UK

Georg Hagel
Faculty of Computer Science
Kempten University, Bahnhofstraße 61
87435 Kempten, Germany

Michael Charnine
Institute of Informatics Problems
119333, Moscow, Vavilova St., 44/2, Russia

Richard Foggie
ESP Central, 4-10 Barttelot Road,
Horsham, RH12 1DQ, UK

Brian R Kirk
Robinson Associates
Weavers House, Friday Street
Painswick, GL6 6QJ, UK

Georgy Kravtsov
Thunder Arts Ltd46 Graveney Road
London, UK, SW17 0EH

Igor Schagaev
FLSC, London Metropolitan University
166-220 Holloway Road, N7 8DB, UK
i.schagaev@londonmet.ac.uk

Abstract — Aspects of curriculum formation, supportive framework development and a scheme of delivery are introduced, using a web-semantically driven approach accompanied by a model of human perception, information delivery and assessment supportive schemes. Shown how semantic web research can help with monitoring of knowledge base, defining structural knowledge dependencies, detecting the change of knowledge state and estimating functional or structural deviations. The impact of Information and Communication Technology (ICT) and web-media on human perception and knowledge formation is analyzed to avoid use that might impair human development through brain/sensory/perception models. The visual, textual and audio channels of information delivery are analyzed together. Shown how web-semantic approach can be supported by modification of assessment process with better tuning to knowledge formation and student ability making assessment frameworks based on automated support for reliable and quality of knowledge delivery. Structure of further research is suggested.

Keywords - curriculum; web-semantic search; human perception; signal processing; assessement methods and tools

I. Introduction: Concepts & Objectives

The high education curriculum has not utilized the speed and intensity of the technological and scientific revolution boosted by ICT. At the same time semantic Web research might help in an upgrade of the curriculum, making possible fully automated monitoring of the knowledge base. The next generation of curriculum development requires:

- Revision of the process of curriculum design including as information processing scheme; use of semantic web searches based on keywords combined with the current state of knowledge define a core; modification of teaching schemes; incorporation of system support for maintenance of core of knowledge using both: historical and current information, checking the structural knowledge dependencies and state transitions in a single model.
- Study of the impact of application of ICT on the human creativity, differencing two dependent models:
  a) Web/ICT based systems of knowledge delivery and consumption
  b) information delivery to brain in a practical and effective way.

Bullet b), in turn, requires an understanding of the structure of the brain in terms of information delivery and perception. There are three main channels of knowledge delivery: visual (imagery), textual and auditory; all must be considered together.

Load of new curriculum development and delivery should be estimated using performance/workload monitoring model and delivery control as form and cost of workload change for lecturers and students. Load should be monitored - we need to know how much innovation we can carry / cope without killing ourselves.

Clear, new web-based curriculum development should be supported by an automated process for certification of the results of education, the quality and quantity of knowledge and skills, assisted using web-based semi-automatic assessment scheme. This process needs to be 'self-tuning' on the level of student knowledge as well as to provide guaranteed coverage

This work is partially supported by the Russian Foundation for Fundamental Research, Grant 13-07-00272

of the required scope of knowledge and significantly reducing cost. All these points are inter-related and require co-design and co-development, i.e. considered as one approach.

II. INTEGRATED APPROACH

This integration of theoretical formation of curriculum, a framework for development, a research of a process of delivery, models of the workloads and automatic supportive assessment scheme can be integrated if all these processes will be considered as an information processing system or model, including the human as a part of this process. A knowledge delivery through curriculum should consider properties of human brain: this helps to exercise an adjustment of speed of information delivery to student abilities. Figure 1 below presents a bird-view on our approach followed by further explanation.

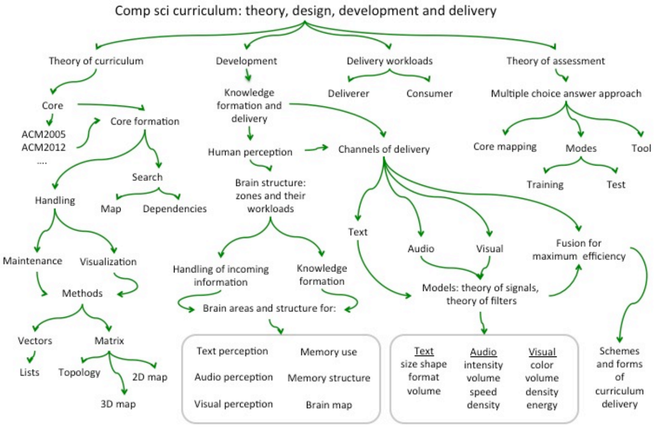


Figure 1. WEDUCA approach bird view

III. WEB SEMANTIC FOR THEORY OF CURRICULUM

Making curriculum enhancement is possible using a web-semantic approach considering a knowledge delivery system by analogy with [1] and [2]. That System might be analyzed through closely connected functions: Definitive, Characteristic and Predictive. A Definitive Function (DF), nominates terms, DF answers the question “What is it”? The second function describes the interrelations between Definitions and thus characterizes them. It is called the Characteristic Function (CF). CF answers the question “how are these definitions connected”? The Predictive Function (PF), in turn, answers the question “What if”? DF and CF and their application are essential elements of knowledge. Application of definitions and characterization of their interrelations enables predict behavior of a learned element.

The evolution of a subject area can be described through a joint Glossary {DFj} of terms, as well as an essential Glossary {DFe}; through their change we can observe features such as merging, moving and separation of a curriculum core. A moving core occurs when curriculum shows growth when using some descriptive elements (DFi) and decrease in the use of others. This process of core motion can lead to the separation of a discipline into several new disciplines. This separation should be detected early before it becomes significant, as the discipline can loose its predictive function (PF) and thus become obsolete.

The other process of changing the core is merging of several cores of various disciplines. This process can be caused by significant discoveries and inventions, practical demand or technological revolution or an accelerating fusion of several disciplines where the sum is greater than the parts.

Fusion becomes visible by the growth of the Predictive Function - resulting in the applicability of a new discipline to society.

In order to make continuing process and establish theoretically useful concepts one has to develop a sort of engine to make the motion of knowledge visible – a knowledge seismometer that warns that the curriculum needs to be adapted to the motion of the knowledge landscape that is happening.

```
-A set of initial terms/keywords is given defined manually,
  taken from the curriculum.
-Collect text from the Internet with predefined keywords
  (formation of body of texts for various time slots).
-Design context vectors that determine meaningful closeness
  and dependency between keywords at various moments of time
  and time slots (point and evolution dependency).
-Describe keywords on a 2-dimensional chart.
-Searching clusters of closely related keywords.
-Searching of the hierarchical structure of cluster
  dependency (hierarchical clustering).
-Develop a closed curve that includes clusters of keywords
  (this is a landscape version and change of form and size
  of this curve is landscape motion).
-Those new terms that appeared inside this closed curve and
  thus became closer to the central keywords are candidate
  terms for inclusion in the curriculum.
-Those keywords that are more distant from central keywords
  are candidate terms for exclusion from the curriculum.
  After deletion these candidate terms the curve changes its
  form and thus motion of the curriculum is defined.
```

Figure 2. Algorithm for detection of landscape motion

When structures of clusters become virtually disconnected clusters are too remote then it indicates a process of core separation - formation of a core emerge for new discipline. Monitoring of evolution of core is possible algorithmically. Testing and initialization of proposed algorithm can be approached using as a starting point three ACM curriculums for computer science of 2001, 2005 and 2012 respectively:

```
-Define keywords from curriculum 2001, search new candidate
  terms using the proposed algorithm.
-Define keywords from curriculum 2005, compare with
  previous results, search new candidate terms.
-Define keywords from curriculum 2012, compare with
  previous results and define new terms and topics to be
  included into curriculum.
```

The more new candidate terms are included into new curriculum the more visible core motion and landscape change we have – indicating changes required into next generation of curriculum. As a trivial example of checking of conceptual and semantic links between terms and core migration in computer science area using the web using Multidimensional scaling (performed by a tool R script) one can see initial mapping of cluster of knowledge, Figure 3. Checking of moving for term "friend" might be helpful for formation of landscape change as a whole.

A. Visualization of dynamics

The Web-enhanced Design of University Curricula (WEDUCA) approach represents the curriculum in the form of several clusters of terms that dynamically change over time in context, quantity and even meaning. In order to define, analyze and "identify" clusters there is a need to understand the measures of clustering between entries and terms thus measuring semantic connectivity and distance, usually called the "semantic similarity of terms". The most modern approach to calculate this is based on the use of distributional semantic models [20], successfully used for word similarity, word clustering, automated thesaurus generation, word sense disambiguation, and query expansion. For WEDUCA purposes the first application of distributional semantic models comes from analysis and dynamic monitoring of curriculum evolution.

Distributional semantic models, variously known as vector spaces, semantic spaces, word spaces, corpus-based semantic models, all rely on some version of the distributional hypothesis, stating that semantic similarity between two words/ terms can be modeled as a function of the degree of overlap among their linguistic contexts. Having web astronomical and constantly growing amount of textual material we can provide the basis for identifying linguistic contexts and semantic dependency as well as differences (of terms) evolving over time. This will enable the observation of curriculum migration with growing precision.

A subject map (or landscape) is a kind of visual representation of the important subjects and terms in the target domain and their mutual conceptual relationships and connectivity, on a 2-dimensional chart. The reduction of dimension and scale of the original semantic space allows the representation of the distances among terms on a subject map.

Semantic Vector Space enables visualization of super-scale volume of data, simply because distances between terms on the subject map depend on distribution of context vectors, which components depend on distribution of large amount of invisible on the map (landscape) word combinations. This way a relatively small number of visible terms on the map might depend on distribution within knowledge domain of millions of invisible word combinations.

This makes possible to create a compact map of a subject, aggregating information of 'super-scale' body of texts. Introduction of new information into the body of texts might substantially change distances and dependencies as a whole between different terms on the map, making visible the impact of new information. This also makes possible to detect and process an early identification of missed terms/keywords in the curriculum.

Thus maps of subject area, based on Semantic Vector Space model becomes essential and unavoidable instruments for monitoring of distances and differences of super-scale and large-scale bodies of texts and data.

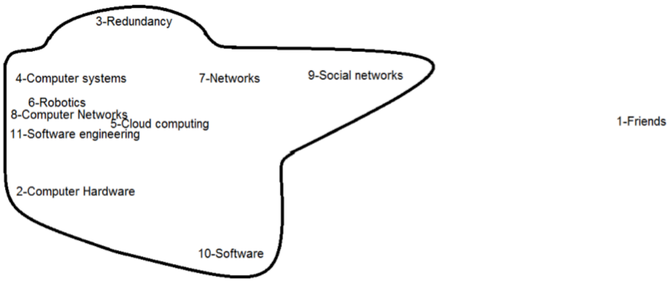


Figure 3. Clustering of terms within enveloping curve

Checking of moving for term "Friends" on Fig.3 might be helpful for observation of landscape change as a whole. Space position of terms on the map defines their semantic connection. Terms from Computer science area are inside enveloping curve that evolves permanently, together with modifications of dependencies between terms. External terms in relation to enveloping curve also depend on semantic connectivity. Thus the external term "Friends" is connected with the new cluster "Social networks". The term "Social networks" is relatively recent and therefore resides at the border of computer science area - enveloping curve. This kind of dependencies are visually detectable and can be analyzed using 2-dimensional chart. We can consider it as simplest test of work for our model.

IV. Knowledge delivery aspects

The development of two areas of knowledge formation and delivery: human perception and a theoretical analysis of ways to deliver knowledge efficiently should be included into design of curriculum. A problem of searching the core of knowledge has a long history. Surprisingly, we still follow a lot of ancient scientists, their results & methods, hardly developing our own and new. To quote Jonathan Swift [3]:

"*ancient virtuosos were carrying a spirit in casks and barrels which was out of great assistance upon long life voyages*."

In the past people had time to think, to reflect, time to react. They had time to use their associative memory and brain. But this gentle approach happens less and less. We are now swamped by information the Internet, like children in a sweetshop. To cope with this information avalanche we have to introduce semi-automated ways to identify, track and adapt knowledge drifting. In other words use the web-semantics to extract knowledge from web and manage it is crucial. ICT driven gargantuan information noise thrown at us at an enormous rate and volume using for this all possible delivery channels and is overloading our brains - "*aliqno modo essentiae adhaerese*" (in some way affect the substance). An ever increasing range of technologies are delivering the information deluge and are unwittingly involved in degrading our ability to filter and follow the knowledge we desire from it. To counter balance workload on the brain efficiently we should consider both: brain organization and theories of control and signals.

A. Human perception aspect

Each student brain is capable of absorbing substantial amount of knowledge. The knowledge delivery process can use any combination of channels of delivery: via text, via audio or visual. 'Text' should be separated from 'visual' and considered as special channel. All three processes of information absorption consume brain energy; all have a different level of information density and all need to be effectively used in a concert to achieve the best results. As an example, let's look at the visual channel in a bit more detail and understand where this input is processed and how much energy the visual path of information delivery requires. The visual cortex deals with visual image sensing, while recognition and interpretation of "what we see" uses other segments of brain. So, to an extent our abilities to think while we concentrate on the screen are actually reduced or even blocked. This is only part of the problem, the perception of textual, audio and possibly other sensory information also may be needed to contribute to the recognition of knowledge but all must be correlated together.

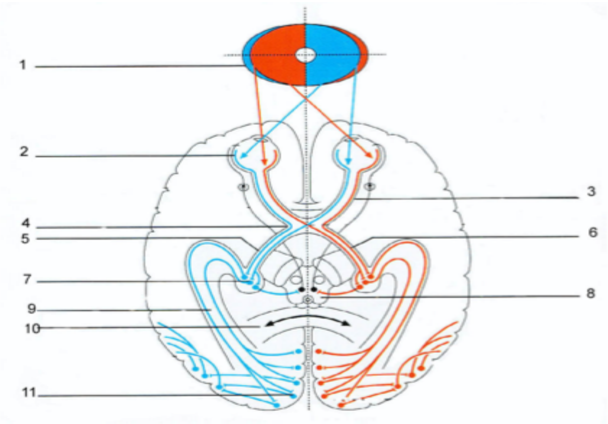


Figure 4. Visual recognition within of human brain

1 – Areas of vision; 2 – Eye perception (retina); 3 – optic nerve (nervus opticus); 4 – optic chiasma – (chiasma opticum); 5 – Left visual tract (tractus opticus sinistra); 6 – Right visual tract (tractus opticus dextrum); 7 – lateral geniculate body of thalamus (corpus genicula tumlaterale thalami optici) – primary processing of information; 8 – Four-hill area (corpora quadri gemina) – internal regulation of visual sensors; 9 – optic radiations (radiation optica); 10 – commissura fornicis (corpus callosum, commissura anterior) – connecting links; 11 - visual cortex of occipital lobe

Main function of the visual system is the transformation of energy produced by direct or reflected light. Photosensitive cells absorb light and cause a nerve signals to transfer information to the brain. These photoreceptors (cones and rods) transform light by photochemical reaction into nerve impulses. The variations in intensity in time and space provide the information that the brain interprets to create its changing images of the environment being viewed. Figure 4 illustrates transfer of primary visual information in more detail.

The visual cortex includes optic elements (retina), the optic nerve and optic tracts that transfer signals within the brain. The visual image is captured by retina as an inverted image (we see ground and sky upside-down). When signals go through internal areas and visual cortex, the signals are transformed into an image that we interpret as being right-side-up. Left and right eye optic nerves (n.opticus) go through the whole brain, form a cross and through visual area of the thalamus reaches visual cortex. Eyes optic nerves, optic tract, optic chiasma are primarily signal transfers connecting the visual area of the thalamus.

When visual signals reach visual cortex the image information is analyzed, recognized and interpreted in terms of knowledge. Zones 7 and 8 are inside our brain; they serve like "routers" of higher order of signals, leading to visual cortex. Visual recognition is not instant, but a process that includes above-mentioned actions of centers of a brain (Figure 4). This demonstrates that "digesting" of visual information occupies several areas of our brain and consumes a lot of brain energy.

Note that brain blood supply and neuron system were developed through hundreds of thousand years of evolution making human sapiens. Intensive use of brain segments for real time processing of visual information requires a large quantity of energy, reducing our power to think or create - pushing us back down along evolution line toward *a hominini*[1].

Thus from one point of view we have to analyze how the learner's brain works with the information delivered and then how this information is converted into new knowledge (new in terms of "customer", based on the previously mentioned definitive (DF), characteristic (CF) and predictive (PF) functions of knowledge use, which are presented in [10]). We should address also the transformation of introduced above and earlier [15],[16] functions DF into CF and PF, as society needs more "active animals", not just website followers; besides, virtual reality does not make real world better. Extending further A. Schopenhauer and Pope [22]:

"...readers will not be read..."

We might, regretfully, state that watchers in foreseen future will not be able even to read. The example of teenagers glued to video games for hours on end completely lost and locked into their own private world is all to common!

Thus handling of incoming information and investigating how knowledge is created inside our brain is a part of scheme of knowledge delivery and formation from the physiological point of view. The next steps in this direction are:

- Create a model and monitoring scheme of information flow through visual tracts, tracing activation of segments of brain;
- Experiment of formation of delivery using various channels;
- Set up experiments how visual information acts on brain information "digesting";
- Balance delivery of information though audio and text;
- Create a model of workload of a brain when mentioned information channels are used;

The result of this segment of research will insight the inputs for design of a control model of information delivery.

[1] *homini* - animal that looks like human

B. Channel delivery aspects

From theory of information point of view the question is: “What is the best combination of text, visual and audio information to maximize the efficiency of the delivery of meanings and knowledge?” Text, visual and audio are processed completely by different parts of brain, thus the effect of separating these channels (Figure 5) needs to be evaluated.

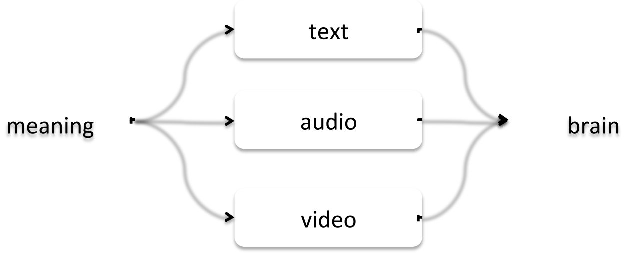


Figure 5. Delivery channels for meaning

This separation includes information volume and density, which channel might bring and strength of impact. Questions such as “how much text we should have in the process of knowledge delivery” and “how many letters, words and which structure and fonts of text sequence would be most effective” are much more important than we think.

So far nobody has provided a rigorous answer to even this simple question: “Which layout of text information should be efficiently delivered: one, two or three columns?” Book design experience [4] and web-design, see, for example [5], where the role and importance of proportions, visual and colors used described are good starting points. Note here that modeling of information delivery even in terms of text has not been considered quantitatively and systematically.

The formation of performance flow for attention control and the role of empty space was developed by Stanislavski's [6,7] and Gaston Bashlyar’s [8,9] systems. Thus formation of new knowledge and information delivery should be considered together with the aim of maximum efficiency in use of perception and learning abilities. The outcome of this research might have immediate commercial application – for web-design, education and art.

Three theories here might be used together: the theory of filters, the theory of signal processing and the theory of information. All mentioned channels (visual, audio and text) have different properties in terms of information density and in terms of absorption of delivered information within brain. The fundamental question here is: *How to combine text and visual information to maximize delivery of knowledge?*

C. Knowledge delivery using text

There is no doubt that factors such as the amount of text, its appearance and format (including fonts, color, size, number of columns) have an impact on us, they depend on each other and our ability to absorb/learn, for example: Size of sentences, Length of words all impact, Intensity of visual support (might have negative or positive impact on brain).

Text recognition is one of the fastest processes in the brain, it involves both primarily sensor groups - cones or rods (the latter for black and white sensing) that are much higher in volume – easier to perceive. Text translation in meanings within the brain activates associative memory and other areas, called sometimes “thinking schemes” [20] in contrast to visual recognition ones.

Text recognition for different languages varies in amounts of patterns (letters or symbols): Chinese, Japanese languages involves recognizing thousands of hieroglyphs while for Greek, Latin and their derivatives number of letters to recognize is only in the range 27 to 33. It has also been discovered and already documented [10] that the amount of symbols required for remembering and recognizing text and meaning has strong impacts creativity (in our terms, the predictive function gets weaker).

Clearly, such an amount of patterns, as well as texts (bible, other religious texts), requires memorizing and requires more space and brain energy, as well as increasing the length of recognition algorithms and inefficient use of energy used to decode symbols.

If we apply an algorithmic scheme of meaning decoding it becomes clear that cost of decoding has a double-hit overhead: cost for decoding by itself - cost of energy taken from the brain for decoding instead of thinking. To understand how efficiently we can process information into knowledge a set of experiments is needed to answer the questions:

- What amount of symbols and in which font and format “mean” brain can reliably absorb in given time slot?

- How many brain zones (areas) are activated to “digest” a text?

-What amount of energy from central nervous and blood supply systems both needed “to absorb and interpret” text by brain?

Then the different approaches of information delivery can be analyzed using required power, volume and time as independent variables. Taking into account that repetition and practicing might be both involved delivery of knowledge “at once” is not panacea.

Some knowledge only lodges in the brain and stays there for a long time when you practice it. For example in the area of design patterns: if you only hear and see them and you never code them you can never understand them in depth. The proposed model has to take into account constructivist approach as well [18,19,20].

Every person constructs his or her own personal reality by communicating with and interpreting the environment, it defines speed and cost of individual knowledge absorption.

In principle, the basis of analysis of information delivery via audio is similar to visual and text: analysis of segments of brain involved and estimation of audio information; amount of required energy to consume/process information. Audio flow of information might be measured and analyzed in terms of:

- Number of bytes per second;
- Frequency range use impact;
- Mean time of reliable delivery of segment of information;
- Size of sentences; length of words;
- Foreground and background noise.

The open questions related to audio form of knowledge delivery are:

- How long can speech be delivered with understanding? Is the rule of golden 20 min justified?
- What if we use text and audio? Does it make an impact? How much it extend attention time?
- Audio impact on alpha rhythm "massaging" of the brain
- Impact on information delivery when text delivered via audio and video means (for example subtitles …);
- Content vs. the length of speech, density of the meaning? (math, programming, philosophy)?
- Impact of repetition - how it works here with text? Audio & text? Only audio? Only visual?
- Web-delivery of knowledge? How it might be maximized in efficiency?

This opens the question of whether services like YouTube can ever be really effective delivery platforms for education. When this phase is relatively clear, semantics of phase one is in need of further "fine tuning" i.e. how do we know what the goals are needed? Two key dimensions are here to consider: science and industry where the most of our students will work.

In addition, neither [11] nor [12] nor [13] even mention resource dependencies of the discipline and the means for its delivery, as well as formation of the curriculum and its required assessment schemes. The delivery process mentioned above heavily depends on student's innate ability to learn. For weaker students one has to prepare sort of spoon-feeding forms of teaching and frequent assessments. For the more academically inclined the model of curriculum delivery and assessment needs to different with higher level of abstraction and self-study involved, supported infrequent assessments. To some it up with an old adage: "horses for courses".

There is a need to prepare a quantifiable model of workload of the student and the lecturer taking into account the range of abilities and prior knowledge of the intended students. Clearly that longer journey costs more and it requires greater efforts, thus quantitative analysis and models of curriculum delivery is justifiable subject of research.

V. Web-supported assessment

Up to now, except for pure math disciplines, the rigorous formation of a set of essential questions relevant to the current curriculum has been very rare. A logically cohesive sequence of the questions should be arranged for the assessment, mapped to the process of delivery of the discipline; even a semi-automatic procedure might be quite useful, as it reduces teacher's workload in formation of assessment and, at the same time, guarantees quality.

The ability of student knowledge should be formally measured, analyzed and terminated when PFs > PFtr. Indices stand for: s - student and tr - threshold). Threshold indicates the required level of understanding, proven by assessment through analysis of abilities developed in the student's mind.

The two equations below define course content in terms of essentiality and completeness by making joint and essential glossaries:

$$DFj = DF_1DF_2 \ldots DF_{n-1}DF_n$$

$$DFe = DF_1DF_2 \ldots \cap DF_{n-1} \cap DF_n$$

The building of a set of questions around an essential glossary DFe forms an essential set of questions CFe and this constitutes a complete formal preparation of the required assessment. Also the combination of DFe and CFe form a core of the discipline assessment.

A. Modes of training/assessment

Assessment using web has two major advantages. Firstly, it can be introduced into the learning process in a training mode, thus becoming an integral part of learning and increasing self-assurance and confidence of learner. During the learning process the student is able to make his or her own judgment using a "training mode" of assessment and there is also the advantage that this can be done anywhere in the world and at anytime.

Secondly, in a secure environment during a defined and scheduled assessment time the student has much more confidence regarding the subject and process of assessment as a benefit of their 'assessed' previous training. Training mode might be organized with at various depths and coverage of assessment making available, in principle, unlimited amount of generated questions on the subject.

An interesting research direction is to consider the application of AI for the formation of narrower, special purpose created assessments for any student during training mode, for checking and improving own knowledge on different segments of the subject taught. To achieve this a new web-based tool and formation of semantic channeling and tunneling as a function of student knowledge and progress is crucial. We strongly believe that our computer age needs revision and radical improvements regarding how teaching is achieved as well as a revision of the theoretical principles of curriculum design and automated support for the formation of relevant assessments.

B. Individual Assessment Tool

A supportive net for further development of theory of curriculum design for computer science, invented theoretically [1], estimated in terms of reliable delivery [2] initially tested [14] and developed up to implementation 15],[16] during last few years needs an assessment re-development. A special assessment tool automatically tuned for individual automated assessment is required. Temperature of the problem grows along with grow of number of students (up to 250 - 300 per module so far), while web-classes will attract thousands [17], or even millions of students, e.g. teaching English in Europe and Asia existing methods and schemes will not be helpful.

The automatic assessment tool we propose can be used in a form of multiple choice answer approach (MCA) [16]. This approach assumes several logically connected (say m) correct answers from n possible options.

The development of special individual assessment tool (IAT) to create and run MCA would be useful across a wide spectrum high education disciplines. An automated individual reenter-able scheme of self-assessment with IAT helps students to assess themselves during semester using their gadgets for checking own performance, seek relevant advice and even pass module as soon as they are ready.

First platform independent version of IAT was released for Apple IPAD, see screenshot on Figure 6 with implementation of:

-Multiple choice answer approach, MCA
-Penalty functions, their forms and quantification
-Iterations of learning and assessment: how it should be
-Individualization of assessment

Figure 6. First version of IAT application of WEDUCA

VI. CONCLUSION

- Proposed approach provides a theoretical basis for formation of the curriculum for high education using web-semantic methods and schemes for clustering web-based information

- Monitoring and observing changes in knowledge domain, including segmentation and landscape evolution are possible to exploit using web-semantic searching methods making automatic curricular adjustment possible

- A framework for curriculum development and delivery including human perception features and three ways of information delivery (audio, visual and text) is described toward improvement in quality and efficiency of personal knowledge formation; brief description of essential models are given

- A formation of automatic supportive assessment mechanisms with substantial and justifiable coverage as well as prototype individual assessment tool are described and illustrated.

REFERENCES

[1] I. Schagaev, “Yet another approach to classification of redundancy IMEKO,” CIM, Helsinki, Finland, pp. 117-124, 1990.

[2] I. Schagaev, “Reliability of malfunction tolerance,” IMCSIT, 2008.

[3] J. Swift, “A tale of a tub,” Pinguin books, 2004, ISSBN 0-141-01887-9.

[4] A. Hanslam, “Book design,” Series: Portfolio, 2006, ISSBN 13-978-1-85669473-5.

[5] S. Plyaskota, “Web-design course for MSci and MBA,” Course materials for Highs School of economics, Moscow, 2006.

[6] K. Stanislavski, 1936, “An actor prepares,” London: Methuen, 1988, ISBN 0-413-46190-4.

[7] K. Stanislavski, “An alphabetical arrangement of concise statements on aspects of acting,” Methuen, ISBN 0-413-63080-3.

[8] Bachelard Gastón, “Dialectic of Duration,” Cinamen Press Limited, ISBN 1-903-08307-9.

[9] Bachelard Gastón, “The Poetics of Space,” Penguin Group USA, ISBN 0-670-56269-6.

[10] Hannas William C., “The writing on the wall,” University of Pensilavaniya Press, 2003, ISBN 0-8122-3711-0.

[11] R. Diamond, “Designing and improving courses and curricula in higher education: A systematic approach: Jossey-Bass,” SF,1989.

[12] P. Ramsden, “Learning to teach in higher education,” London: Routledge, 1992, DOI: 10.4324/9780203413937.

[13] S. Toohey, “Designing courses for higher education,” Buckingham, UK: The Society for Research into HE & Open University Press, 1999.

[14] M. Robbins, I. Schagaev, J. Yip, “Effective teaching in theoretical IT modules,” 33rd Annual Frontiers in Education Conference," (FIE'03), 2003, http://doi.ieeecomputersociety.org/10.1109/FIE.2003.1263280.

[15] E. Bacon, N. Folic, N. Ioannides, I. Schagaev, “Curriculum design, development and assessment for computer science and similar disciplines,” FECS'12 - The 2012 Int’l Conference on Frontiers in Education: Computer Science and Computer Engineering, Las Vegas, July 2012, Paper ID: FEC4130.

[16] E. Bacon, N. Folic, N. Ioannides, I. Schagaev, “Multiple choice answers approach: Assessment with penalty function for computer science and similar disciplines,” International Journal of Engineering Education Vol. 28, No.6, pp. 1-7, 2012.

[17] D. Patterson, Keynote speech, Worldcomp2012, Las Vegas, July 2012.

[18] K. Reigh, “Konstruktivistische Didaktik – Lehr- und Studienbuch mit Methodenpool,” 4. Ed, Beltz Verlag, 2008.

[19] G. Hagel, J. Mottok, M. Utesch, D. Landes, R. Studt, “Software engineering lernen für die berufliche Praxis - Erfahrungen mit dem konstruktivistischen Methodenbaukasten,” im Tagungsband des Embedded Software Engineering Kongress, 2010.

[20] M.Baroni, A.Lenci, “Distributional memory: A general framework for corpus-based semantics,” Computational Linguistics. V.36, Issue 4, 2010, pp. 673-721.

[21] M. Charnine, V. Charnine, “Keywen category structure,” Wordclay, USA, 2008, pp.1-60.

[22] Schopengauer - On the Suffering e World, Penguin Books ISBN-0-191-01894-1

[23] http://teachflowapp.com

A Curriculum Model for Preparing K-12 Computer Science Teachers

Cameron L. Fadjo[1], **Ted Brown**[2], **and Leigh Ann DeLyser**[3]
[1] The CUNY Institute for Software Design and Development,
City University of New York, New York, NY, United States
[2] Department of Computer Science, Graduate Center,
City University of New York, New York, NY, United States
[3] Department of Computer Science, New York University
New York, NY, United States

Abstract - *A comprehensive approach to preparing in-service and pre-service K-12 teachers to teach computer science and software engineering through an interdisciplinary masters-level graduate program is proposed and is described herein. Consisting of traditional graduate-level courses, field-based classroom experiences, and industry mentorships, the proposed program is designed to prepare current (in-service) or prospective (pre-service) teachers to develop the practical, theoretical, and conceptual knowledge necessary to effectively teach computer science and software engineering in the K-12 classroom. Current trends in computer science K-12 teacher education are examined and our innovative model for comprehensive teacher preparation in the fields of computer programming, computer science, and software engineering instruction is described.*

Keywords: Computer Science Education, Software Engineering Education, Computer Science Teachers, Computing, Computational Thinking, Pedagogy

1 Introduction

Computer science and software engineering education is re-emerging as a major academic discipline within the 21st century educational system. From online schools and MOOCs to immersion schools and programs [15], the traditional education system is being transformed, upended by a surge in interest fueled by industry and government mandates for innovative approaches to preparing students for careers in new and emerging high tech fields [1]. Paradoxically, however, the rapid rise in innovative programs to improve the education of young students quite frequently overlooks the professionals that are most likely to lead the transformation - the teachers. In the United States, where the need to prepare students for the new economy is particularly acute, there are a few programs, like the NSF-funded CS10K project [16] and the Georgia Computes! [17] program, that are working toward solving the most critical question in computer science education: How do we best prepare computer science educators to enter the classroom and teach computer science to all students [18]?

Since the post-dot-com resurgence of computer science, many institutions of higher education have attempted to address the need to prepare the two major teacher populations: namely, pre-service and in-service teachers. For pre-service teachers, those who are enrolled in an undergraduate or graduate-level teacher preparation program before entering the classroom as a full-time professional, the number of programs available in computer science education has slightly increased in tandem with the perceived local need to fulfill this occupational role. In Indiana, for example, Purdue University has developed a 'Computer Science Teaching Endorsement' which offers an undergraduate a sequence of education and computer science courses that would provide her or him with a 'teaching endorsement' [4] in K-12 computer science. To ensure that students in this program are 'prepared' to teach, endorsement candidates in this program take two courses in the CS Education focus area [4] ('Contemporary Issues in Computing,' an examination of technology and society, and 'Methods of Teaching Computer Science,' which surveys current theories of computer science education), as well as four CS content courses in computer programming and computer science [6]. Unfortunately, severely low enrollment numbers [7] and a lack of state-level recognition for computer science as an academic content area [8] have contributed to an ineffective start to this teacher preparation program in CS Education at Purdue. Remarkably, in states where high school-level computer science courses count as a "core" credit and programs have been developed at a local college or university to prepare the pre-service teachers, such as in Georgia, low enrollment numbers continues to be an issue for pre-service teachers [7]. In both cases, the lack of coordination between the colleges/universities and the state-level Boards of Education seem to be a primary contributor to the lack of success in launching rigorous computer science education programs that lead to a professional pathway for pre-service teachers.

For in-service teachers, professional development models in computer science education have often been designed to take a professional, either teacher or industry, and prepare them through a one-day, one-week, or longer workshop or class for the rigors of teaching computer science at the high school level. These programs have been run through the College Board (with AP workshops), through universities or colleges, or even through individual school districts that service multiple schools and have a cadre of teachers to train. Programs for in-service teachers, such as the ones mentioned above, often focus on the individual needs of a particular population or school district and neglect to consider the need to build capacity within the immediate cohort to build capacity. With the graduate-level course sequence we are proposing, the bigger picture of developing a program that both meets the current needs of the computer science and software engineering programs throughout New York City [14] was the bedrock of our discussions to produce a model of professional development that is inclusive of pre-service and in-service teachers who may or may not have a background in computer science.

1.1 Certification and the Challenges It Presents

An important piece to consider when designing any pre-service or in-service teacher preparation program in the US is that of teacher certification. Teachers of computer science face many issues regarding certification, and the individuality of state certification programs present an additional challenge when considering the design and development of a national program for teacher preparation in computer science education [9]. Many states allow any subject certified teacher to teach computer science, while others require either a specific subject discipline such as Mathematics, Business/Library Media, Science or a technology specific certification, but have no certification for computer science.

In many of the current teacher preparation programs around the United States, teachers receive formal pedagogical training in the prototypical core discipline (such as Mathematics or 'Technology') but many, if not all, of these same programs that claim to prepare teachers for computer science neglect the integration of best practices from the field of computer science education into the instruction of computer science pedagogy. The 'Computer Science Education', Educational Specialist program at Nova Southeastern University [5], for example, is a graduate-level program that is designed to prepare teachers for the K-12 computer science classroom. Unfortunately, the program takes a modest approach to ensuring that a relatively broad range of STEM student are prepared to teach some form of computer science in the K-12 classroom and do so at the risk of neglecting the core of computer science education. Given the recent emergence of a core set of best practices that computer science educators use to teach abstractions [20, 21], there is also research to support the identification and use of pedagogical content knowledge (PCK) that is specific to the academic discipline being taught [10]. The lack of specific preparation in computer science pedagogy leaves many teachers with the need to do additional professional development outside of their programs in order to feel confident teaching computer science.

In New York State, computer science is most often counted as a mathematics credit, and most teachers who teach computer science also have a mathematics certification [9]. Preparing teachers for appropriate certification in mathematics education as well as computer science education are conflicting goals, however we outline a sequence of graduate-level courses that, we believe, must be integrated in any program addressing teacher preparation that ensures teachers are prepared for the computer science classroom while being simultaneously prepared to obtain a state-level teaching credential.

2 Building Teacher Capacity Through a Comprehensive Curricular Program

New York City is well positioned to create a large-scale computer science teacher educational program. Over the past couple of years, the Mayor of New York City, Michael Bloomberg and his administration have developed a number of initiatives and programs that are designed to establish New York City as a national high-tech hub. Many of these programs, such as the Center for Urban Science and Progress (CUSP) [11], and collaborations, such as the partnership between Cornell University, Technion – Israel Institute of Technology, and the City of New York, have become the hallmark of the public-private partnership model [12].

At the primary and secondary school level, in particular, the Bloomberg Administration has been working to ensure that computer science and software engineering education is open and accessible to a wide range of students. Last year the Bloomberg Administration, in collaboration with private industry partners such as venture capitalist Fred Wilson from Union Square Ventures and software engineers from high tech companies such as Google, Facebook, and Foursquare, partnered with the NYC Department of Education to create the Academy for Software Engineering (AFSE) [13], with a second school (the Bronx Academy for Software Engineering - BASE) opening in the fall of 2013. More recently the Administration announced the Software Engineering Pilot (SEP) program, a comprehensive, full-year computer science and software engineering curriculum, that will be placed in twenty NYC public schools this coming fall (2013) [14]. Forty in-service teachers, two from each of the twenty SEP schools, have been selected to participate in an extensive professional development program that will culminate in the teaching of a broad-based computing curriculum to over 2,000 students across NYC.

In the case of all three NYC Department of Education Software Engineering initiatives (AFSE, BASE, and SEP), there currently exists no graduate-level courses or programs, either locally or nationally, that prospective pre-service or in-service teachers can take to prepare them for a career as a computer science and software engineering educator. For AFSE and BASE the need to identify qualified educators is particularly acute since both schools require teacher candidates with a substantial background in computer science, pedagogy, and instructional technology as a baseline qualification.

3 Proposed Computing Education Degree

A thirty-six credit masters degree has been mapped out for in-service and pre-service teacher candidates in conjunction with experts in computer science, the learning sciences, computer science education, and instructional technology from multiple departments and institutes within the City University of New York (CUNY) and leaders from the Software Engineering high schools (AFSE) within the New York City Department of Education. The following nine courses (27 credits of a 36 credit masters) are the core of the proposed graduate-level course sequence that will eventually culminate in a Master of Science (M.S.) degree in Computing Education (see Table 1).

Group	Course	Semester	Prerequisite(s)
Computing & Mathematics	Computational and Mathematical Thinking 1 (CMT 1)	A	-
	Computational and Mathematical Thinking 2 (CMT 2)	B, C	CMT 1
	Computational and Mathematical Thinking 3 (CMT 3)	C, D	CMT 2
	Discrete Structures (DS)	A	Calculus
	Algorithms (AL)	B, C	CMT 1; DS
	Computational Science (CN)	C, D	CMT 2; CN
Computing Education	Pedagogical Practice and Methods in Computation and Mathematics (PMCM)	A, B, C	CMT 1
	Practicum in Computing Education (PCE)	A, B, C, D	PMCM
Learning Sciences & Technology	Foundations: Learning Sciences & Technology (LST)	A, B, C, D	-

Table 1: Proposed Master's Degree Courses in Computing Education

Notes: The 'Group' corresponds to the specific discipline with which the courses are most directly connected. 'Semester' A, B, C, and D correspond to the semester during which each course should be taken. For example, Computational and Mathematical Thinking 2 (CMT 2) will be offered during Semester B and C (the second and third semester of the program sequence).

4 A New Approach to K-12 Computer Science Education Teacher Preparation

At the foundation of the proposed program, and the nine courses that comprise the core of the program, is a comprehensive approach to teacher preparation in computer science education (see Figure 1).

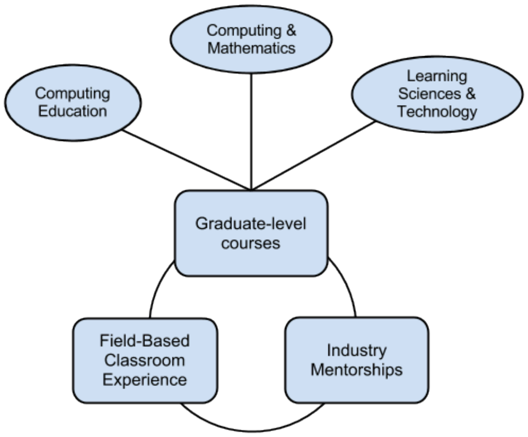


Figure 1: Computing Education Degree: Conceptual Framework

4.1 Graduate-Level Courses

The proposed courses for the Computing Education degree consists of courses across three sub-groups: Computing Education, Computing & Mathematics, and Learning Sciences & Technology.

4.1.1 Computing Education (CE)

In the *Computing Education (CE)* sub-group the two proposed courses, Pedagogical Practice & Methods in Computation & Mathematics (PMCM) and Practicum in Computing Education (PCE), provide both in-service and pre-service teachers with the experience and theoretical foundation to develop independent instructional lessons for primary and secondary school students using grade-level appropriate content and validated 'best practices' to inform pedagogical classroom design.

4.1.2 Computing & Mathematics (C&M)

The *Computing & Mathematics (C&M)* sub-group contains a series of courses, Computational & Mathematical Thinking (CMT) 1, 2, and 3, that cover a wide range of core computer programming, computer science, software engineering, and mathematical thinking topics. As the core content courses for the degree, CMT1, 2, and 3 are designed to establish a solid foundation in computing while simultaneously introducing key mathematical thinking principles to ensure teachers are prepared to teach abstractions in the K-12 classroom. In addition to these core content courses, the C&M subgroup contains courses that cover discrete structures, algorithms, and computational science with a focus on 'big data.'

4.1.3 Learning Sciences & Technology (LS&T)

The *Learning Sciences & Technology (LS&T)* sub-group contains one course, Foundations: Learning Sciences & Technology, which provides all students with a solid foundation in the central disciplines (computer science, educational psychology, cognitive psychology, and applied linguistics) that define the learning sciences within the context of computer science education.

4.2 Field-Based Classroom Experience

In addition to the proposed sequence of courses, each student will be placed in a semester-long classroom-based teaching experience (in the case of pre-service teachers) or will work alongside a faculty mentor to augment a current CS education classroom experience (for in-service teachers). The experience, part of the CE sub-group course 'Practicum in Computing Education (PCE),' ensures that all teachers are receiving authentic teaching and learning experiences as they develop their content and pedagogical knowledge expertise.

4.3 Industry Mentorships

The third major component of the Computing Education Degree: Conceptual Framework is the placement of a teacher in an industry mentorship. As a degree of study that is connected to professional industry, the Industry Mentorship ensures that teachers are creating authentic learning experiences for the students while simultaneously gaining professional experiences themselves. In states where industry experience is critical to teacher certification in the technology fields, such as New York State, Industry Mentorships help to establish the professional foundation upon which a teacher can effectively build academic experiences for their K-12 classroom.

5 Conclusion

Developing a comprehensive master's degree for in-service and pre-service teachers in K-12 computer science education ensures that teachers are being prepared to enter the classroom ready to offer effective instruction. As a conceptual model, the proposed program in Computing Education provides colleges and universities with a comprehensive outline for designing and developing a rigorous sequence of graduate-level instruction that ensures teachers, upon completion of the course, will possess the requisite knowledge needed to teach computer science and software engineering at the K-12 level.

6 References

[1] Bureau of Labor Statistics, U.S. Department of Labor, Occupational Outlook Handbook, 2012-13 Edition, Software Developers, on the Internet at http://www.bls.gov/ooh/computer-and-information-technology/software-developers.htm.

[2] Margolis, J., Estrella, R., Goode, J., Jellison-Holme, J., & Nao, K. (2008). *Stuck in the Shallow End: Education, Race, & Computing*. MIT Press: Cambridge, MA.

[3] Gal-Ezer, J. & Stephenson, C. (2010). Computer Science Teacher Preparation is Critical. ACM Inroads, Vol. 1(1), 2010, 61-66.

[4] Purdue University Computer Science Teaching Endorsement (2010) Accessed from http://www.cs.purdue.edu/news/11-05-10CS4EDU.html

[5] Nova Southeastern University (nd) Education Specialist Program in Computer Science Education, Accessed from http://www.fischlerschool.nova.edu/es/computer/degreeonly

[6] Purdue University (nd) *Computer Science Teaching Supplemental Licensure Program*, Accessed from http://cs4edu.cs.purdue.edu/license

[7] Guzdial, Mark (2011) *Whats the argument for becoming a computer science teacher?*, Accessed from http://computinged.wordpress.com/2011/02/07/whats-the-argument-for-becoming-a-computer-science-teacher/

[8] Wilson, C., Sudol, L., Stephenson, C., & Stehlik, M. (2010) *Running the Empty: Failure to Teach K-12 Computer Science in the Digital Age*. Association for Computing Machinery

[9] Khoury, G. (2007). Computer Science state certification requirements. CSTA Certification Committee Report. Retrieved June 11, 2009, from http://www.csta.acm.org/ComputerScienceTeacherCertification/sub/TeachCertRept07New.pdf

[10] National Academies Press (2000). *How People Learn: Brain, Mind, Experience, and School* National Research Council

[11] Center for Urban Science and Progress (2012) *Mayoral Press Conference Announcing Launch of CUSP*, Accessed from http://cusp.nyu.edu/mayoral-press-conference-announcing-launch-of-cusp/

[12] Kaminer, April (2013) *New Cornell Technology School Tightly Bound to Business*, Accessed from http://www.nytimes.com/2013/01/22/nyregion/cornell-nyc-tech-will-foster-commerce-amid-education.html

[13] Herman, C., Tucker, D., (2012) *Teaching a New Generation of Coders and Web Developers*, Accesed from http://www.wnyc.org/shows/newtechcity/2012/sep/11/

[14] New York City Department of Education (2013) *Mayor Bloomberg and Schools Chancellor Walcott Announce the 20 Schools Selected for the New Software Engineering Pilot Program*, Accessed from: http://schools.nyc.gov/Offices/mediarelations/NewsandSpeeches/2012-2013/AppliedSciences.htm

[15] Miller, C.C. (2013) Opening a Gateway for Girls to Enter the Computer Field. The New York Times, Accessed from http://dealbook.nytimes.com/2013/04/02/opening-a-gateway-for-girls-to-enter-the-computer-field/?src=twrhp

[16] Guzdial, M. (2012) Biggest challenge for CS10K: Recruiting the Teachers. Access from http://computinged.wordpress.com/2012/02/09/biggest-challenge-for-cs10k-recruiting-the-teachers/

[17] Guzdial, M., Bruckman, A., Ericson, B., & McKlin, T. (2011) Project "Georgia Computes!" An NSF Broadening Participation in Computing Pipeline Alliance, Retrieved from http://gacomputes.cc.gatech.edu/Members/guzdial/GaComputes-2pgr-Jan11-CE21Mtg-smaller.pdf

[18] Cuny, J. (2011) Transforming Computer Science Education in High Schools. IEEE Computer Society, p. 107 - 109.

[19] Depillis, L. (2013) How to Close the Tech Industry's Gender Gap. The New Republic, Retrieved from http://www.newrepublic.com/article/112784/tech-industry-gender-gap-closing-it-starts-classroom#

[20] Fadjo, C. L. (2012) Developing Computational Thinking Through Grounded Embodied Cognition. Unpublished Doctoral Dissertation. Columbia University.

[21] Grover, S. & Pea, R. (2013) Computational Thinking in K-12: A Review of the State of the Field. Educational Researcher, Vol. 42 No. 1, pp. 38 - 43.

Development of ICT Curricula through Graduate Career Outcomes and Required Skills

I. Lewis[1], K. de Salas[1], N. Herbert[1], W. Chinthammit[1], J. Dermoudy[1], L. Ellis[1], and M. Springer[1]
[1]School of Computing and Information Systems, University of Tasmania, Hobart, TAS, Australia

Abstract - *Career outcomes are widely used by Universities to market their programs but there is scant evidence that they are attainable by graduates or if they inform curriculum design. This paper reports on a process for designing a University ICT curriculum that is directly informed by the career outcomes relevant to both local and national ICT industry. Outputs from this process are a set of classified attainable graduate career outcomes and a set of graduate skills that are the basis for the further stages of the curriculum development.*

Keywords: ICT career outcomes, ICT skills, ICT curriculum, ICT graduates, ICT degree

1 Introduction

ICT curricula are in a constant state of flux in response to continuing changes in emerging technology and resources such as staffing levels, student numbers, and funding models. It is often unclear whether specified career outcomes for particular degrees are part of the curriculum development process or just an advertising mechanism. Curriculum change is predominantly driven by outspoken individuals, budgetary constraints, and student demand rather than academic merit and external curricula [1]. In attempts to respond to external constraints and ever-changing technology it is easy to lose sight of the advertised career outcomes as a focus. Academics and students need to acquire a thorough knowledge of ICT career outcomes and that "*universities must link and publish computing programs, linking each program with specific career tracks, indicating specific career specialisation and knowledge"* [2].

There is little evidence that career outcomes as stated on marketing materials are really attainable by students. Graduate career prospects are one of the major influencing factors when pre-tertiary students (and their parents) are selecting their degree. The main reason for the lack of interest in a career in ICT by pre-tertiary students is that computing is traditionally perceived as asocial, focusing on programming and having limited connections to the outside world [3,4]. To counter this negative and inaccurate perception, and to promote the future growth of the industry, it is essential that the career outcomes for modern ICT degrees reflect the myriad of career opportunities now available and the curriculum is designed such that graduates can attain these careers.

While theoretically, linking curricula design closely with career outcomes might be an ideal situation, in practice, tertiary institutions are currently juggling the different demands of local and international students and there has been increased specialisation of programs and a correspondingly large growth in the number of units (subjects) on offer. This is a common problem as an emphasis on academic objectives tends not to be coherent but results in a large range of topics for students and will typically include the research interests of staff [5]. Alternatively, when emphasis is placed on employment objectives the resulting curricula are more directed and coherent [5].

While an abundance of units might allow for an abundance of career opportunities, this makes isolating core career outcomes very difficult and therefore also difficult for students to know exactly what units to take to achieve a desired career outcome. Graduates find it very difficult to identify ICT career opportunities that relate to the skills they have developed during their study [6,7]. Furthermore, this abundance in units, and course specialisations, makes it difficult for industry to determine solely on the basis of a graduate's degree whether they are qualified for a particular career, instead requiring knowledge of specific unit content.

The Australian Computer Society (ACS) provides a process (*what to do*) to guide the development of new curricula [8], but not the specific activities to undertake (*how to do it).* This is consistent with the more general absence of literature focussing on how to link career outcomes and ICT curricula. As a result, this paper will describe a method for identifying classified potential career outcomes and required skills during an ICT curriculum development effort guided by the ACS process. Specifically, we focus on how to perform the first three of seven steps in the ACS process to develop a new ICT degree, namely:

- *identify potential ICT roles that could be undertaken by graduates of a given program of study;*
- *identify the skills required by professionals in a given ICT career role; and*
- *identify the responsibility level required to be developed for each skill.*

As our implementation of our process will also be important to some readers, our constraints, resources, and outcomes of each phase of the process are included for completeness.

2 The process

Figure 1 outlines our four-stage process by which career outcomes and required skills are first identified, then classified before being used as inputs for subsequent curricula design decisions. The process is based on that of the ACS [8] with additional details on *how* to perform each step and feedback from each stage used to develop and refine the list of potential career outcomes.

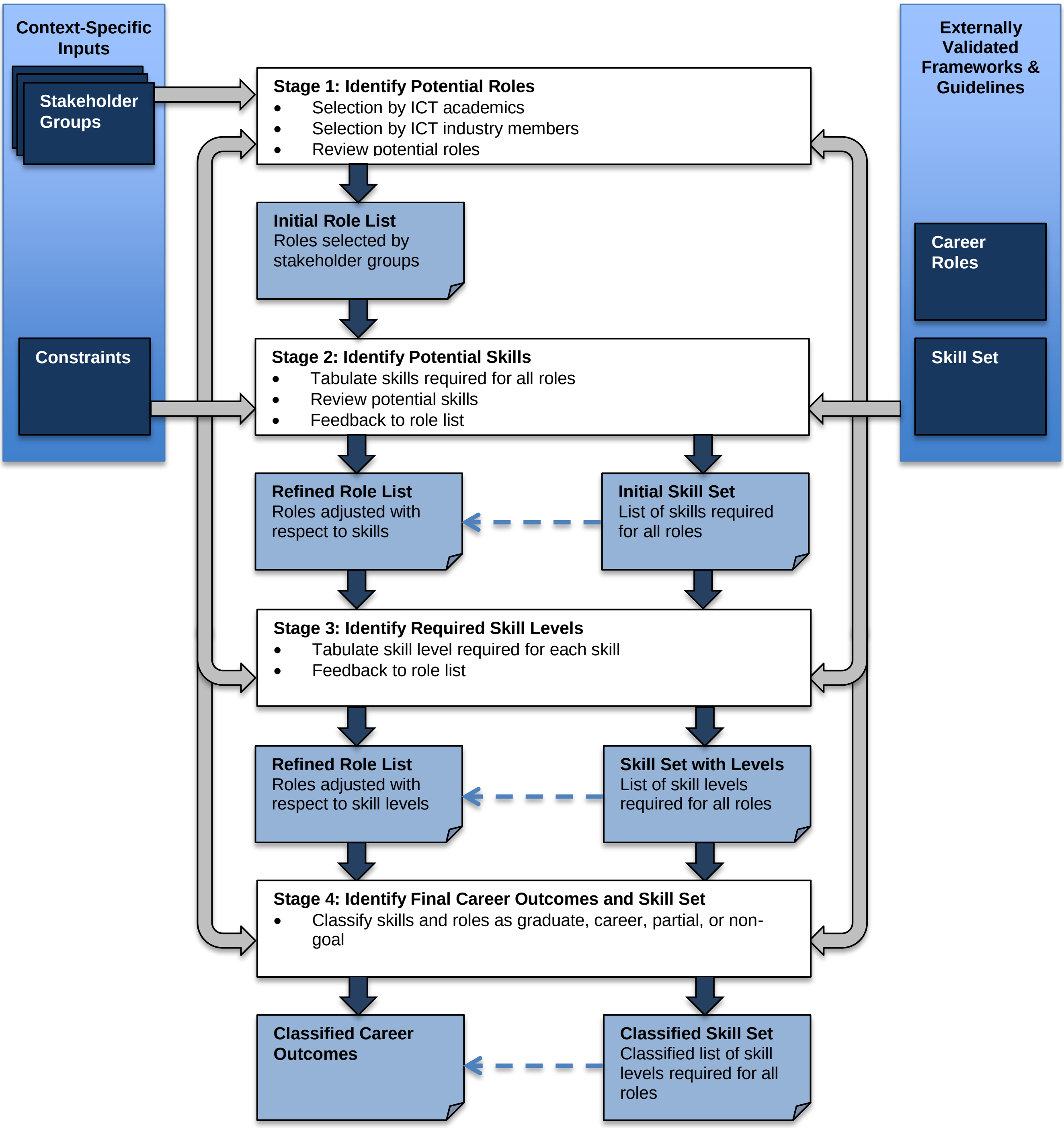

Context-Specific Inputs
Stakeholder Groups
Constraints
Externally Validated Frameworks & Guidelines
Career Roles
Skill Set
Stage 1: Identify Potential Roles
• Selection by ICT academics
• Selection by ICT industry members
• Review potential roles
Initial Role List
Roles selected by stakeholder groups
Stage 2: Identify Potential Skills
• Tabulate skills required for all roles
• Review potential skills
• Feedback to role list
Refined Role List
Roles adjusted with respect to skills
Initial Skill Set
List of skills required for all roles
Stage 3: Identify Required Skill Levels
• Tabulate skill level required for each skill
• Feedback to role list
Refined Role List
Roles adjusted with respect to skill levels
Skill Set with Levels
List of skill levels required for all roles
Stage 4: Identify Final Career Outcomes and Skill Set
• Classify skills and roles as graduate, career, partial, or non-goal
Classified Career Outcomes
Classified Skill Set
Classified list of skill levels required for all roles

Figure 1: A Process for Identifying and Classifying Career Outcomes for a Degree

2.1 Constraints

Before commencing the process it is vital that any specific constraints relevant to the curriculum development effort be identified. These constraints will impact on decisions made about career outcomes and skills developed throughout the steps of the process.

2.2 Stage 1: potential role identification

Our initial investigation into ICT degrees throughout Australia indicated that degrees aim to produce graduates qualified for a range of ICT careers, and although there are some common career outcomes, most are quite different in their emphasis. ICT is constantly changing and new technology is continuously emerging and as a result career titles and definitions are changing. Our investigation indicated there appears to be no nationally recognised standard set of career titles and definitions that are used or maintained.

If career outcomes are to be achieved, they must be embedded into design. The first essential step must be to identify an externally validated set of ICT career definitions that covers a broad range of ICT careers. External validation limits the "*influence of outspoken individuals*" [1]. At the conclusion of this stage a number of career outcomes is required to create a degree that will meet its objectives as well as give graduates options.

There is a difference between the roles a graduate could be fully qualified for on graduation and those careers that they might aspire to over time. It is useful, not only for accurate marketing to potential students and their parents but also for the latter stages of the process, to be able to distinguish between the different roles available: graduate roles that students can perform when they enter the workforce, career roles they might eventually achieve after a few years of experience, or partially-qualified roles that they might not develop all the skills required during an undergraduate degree and require further study. As a result, to guide the process of identifying relevant career outcomes it is necessary to annotate all identified roles with an identification of the extent of qualification required:

- *Fully*—indicates students should be fully qualified for graduate entry in this role. Some short specific training maybe required, but graduates are expected to be fully capable of performing this role in a business within six months. Unit content should be focussed towards this role.
- *Partially*—indicates students should have some useful skills for this role but not all. There may be content that should not be supplied at university undergraduate level. It could be supplied by another organisation or a postgraduate degree.
- *No*—indicates this is not a role to aim for with the degree (whether it is achieved by skill overlap is irrelevant at this time).
- *Unsure*—indicates the reviewer was undecided.

Unless an institution has unlimited resources or unless a small set of career definitions was chosen to begin with, it is necessary to identify a subset of the roles that are relevant for the new degree. It is recommended that input from all stakeholders is sought but at the very least a two-stage process consulting ICT academics and ICT industry members is recommended.

2.2.1 ICT academics

Academics that will be implementing the new degree should be involved in the identification process. Inviting academics who will be involved in the implementation of the new curriculum to contribute to the design from the outset builds a sense of ownership that will facilitate change [9]. To ensure that the new degree is not heavily influenced by any one individual a range of staff should be invited to identify the roles they deem relevant from the externally validated set of career definitions.

2.2.2 ICT industry members

While academics have a good understanding of the careers relevant to their graduates, it is also important to get relevant industry members to identify the ICT graduate roles "*so as to incorporate the elements that are crucial for employability of graduates as a part of curriculum development, design, training and assessment*" [7].

Each industry member should review each career definition and rate them on the same four-point scale (*Fully*, *Partially*, *No*, *Unsure*) used by the ICT academics. To ensure that the roles are actually available and attainable, each industry member should additionally rate each career as:

- *Employed*—have employed a (Bachelor's level) graduate into this role in the last three years.
- *Would Employ*—would employ a graduate into this role if a vacancy existed.
- *Not Graduate*—would not employ a graduate into this role.
- *Not Relevant*—not relevant to my organisation.

2.2.3 Potential role review

On completion of the initial identification activity, it is necessary to have a reflective discussion with industry members to share and discuss any differences in outcome identification amongst the industry members and with the careers identified by the ICT academics. The discussion should also consider the impact of any constraints.

Even the best externally validated list of career definitions may be missing some key roles that are particular relevant to an institution's particular circumstances; this stage is an opportunity to review the role list and add any missing roles. For example, there may be a significant local industry sector or a significant key research/innovation direction for the university or state. A high-quality and focused degree will also potentially attract students into research.

2.3 Stage 2: Potential skills identification

Stage 1 of our process identified a list of potential roles deemed desirable to use as a guide for the new curriculum development. While this list is a useful starting point, the next stage is to determine the specific skills required for the attainment of these roles by graduates.

As each role relies on the development of a combination of separate graduate skills, it is crucial to identify an externally validated set of skills for each career. Such skillset lists can usually be sourced from relevant industry, educational, or professional organisations. For example while the ACS endorses the use of the SFIA (Skills Foundation in the Information Age) skillset [10], our process does not mandate its use.

2.3.1 Potential skills tabulation

An essential part of the process is identifying the required combination of skills for each role be drawn from an externally validated set of career skills.

To identify a potential list of skills for the entire curriculum, each role identified as a potential role should be considered and the required skill set tabulated. The process will identify some obvious skills to include that are necessary for a number of roles as well as a number of skills that are not needed for any potential role.

2.3.2 Potential skills review

Once the skills required for all potential roles have been tabulated, a review of all the skills from the externally validated set should be undertaken to ensure that no essential skills have been missed. It may be that the set of career role definitions and the skill mappings did not cover the full list of skills. Consideration should also be given to any constraints during this review.

2.3.3 Feedback to role list

Once the list of potential skills is identified, these skills can be used to influence the list of potential roles. For example, there will be some skills that are only needed for a few roles and, if there are constraints, consideration can be given to removing these roles or not developing those specific skills and only partially qualifying a graduate for these roles.

There may be some roles that were not in the list of potential roles but an analysis of the skills required might identify that they are all being covered and the role could be included in the list of potential roles.

2.4 Stage 3: Competency level identification

To this point, a list of potential roles and the specific set of skills required for the attainment of each role have been identified. The third stage of the process is to identify the competency level required for each skill to perform each role.

2.4.1 Competency level tabulation

All the identified skills should be reviewed against the competency level required in each to determine the extent to which the skill could be developed in an undergraduate degree. The process will identify skills across a range of levels.

2.4.2 Feedback to role list

There might be a number of roles that required skills at a level beyond that of a typical undergraduate degree. For these skills there are three scenarios:

- *Skills deemed essential*—develop depth in the skill throughout all years of the degree recognising that some of the final development might be achieved in the first six months of employment.
- *Skills unachievable within the constraints*—develop these skills only to a typical undergraduate level.
- *Skills unachievable within the constraints with no lower level of competency*—graduates require 1–2 years of employment to attain required knowledge.

There may be some roles that were selected that have skill levels that are too low or too high for an undergraduate degree. Removing these careers from the list can be considered.

2.5 Stage 4: Career outcome and required skill level identification

Using combined insight developed from the previous three stages—identification of potential roles, identification of required skills for each potential role, and identification of the competency level required for each skill—an informed decision can now be made about the final set of career outcomes that would be attainable for the students, and would therefore guide the curriculum development into the future.

Given that not all potential roles and skills identified will be deemed attainable by undergraduate students immediately on completion of their studies, we developed four categories to distinguish the differences in the attainability of these career outcomes:

- *Graduate Roles*—all skills would be fully developed and the role is suitable for graduates (though they may need six months of experience to reach the specific competency level);
- *Career Roles*—all theoretical skills would be covered and the role is suitable for graduates who have acquired one to two years of experience and shown competence;
- *Partially Qualified Roles*—some key skills may be absent from the undergraduate degree which might be available from another discipline of the university or other educational institution or in a postgraduate degree; and
- *Non-goal Roles*—all the skills would be developed however the delivery of the unit content and discussion would not be focused towards these particular roles.

3 Our experience

The following section of the paper shows our experience in applying the process as described in our own curriculum redevelopment process.

3.1 Constraints

The University of Tasmania (UTAS) is the only university within Tasmania, and the School of Computing and Information Systems, as the only ICT School at UTAS, must meet the ICT higher education needs of the ICT industry in Tasmania.

An external school review conducted in 2011 recommended the consolidation of the two existing undergraduate degrees (a Bachelor of Computing and a Bachelor of Information Systems) into a single Bachelor of ICT. Additionally, due to a shrinking staff profile coupled with pressure for increased research output across lead to a recommendation to reduce the number of undergraduate unit offerings from 50 to just 30.

3.2 Stage 1: potential role identification

Our externally validated list of career roles was sourced from the Queensland (QLD) ICT Public Sector Development Office [11] ICT career streams diagram. It is maintained to keep it current, and was last updated in 2012. This diagram identifies four different career streams and 55 key ICT roles. The online version of the diagram is interactive and selecting a role will take the user to further information that clearly defines the role and also has information essential to the later stages of our process as it identifies the SFIA skill set required to perform the role along with the competency level (referred to as "level of responsibility" in SFIA) for each skill [10].

Given our constraint of only 30 units (as recommended from our external review) it was necessary to identify a much-reduced subset of the 55 roles that would be career outcomes for our new degree. At the conclusion of this stage a broad range of career outcomes was required in order to create a non-specialised ICT degree that would have wide appeal.

To identify a practical subset consultation was sought from both academics within the school and local ICT industry members.

3.2.1 ICT academics

A working party was formed consisting of eight academics, heavily interested in teaching and learning with a variety of different characteristics and backgrounds: drawn from geographically separate campuses; three primarily from the Information Systems discipline, five primarily Computer Science; and three being female and five male.

Only a small number of roles, 8 (out of 55), received 5 (out of 8) or more *Fully* votes. There were 16 roles that received at least 75% (6 out of 8) of the votes when combining the *Fully* and *Partial* votes. The results are shown in Table 1. The careers that are different to the ICT industry member responses (as discussed further in the next section) are shown with a shaded background.

3.2.2 ICT industry members

Three industry forums were held and eighteen representatives of the local and national ICT industry and Government participated in an exercise to identify career outcomes with a broad range of organisational focusses including: recruitment, consulting, security, education, research, development, and government.

Our attendees represented organisations with varying number of ICT employees from one to thousands. Nearly all had employed graduates into various positions throughout their career; most less than ten, some as high as fifty or more.

Only a small number of roles, 5 (out of 55), received 10 (out of 18) or more *Fully* votes from the industry representatives. There were 12 roles that received over 75% (14 out of 18) of the votes when combining the *Fully* and *Partial* votes. The results are shown in Table 1. The careers that are different from the ICT academic responses are shown with a shaded background.

Additionally, the industry members were asked if they would have employed in the past or would employ in the future a graduate into the role. The total number of respondents (out of 18) is shown in Table 1.

3.2.3 Potential role review

The most interesting and relevant points from a discussion between all parties who had participated in the career outcomes exercise were:

- Employers commonly place graduates in a Help Desk Operator role initially to test competence, and if they show ability, they are quickly advanced to a Systems Administrator or Software Developer role.
- Industry members believed the role of Graphics Designer was attainable and of high demand, however it was questioned whether this role was likely to be attained by graduates solely undertaking an ICT degree, as specific skills would be required from Fine Arts related units.
- Only two industry members identified the Game Developer role as one that should be *Fully* achieved, but all recognised that this role was a strong draw card for students and they welcomed the potential increase in graduate numbers it provides.
- Industry members believed it was essential that graduates were exposed to concepts in project management and business analysis during their degree but that a graduate could not enter into a Project Manager or Business Analyst role without job experience. Once shown competent, they would be rapidly promoted to these roles.

Table 1: Roles identified by working party and industry

Agreement on Role's Importance	Industry Would Employ
> 50% Fully Votes	
Data Modeller	14
Software Designer	12
Software Developer	16
Web Developer	12
Database Administrator	11
Systems Administrator	
Project Manager	
Games Developer	
> 75% Fully + Partial	
Systems Analyst	12
Help Desk Operator	11
Network Analyst	7
Security Specialist	10
Business Process Modeller	11
Project Support Officer	10
Multimedia Designer	5
Multimedia Developer	7
Technical Architect	8
Security Architect	
Testing Manager	
Network Manager	
Information Management Specialist	
Solutions Architect	
Technical Development Manager	
ICT Manager	
Graphics Designer	9
Business Analyst	11
Project Manager	10
Added to Meet Constraints	
ICT Researcher	

Key
Agreement between academics and industry
Roles identified by academics only
Roles identified by industry only

The ICT careers stream diagram was very focused on business careers and does not include titles that might fall under ICT Scientist or ICT Researcher. These careers are not necessarily of high interest to industry, but given that one of our constraint is an increase in research output it is clear that they are of significant interest to the University and the School and other research institutes within Tasmania especially with the introduction of the NBN, Sensing Tasmania, CSIRO ICT Centre, and the HITLab. As a consequence ICT Researcher was added to the list of potential roles as shown in Table 1.

3.3 Stage 2: potential skill identification

In our process we relied on the SFIA skillset as it is the world's most popular definition of information technology skills. SFIA provides a common reference model for the identification of the skills needed to develop effective information systems making use of ICT. It provides a standardised view of a wide range of professional skills needed by people working in information technology. Specifically, it lists 86 professional ICT skills, with each skill being mapped across seven levels of “responsibility” (i.e. competency).

Table 2: Roles identified after skill identification

Roles	
Fully Qualified Roles	
Data Modeller	Security Specialist
Software Designer	Business Process Modeller
Software Developer	Project Support Officer
Web Developer	Technical Architect
Database Administrator	Security Architect
Systems Administrator	Testing Manager
Project Manager	Technical Development Manager
Games Developer	ICT Manager
Systems Analyst	Business Analyst
Help Desk Operator	Project Manager
Network Analyst	ICT Researcher
Benefits Analyst	Customer Services Manager
Animator	Incident Manager
Hardware Engineer	Change Manager
Partially Qualified Roles	
Multimedia Designer	*Information Management Specialist*
Multimedia Developer	*Solutions Architect*
Network Manager	*Graphics Designer*

Key
Fully qualified roles selected by stakeholders
Incidentally qualified roles
Partially qualified roles

Table 3: Final Career Outcomes

Roles	
Fully Qualified Roles	
Data Modeller	Systems Analyst
Software Designer	Business Process Modeller
Software Developer	Project Support Officer
Web Developer	ICT Researcher
Games Developer	
Career Roles (After 1 or 2 Years Experience)	
Database Administrator	Security Specialist
Systems Administrator	Technical Architect
Project Manager	Security Architect
Network Analyst	Business Analyst
Non-goal Roles	
Benefits Analyst	Customer Services Manager
Animator	Incident Manager
Hardware Engineer	
Partially Qualified Roles	**Missing Skill**
Testing Manager	TEST Level 5
Help Desk Operator	USUP, SLMO
Technical Development Manager	Level 6
ICT Manager	Level 6
Change Manager	CHMG Level 5
Multimedia Designer	INCA
Multimedia Developer	INCA
Network Manager	NTDS, ITMG
Information Management Specialist	IRMG
Solutions Architect	ARCH
Graphics Designer	INCA

3.3.1 Tabulation of skills for potential roles

We identified 38 skills needed for our potential careers out of the 86 defined by SFIA including a number of skills that were necessary for a range of careers.

3.3.2 Reviewing potential SFIA skills

When reviewing all the SFIA skills that were not included, we discovered two that we decided to include: HFIN (Human factors integration) and UNAN (Non-functional needs analysis). Both these skills relate to the recommendation in the ACM IT curriculum [12] that user-centeredness become a pervasive theme.

3.3.3 Feedback to role list

We identified a number of roles that require a specialist skill that we were unable to include due to our constraints on unit numbers and roles that were not identified as potential roles for which all the skills are covered incidentally (see Table 2).

3.4 Stage 3: level of responsibility identification

SFIA recognises seven levels of responsibility (competency level) ranging from 1 at basic entry to 7 at a very senior level, normally in a large organisation [10]. This step is guided by the ACS recommendation that undergraduate degrees should produce graduates with skills around SFIA level 4 of responsibility [8].

3.4.1 Tabulation of level of responsibility

Given the constraint of 30 units, we reduced three skills to level 4 as we will be unable to develop these to level 5. We develop three skills with lowest level 5 but these skills require experience to fully achieve based on feedback from industry members.

3.4.2 Feedback to role list

Whilst we develop all the skills required for the Help Desk Operator role this career is not considered as a worthy outcome of a university degree but more as a side-effect outcome of the degree, as many students take on these roles before graduation.

We added the Project Support Officer and Data Modeller required as these were seen as good graduate roles that students could use to enter the workforce.

Both the Technical Development Manager and ICT Manager require skills at level 6. As both careers also require other skills that we will develop, it was decided these would be partially qualified roles for the new degree.

3.5 Stage 4: final career outcome identification

In total we have identified 33 career outcomes for our degree as shown in Table 3. The categorisation resulted in the identification of 9 graduate roles that would be immediately attainable by our graduating students and would thus be our primary focus in developing an ICT curriculum. In addition to these core ICT graduate roles, we also identified 8 career roles, 11 partially qualified roles; and 5 non-goal roles.

Our curriculum will develop 31 skills (12 to level 4, sixteen to level 5, and three to almost level 5 but experience is needed to achieve that level of responsibility). The skills will be embedded throughout the units, and each unit will work towards developing a number of skills.

3.5.1 Next steps

Having identified the career outcomes, skills and level of responsibility we went onto complete the final steps of the ACS process [8] as documented in Herbert *et al* [12] and are currently creating the units based on the ACM international curricula [13]. Having completed the first draft of the framework we have identified the equivalent of 29 units to cover these skills at the required level to be developed throughout 2013 for delivery in 2014.

4 Summary

Curriculum design is a complex process that must be informed by stakeholders and developed from multiple perspectives. In creating a new ICT curriculum we determined a need to identify those careers that would be attainable by our graduate students and guide our future curriculum design process. While career outcomes seem a logical place to commence curriculum design, there exists little direction available to guide the process of identification and evaluation of potential career outcomes and the required skills for each.

This paper reports on the development of an ICT curriculum that was guided by the ACS recommended process for developing curricula and provides practical suggestions for undertaking the first three steps:

- have a range of academics and industry members select the roles within the constraints using an externally validated set of roles with clear definitions;
- using the roles selected, identify the skills relevant to each career, and use these to modify the list of potential roles (to both remove some options and introduce others) within the constraints; and
- identify the competency level for each skill within any constraints and use this level for each skill to refine the list of roles and to classify these roles to aid in accurate marketing of the degree.

In following the stages of this process, career outcomes can be identified that are informed by a balanced view of academic insight and employer needs, both being further supported by externally validated and industry-standard skill definitions. Furthermore, potential students can be assured that advertised career outcomes as are really attainable, and that the degree was developed with these career outcomes in mind.

Following the process documented in this paper has succeeded in producing an ICT degree curriculum and given the participants confidence that by following this process a curriculum development team can:

- determine exactly what career outcomes from the degree will be covered completely, which will be covered partially, and which will not be covered at all;
- be guided by career outcomes when developing and making decisions about what skill set to include in specific curricula;
- avoid the problems of outspoken individuals having undue influence on curricula; and
- reduce the number of units to operate within budgetary constraints, allowing time for staff to do research, and still offer a broad range of career outcomes to meet student and industry demand.

5 References

[1] Gruba, P., Moffat, A., Søndergaard, H., & Zobel, J. (2004): What Drives Curriculum Change? in *Proceedings of the Sixth Australasian Computing Education Conference*, ACS, pp 109–117.

[2] Calitz, A.P., Greyling, J.H., Cullen, M.D.M., 2011, ICT Career Track Awareness amongst ICT Graduates, *ACM SAISSIT'11*, October 3-5, 2011, Cape Town, South Africa, pp 59–66.

[3] Babin, R., Grant, K. and Sawal, L., 2010. Identifying Influencers in High School Student ICT Career Choice. *Information Systems Educational Journal*, 8(26).

[4] Biggers, M., Brauer, A. and Yilmaz, T., 2008. Student Perceptions of Computer Science: A Retention Study Comparing Graduating Seniors vs. CS Leavers. *ACM SIGCSE'08*, 12-15 March 2008, Portland Oregon, USA, pp 402–406.

[5] Henkel, M. and Kogan, M. (1999), Changes in curriculum and institutional structures, in C. Gellert. ed., '*Innovation and Adaption in Higher Education'*, Jessica Kingsley Publ., 116 Pentonville Road, London, N19JB, England, Chapter 2.

[6] Alexander, P.M., Holmner, M., Lotriet, H. H., Matthee, M. C., Pieterse, H.V., Naidoo, S., Twinomurinzi, H. and Jordaan, D., 2010, Factors Affecting Career Choice: Comparison between Students from computer and other disciplines, *Journal of Science Education and Technology*. Springer, 16 October 2010.

[7] Nagarajan, S. & Edwards, J. (2008): "Towards Understanding the Non-technical Work Experiences of Recent Australian Information Technology Graduates" in *Proceedings of the Tenth Australasian Computing Education Conference*, pp 103–112, ACS.

[8] Australian Computer Society (2011): Accreditation Manual, ACS.

[9] Elizondo-Montemayor, L., Hernandez-Escobar, C., Ayala-Aguirre, F., & Aguilar, G. M. (2008). Building a sense of ownership to facilitate change: The new curriculum. *International Journal of Leadership in Education, 11(1),* 83-102.

[10] SFIA Foundation, Skills Framework for the Information Age. http://www.sfia.org.uk, Accessed 8 Aug 2012.

[11] QLD Government, Chief Information Office, Department of Science, Information Technology, Innovation and the Arts, http://www.qgcio.qld.gov.au/qgcio/projectsandservices/ictworkforcecapability/Pages/ICTcareerstreams.aspx, Accessed 8 Aug 2012.

[12] Herbert, N., Dermoudy, J., Ellis, L., Cameron-Jones, M., Chinthammit, W., Lewis, I., de Salas, K., Springer, M., (2013) Industry-Led Curriculum Redesign, Submitted to *Proceedings of the Fifteenth Australasian Computing Education Conference*, ACS.

[13] ACM, Association for Computing Machinery. http://www.acm.org/. Accessed 8 Aug 2012.

Firing Up All Cores While Introducing Parallel and Distributed Concepts

Jie Liu[1], **David Olson**[1], **Feng Liu**[2], **Mitch Fry**[1]
[1] Computer Science Division, Western Oregon University, Monmouth, OR, USA
[2]Department of Computer Science, School of Computer Science, Beijing Jiaotong University, Beijing, China

Abstract - *Our schools, like most Computer Science programs in the world, are still teaching most of concept in Computer Science under the old sequential execution assumption, even in an era where multicore computers are common place in the market. This paper presents our attempt of expanding our students' horizon by introducing them the concepts and skills in parallel programming and parallel processing.*

We will discuss the benefits of such a class, what concepts and topics are covered in the class, and the evaluation of our teaching.

Keywords: Parallel programming, multi-core, parallel and distributed computing, algorithm complexity, and curriculum

1 Introduction

In the past few years, the development in nanotechnology has allowed chip manufacturers, such as Intel and NVIDIA, to produce CPUs with more and more cores. For example, NVIDIA Tesla C2075 video card has 448 cores on a single GPU, and Intel's Knights Corner 3120A delivers 1.003 double precision teraflops with 60 1.053 GHz cores. Nevertheless, software is still not designed to utilize the additional cores. Checking into our school's 40 capstone projects for seniors for the past five years, only one project applied parallel processing despite the fact that most groups have at least one member knows how to activate tasks concurrently.

With the help of the Spring-12 Early Adopters - NSF/IEEE-TCPP Curriculum Initiative on Parallel and Distributed Computing for Undergraduates program, we are systematically and purposely introducing parallel and distributed concepts (PDC) related topics in several classes. Still, the class with most concentrated PDC concepts is our CS474 Concurrent Systems. In this paper, we first present the benefit of teaching parallel and distributed concepts, and then we will discuss our focus on covering of PDC topics in this course, evaluation of our efforts, and our plan for the course's future directions.

2 Why teaching parallel and distributed programming concepts

The era of a single-processor computer working independently is history. Still, a majority of concepts we teach our Computer Science students imply some form of necessary sequence. For example, in the definition of “algorithm” we use expressions like “set of ordered steps” [Bro2009], “step-by-step”, or “detailed sequence” (www.dictionary.com) that imply some form of necessary sequence must be maintained. For most schools like ours, majority of algorithms introduced to our students in their first few years of college are sequential. It is true that many steps of an algorithm have to be executed in a certain order, and the order of operations is critical to ensure the correctness of the algorithm. Still, for many students, the concept of concurrency may have been introduced in very general discussions, solving a problem with a parallel algorithm and implement such a parallel algorithm is generally not required until in their senior years in a parallel programming class. For students never take a parallel programming class, we can only hope that they are exposed to the concurrency concept and are trained in implement parallel algorithms somewhere soon because many students involved in software engineering are likely to be involved in high-speed computing at some point in their career. This is because the speed of light will prevent the speed of single processor computers to double in every 18 months forever. On the other hand, many computational problems require computers with speed many times faster than the fastest single processor computer today. Clearly, it is essential for today's Computer Science students to learn parallel algorithms because parallel computers, in the form of multicore, have more computation power and thus performance potential. Even students who will never participate in high-speed computing projects will soon find themselves in a computing world with software executing parallel algorithms or working on concurrent devices, such as clusters, multi-core computers, or multiple instances in Cloud. We believe that introducing the concepts of parallel algorithms and parallel programming in colleges can be beneficial to all our future computer.

2.1 Encourages students to think “outside of the box”

A typical question a parallel programmer is asked is “could you do better if you had more processors?” Generally, the problems of interest are clearly defined and have many sequential solutions. Now, can we find a parallel algorithm, or at least, in the existing algorithms, can we identify steps that can be carried out concurrently? If the answer is yes, how can we tell the compiler to do so? Are there any synchronization steps we need to introduce? To answer these questions, students have to have a deeper understanding of the problems and algorithms.

For example, finding the largest element in an array of size n sequentially has a complexity of O(n). After several lectures in a parallel processing class, most students can either give a theoretical O(log n) algorithm using n/2 processors, or a concrete solution using a fixed number of P processors to achieve speedup of roughly P. Now the question “could you do better if you had more processors” forces them to think further. Once so often, with a lot of hints, we would have a student or two come up with the algorithm of complexity O(1) using PRAM, a theoretical parallel computer, with n2 processors that support concurrent read and concurrent write and common write operations. After we discussed such an algorithm, students learned that critical section can be an important opportunity in designing parallel algorithms. This is a typical example where “thinking outside the box” is demonstrated and is strongly encouraged in parallel programming classes.

2.2 Revisiting many concepts discussed in previous classes

Studying parallel processing and parallel programming can provide a great opportunity for students to review concepts learned in other classes and to apply these concepts to solve new problems, as well as learn new algorithms to solve old problems. Generally, each of the Computer Science classes covers a well-defined subject area. Few classes offer opportunities to combine several well established subject areas together. We all know well that real-life problems hardly ever fit into a single subject area. Parallel programming classes not only introduce many innovative ways of solving well known problems, they also combines many Computer Science subject areas, such as programming languages, compilers, computer organizations, algorithm design, performance analysis, application development, communication networks, computational theory, and Operating Systems etc.

2.3 Sharpening programming skills

Parallel programming classes introduce many completely new algorithms with very unique way to approach a problem. Bitonic sort is such an example. Bitonic sort sorts an array of size n in O(log2 n) time using n/2 comparators (simple processors that sort two numbers) [Qui1994]. Coding this algorithm is an excellent programming exercise, where students learn the value of the design and analysis of the algorithm. The list-ranking algorithm, finding the ranks of elements in a linked list, can appear to be simpler. However, truly understanding it requires the comprehension of many important concepts of parallel programming and data structure, such as data parallelism and representing a linked list using an array. In addition, the list-ranking algorithm shows once again the importance of choosing the right data structure. When using traditional “pointer” data structure, the list-ranking algorithm seems to be inherently sequential. However, when we represent the list using an array, an NC class algorithm to solve the same problem becomes available.

2.4 Learning to look at problems from very different angles

Because parallel programming is a relatively new discipline, students are exposed to many new findings. Some of which seem to contradict to each other. This benefits students by demystifying computer science researches and teaches students that studying the same problem from different perspectives can result in very different results. Subconsciously, students develop the ability to think independently and to question well-accepted findings.

One such example of appeared contradicting results is found in the study of the speedup ψ of a parallel algorithm. Amdahl’s Law states that $\psi \leq \dfrac{1}{f+(1-f)/p}$, where f is the fraction of operations in computation that must be performed sequentially, and p is the number of processors. On the other hand, Gustafson-Barsis’ Law states that $\psi \leq p + (1-p)/s$, where s denotes the fraction of total execution time spent in serial code. According to Amdahl’s Law, even if we have infinitely many processors, the speedup is limited to 1/f, i.e. $\psi \leq \dfrac{1}{f}$. However, according to Gustafson-Barsis’ Law, when the total execution time spent in serial code is very small compared to the total execution time spent in parallel code (which is generally true for applications suitable for parallel computers), the s is insignificantly smaller compare to the parallel portion of the algorithm. In addition, as the problem size increases, the s becomes even smaller. Therefore, the speedup is only bounded by the number of processors, i.e. $\psi \leq$ p. Which begs the question, which one is correct? The answer is: BOTH!

Amdahl’s Law answers the following question:

> If an algorithm takes t time on a sequential computer, how long does it take for the SAME problem to be solved on a parallel computer with p processors?

Gustafson-Barsis’ Law considers the fact that we use parallel computers to solve larger problems. It answers the following question:

> If parallel algorithm takes t time on a parallel computer with p processors, how long does it take a sequential

computer with the same type of processor to be solved the same problem?

When looking at the same problem from different angles, we sometimes reach different conclusions [Qui2004].

3 The topics covered in our Concurrent Systems class

Our division's CS 474 -- concurrent Systems is offered every Winter term and is a strongly recommended elective class for students concentrating on software engineering and is designed to study parallel architectures, parallel algorithms, parallel program theory, and parallel programming.

We first offered the course in the early 1990's when Sequent Computer Systems donated two of their Symmetry with 20 processors each. These two machines were retired when a single CPU could outperform all 20 processors combined. We then introduced a Biowulf cluster. In the recent years, students have mainly programmed on Intel's multi-core and IBM's Cell processors on a PS2. This year, we added programming on multi-core GUPs using NVIDIA Tesla C2075 video card. Through coding, students gain firsthand experience on concepts such as process communications, data dependencies, scheduling, load balancing, locking, synchronizations, and effects of granularity. The class also covers many PDC proposed topics such as parallel computer organizations, parallel algorithms on PRAM, parallel programming concepts, and parallel sorting algorithms

For parallel computer organizations, we cover mesh, hyper-tree, butterfly, hypercube, hypercube ring, Shuffle-Exchange networks, etc. The discussion of parallel algorithms on PRAM is an important step to encourage students to think “outside of the box.” In this part, we introduce algorithms such as finding the Max in a constant time, fan-in, parallel prefix sum, list ranking, and parallel merging, etc. Students are exposed to many important parallel programming concepts throughout the course. Some of the important concepts are shared memory vs. distributed memory, message passing, speedup, cost of parallel algorithms, NC and P-Complete classes, Amdahl’s law and Gustafson’s Law, barrier and semaphore synchronizations, data parallelism vs. control parallelism, and Brent Theorem etc. We also introduce several parallel sorting algorithm and ask students to implement for two reasons: many parallel sorting algorithms are very unique in approaching the problems and implementing some of the sorting algorithm, such as Bitonic sort, can be an excellent parallel coding exercise. The sorting algorithms discussed are Bitonic sort, parallel quick sort, and random sampling, etc. In addition, students have term projects and several programming assignments to practice many key concepts and algorithms discussed in lectures. Currently, we are mostly using Microsoft's Visual Studio Task Parallel Library.

The programming component was done using C# by calling the APIs of Task Parallel Library accessible through Microsoft's Visual Studio 2010/2012. The first example on using the APIs is the matrix multiplication problem. We start by providing the sequential O(n3) version for solving the problem. Figure 1 shows the CPU utilization over time on a computer with an Intel i7 920 processor that supports eight concurrent threads. From the figure we can see that the task of multiplying the matrices is moved from different threads, i.e., to different cores. More importantly, at any given time, only one thread is working on solving the problem.

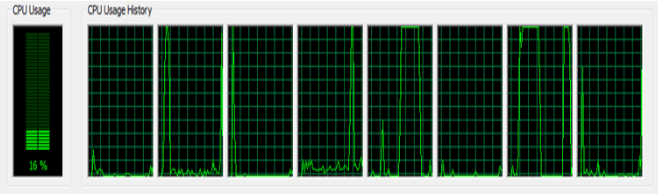


Figure 1. CPU utilization on sequential matrix multiplication.

Figure 2 shows solving the same problem on all the cores using TPL's Parallel.For API. The exciting moment always comes the CPU utilization reaches 100% as shown in Figure 2, only the top of each thread show a green line.

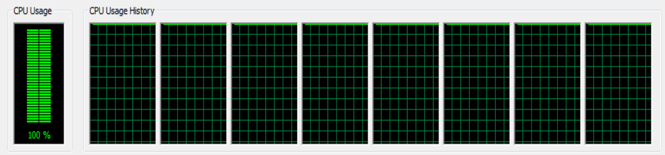
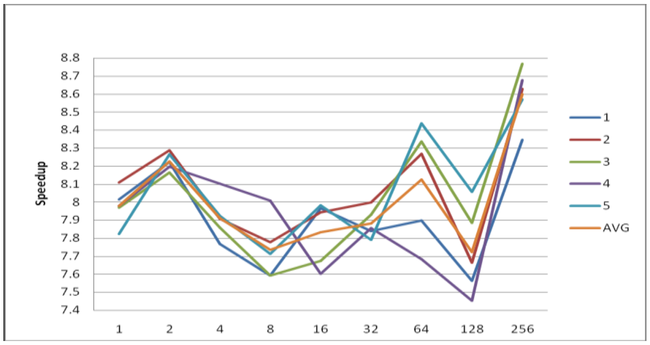


Figure 2. CPU utilization on parallel matrix multiplication.

Figure 3 shows the speedup of matrix multiplication of 2048X2048 matrixes with different chunk sizes. The experiments show consistently that the performance is the best when the chunk size is at 256, that is, each processor is given all the work of its share.



Note that we have achieved super linear speedup. We believe one if the main reasons for this is because the sequential version of the algorithm is relative speaking inefficient. In Figure 1, we can see that the task is assigned to different processors. This hurt the caching of both the code and data. With this example, not only we showed student how to introduce concurrency in algorithms and how to implement parallel programming concepts, we also introduced several important concepts such as data parallelism vs. control parallelism, speedup, granularity, scheduling, and load balancing.

A key component of our class is the required term projects. Right before the midterm, students are each assigned an individual project related to parallel processing. Students are allowed to find their own suitable projects when possible. The projects help students' learning in many ways. First, students need to conduct research either to learn some new tools or to find parallel solutions to problems with well define sequential solutions. Second, students have to implement the parallel algorithms or to experiment with the new tools. Third, students have to give presentations. This not only sharpens their presentations skills, which is an important career preparation step, but also allows students to learn from each other on a wide variety of topics outside the text books. Some of the topics students currently working on are advanced GPU Programming, Parallel programming using F#, Parallel Processing in the Cloud -- the current state, Parallel Sorting by Regular Sampling, Parallel Gaussian Elimination, and False Coloring. Outstanding projects have been selected to be presented in school's Academic Excellence Show-case to award student's excellent academic achievements and to demonstrate the abilities of our students to their peers around campus, professors of other disciplines, and school administrators.

4 Evaluation and the survey

We are using several means to evaluate the effectiveness of integrating PDC topics and about our teaching. One measurement is how many projects being selected to present in the Academic Excellence Showcase. We also survey students to find out what they think about the class.

Out school's Academic Excellence Showcase, organized by the Honor Society of Phi Kappa Phi and the Program for Undergraduate Research Experiences, is a one day event dedicated to presentations of student's scholarly activities. Selected students in each division demonstrate their achievements. Generally, a large proportion of students in our Concurrent Systems course are selected to show their course projects. The Showcase serves several purposes. First, students are motivated when they know their projects are more likely to be selected. Second, the campus in large can assess our integration efforts and teaching effectiveness. And third, the process shows other students what students learned PDC concepts are capable of doing. The simple action of showing a 100% CPU utilization chart excites many juniors and sophomores and has actually attracted a few capable students to switch their majors to Computer Science.

Attached to student's final is an anonymous survey. The table below shows a subset of questions with student's answers in percentage. Clearly, student responses are overwhelmingly positive. Answers on questions #8 indicate students are honest when answering the survey. Answers to questions #10 are extremely encouraging, almost 90% students are glad they are taking the class.

Table 1 Internal Course Evaluation Result

Question	5	4	3	2	1
1. Every CS student should take this class.	37.5	50	12.5	0	0
3. The class discussed interesting topics.	75	25	0	0	0
8. We should have had more programming assignments	0	37.5	37.5	25	0
9. Bitonic sorting is an elegant algorithm and is very interesting	37.5	50	0	12.5	0
10. I am glad I have decided to take the classes	87.5	12.5	0	0	0

The table here uses a number scale for the survey answers due to space considerations. The actual survey uses Strongly Agree, Agree, No Opinion, Disagree, and Strongly Disagree in the place of 5, 4, 3, 2, and 1 respectively.

5 Evolving of the course to cover new concepts

Understanding the Cloud is a migration certainty, as well stated by Gartner's analyst Ken McGee, we are looking into introducing HPC in the Cloud to the class and already looking into research topics and issues of conducting HPC in the Cloud. For example, we have contacted HP Cloud Services to find out ways to create instances on the same CPU or the same motherboard whenever possible.

One of the main drawbacks on using multi-core processor is that the number of cores is still small. We are also looking into parallel processing on Intel's Knights Corner 3120A that has 60 cores. We are also looking into using GPU

6 Conclusions

We believe that teaching parallel-processing to Computer Science undergraduates is both necessary and can be extremely beneficial to their future growth and to the software industry. We argue that it should be a mandatory class like the Operating Systems and Databases

7 Acknowledgements

We would like to thank the Western Oregon University's and Adopters - NSF/IEEE-TCPP Curriculum Initiative for supporting this research.

8 References

[Qui1994] Quinn, Michael J. Parallel Computing: Theory and Practice. 2d ed. New York: McGraw-Hill, 1994

[Liu1995] Liu, J. and J Marsaglia, What Should Freshmen Know About Parallel Processing, in Proceedings of the 12th International Conference on Technology and Education, 1995

[Qui2004] Quinn, Michael J. Parallel Programming in C with MPI and OpenMP, 2d ed. New York: McGraw-Hill, 2004

[Bro2009] Brookshear, Glenn J. Computer Science – and overview, 10ed Addison-Wesley, 2009

[INT2010] http://software.intel.com/en-us/intel-parallel-studio-home

[JIB2010] http://www.axon7.com/flx/home/what_is_jibu/

Design of a Curriculum on Cloud Computing

Ram Krishnan and Eugene John
Dept. of Electrical and Computer Engineering
University of Texas at San Antonio
Email: Ram.Krishnan@utsa.edu and Eugene.John@utsa.edu

Conference: FECS '13

***Abstract*—In this paper, we provide an overview of a design of a course on cloud computing and candidate content of this course. Cloud computing is a set of pooled computing resources delivered over the Internet. The cloud delivers a hosting environment that does not limit an application to a specific set of resources. Depending on the platform, an application can scale dynamically and increase its share of resources on-the-fly. The cloud can quickly scale to thousands of servers to make resources available as they are needed. The goal of this curriculum development effort is to train our students in a technological area that is state-of-the-art which has high demand for personnel skilled in the same. The proposed curriculum will incorporate fundamental aspects of cloud computing that will invigorate the current and future students and set a path towards establishment of a string of new courses around this area: lower power design of large scale systems, efficient computer architecture, materials engineering, efficient resource sharing, computer networks and cyber security to name a few. We also provide an overview of various projects that could be performed using OpenStack, open-source cloud "operating system" software, as a learning platform for cloud Infrastructure as a Service (IaaS). The proposed course is intended for undergraduate students in computer science, electrical and computer engineering. This course is envisioned as a senior level tech elective that can also be cross-listed as a graduate course.**

***Index Terms*—Cloud Computing, Curriculum, Education, OpenStack**

I. Introduction

Cloud computing [1, 10, 2, 5, 3, 12] is a set of pooled computing resources delivered over the Internet. The cloud delivers a hosting environment that doesn't limit an application to a specific set of resources. Depending on the platform, an application can scale dynamically and increase its share of resources on-the-fly. The cloud can quickly scale to thousands of servers to make resources available as they're needed. Application developers never need to worry about buying new hardware to meet increasing traffic demands or huge traffic spikes.

The goal of this curriculum development effort is to train our students in a technological area that is state-of-the-art which has high demand for personnel skilled in the same. The proposed curriculum will incorporate fundamental aspects of cloud computing that will invigorate the current and future students and set a path towards establishment of a string of new courses around this area: lower power design of large scale systems, efficient computer architecture, materials engineering, efficient resource sharing, computer networks and cyber security to name a few. While the traditional computer science and electrical/computer engineering departments have courses around these themes, the cloud technological context completely changes the ball game. Further, there is scope for integrating courses from other departments, possibly from other colleges. For example, effective infrastructure planning (college of architecture) is an important requirement for cloud. Clearly, there is an opportunity for establishing a major concentration area within computer science and electrical and/or computer engineering departments.

In the proposed curriculum, the students will be introduced to salient aspects of cloud computing using OpenStack, an open source platform for cloud computing. Students will build, configure, manage and run a small-scale cloud computing platform from scratch. The authors have built a cloud computing lab during summer 2012 at UTSA which has significant compute capacity, part of which has been devoted to this course. This will provide the necessary hands-on experience for students, which is highly valuable for a course of this nature. However, note that to learn the concepts, once does not need a heavy investment in infrastructure. OpenStack can be freely downloaded and installed within a virtual machine in free virtualization software such as Oracle VirtualBox [15].

II. Cloud Computing Curriculum Design

In this section, we provide an overview of our experience in designing a cloud computing curriculum at the University of Texas at San Antonio. We hope that by sharing our experience, instructors at other universities can adapt the content to their specific needs or improve upon the content and thereby proceed towards a mature curriculum on cloud computing in academia. Since the cloud computing industry has grown rapidly, it is critical that academia is able to train the next generation of workforce with knowledge in this area.

A. Cloud Computing Overview

The National Institute for Standards and Technology (NIST) of the U.S. Department of Commerce has provided one of the earliest characterizations of cloud computing [10]. We discuss this briefly below. NIST defines cloud computing based on 5 essentials characteristics, 3 service models and 4 deployment models. The 5 essential characteristics of cloud computing are:

- *On demand self-service:* Computing capabilities are accessible to consumers on demand and can be self-provisioned without the need to interact with a human.

- *Broad network access:* A wide-variety of client-side platforms can be used to access computing capabilities remotely over a network.
- *Resource pooling:* A computing capability service provider is able to pool its resources and offer it to multiple customers, with often-times a single physical server being shared between multiple customers.
- *Rapid elasticity:* Computing capabilities can be scaled-up and scaled-down automatically depending on customers' need.
- *Measured service:* Ability to provide a pay-as-you (or) pay-for-what-you-use) model for computing capabilities much like electric utility.

The 3 service models of cloud computing are:

- *Infrastructure as a Service (IaaS):* In IaaS, computing infrastructure capabilities such as storage, compute and networking are provided as a service by a service provider to its customers.
- *Software as a Service (SaaS):* In SaaS, various applications are provided as a service instead of a standalone desktop application. The client platform should be able to access these services using standard tools such as a web browser.
- *Platform as a Service (PaaS):* PaaS provides layered enhancements to IaaS and provides it as a service that can be used to build software that can utilize IaaS benefits. For example, specific application development platforms such as Ruby on Rails can be configured on computing infrastructure and offered as a service. The applications developed on such a PaaS may be provided with the capability to leverage IaaS benefits.

The 4 deployment models of cloud computing are:

- *Public Cloud:* In this deployment, the underlying computing infrastructure provided by the service provider is shared by multiple clients. This is by far the most economical deployment but clearly has a number of privacy concerns [17, 8, 9, 16] from the customers' point-of-view.
- *Private Cloud:* In this deployment model, the underlying computing infrastructure provided by the service provider is not shared by different clients. Each client has exclusive access to the resources. This could be more expensive but clearly has the least privacy concerns from the customers' point-of-view. Note that the physical location of the computing infrastructure might be at the service provider owned facility or within customer premises.
- *Community Cloud:* In this deployment model, the underlying cloud infrastructure can be shared between a community of clients that share similar concerns.
- *Hybrid Cloud:* In this deployment model, the underlying cloud infrastructure can be deployed using more than one deployment model above.

B. Curriculum Design Tasks

The following is based on our experience in developing a cloud computing course in the Department of Electrical and Computer Engineering at the University of Texas at San Antonio. The proposed course has a built in lab component. There are 5 major tasks to build this curriculum in a concrete manner as outlined below:

- **Theoretical Underpinnings:** The task here is to develop a concrete curriculum that encompasses theoretical aspects of cloud computing. As mentioned earlier, there are three cloud service models: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). In the simplest sense, IaaS provides virtual hardware, PaaS provides virtual software development platforms and SaaS provides applications as a service as opposed to typical stand alone installations in a computer. Further, there are three deployment models: public, private and hybrid cloud. Again, in the simplest sense, in a public cloud, the cloud infrastructure is open to the general public while possibly managed and operated by business, academic or government organizations. In contrast, in the private cloud, the cloud resources are restricted to a single organization. Hybrid cloud, as the name implies, is a composition of two or more public and private cloud infrastructures. This task will develop necessary contents to lay a solid foundation for following tasks.
- **Technological Underpinnings:** This task will develop a concrete curriculum that encompasses the key technologies in cloud computing. For example, Virtualization technology, where a virtual machine monitor exposes a single resource (e.g. server hardware) as multiple virtual resources, is a fundamental aspect of cloud computing. VMWare and Xen have leading solutions in this area. Another key technological aspect is architecture for cloud management. Eucalyptus provides open source platform for specifying cloud architectures. This task will develop course contents on virtualization using VMWare and cloud architectures using Eucalyptus.
- **Lab Infrastructure:** This task will develop necessary lab infrastructure for cloud computing that can be utilized by this course. As mentioned earlier, such a lab is already being developed for the purpose of cyber security research. We plan to leverage this lab for teaching cloud computing as well. The cloud will be made accessible from classroom workstations so students can manage the infrastructure remotely during typical lecture hours. The key task here is to build this infrastructure and devote a smaller component for this course.
- **Experiments:** This task will develop laboratory experiments for this course that provide hands-on experience on the theoretical and technological underpinnings of cloud computing. Experiments will be developed based on OpenStack, a open source platform for building and managing a cloud. Initially developed by Rackspace Hosting, a world-leading cloud provider head-quartered in San Antonio and NASA, OpenStack has been adopted by more than 150 major organizations including AMD,

Intel, Cisco, Dell, HP and IBM. Students will use OpenStack to build and manage a cloud. Specific programming experiments will be developed that leverage the power of cloud in dynamically scaling compute power based on demand. Students will configure public, private and hybrid clouds. They will perform experiments that will help them learn IaaS, PaaS and SaaS cloud deployment models.

- **Expert Guest Lectures:** This task will coordinate guest lecturers from local technology companies. Specifically, the majority of these lecturers will be technical staff with some offering business perspective. Rackspace Hosting and SiteB Data Services (a San Antonio based data center company) have expressed great interest in volunteering technical support in terms of guest lectures. The objective of this task is to plan and schedule such lectures and lecture contents from local technology-related companies if accessible. This is not a necessary task but highly recommended.

Cloud computing is a hot area. Offering a cloud computing course at the undergraduate level will have multi-level benefits and huge impact. Based on the authors' interaction with Rackspace Hosting Inc., a leader in cloud computing technology headquartered in San Antonio, they have expressed very high interest in hiring unoversity students who have appropriate skills. We expect that students doing this course will develop expertise in, among other things, OpenStack making our students highly employable. Since OpenStack is adopted by over 150 companies including AMD, Intel, Cisco, Dell, HP and IBM, the students will have a broad scope during their job search. There is a dearth of courses on cloud computing in academia. The timing is perfect. The academia should take the lead in developing this curriculum.

III. A Sample Course Outline

We provide an outline of content that can be covered during 13 weeks in the semester. Each week constitutes 2 hours and 30 minutes of lecture hours.

Week 0: Basic information technology and data center operations

Week 0 can give a practical overview of operations of a typical data center and an IT department in an organization. This will help the students understand how the computing infrastructure relates to various operations.

- Prerequisite quiz on basic programming concepts in C/C++/Java and basic programming constructs and syntax.
- Discussion of quiz solution.
- Introduction to traditional IT operations.
- Data center models and software service models.
- Assignment

Weeks 1 and 2: Cloud computing overview

The next 2 weeks can focus on introducing the benefits of cloud computing, its potential disadvantages, various characterizations of the cloud and security issues in the cloud. The goal is provide a theoretical overview of cloud computing and ensure that students understand how cloud computing is an evolution of IT operations and is less of a revolution.

- Introduction and the case for cloud computing
- Services in the cloud: infrastructure, platform, software and communication as a service
- Deployment models: public, private and hybrid
- Benefits and challenges in cloud
- Security issues in the cloud
- Assignment

Weeks 3 and 4: Programming overview

Programming is an essential skill needed to develop applications on the cloud. The goal of weeks 3 and 4 is to introduce students to programming languages and web application platforms typically used in the cloud. Example include Ruby, Python, Ruby on Rails (RoR) for rapid web application development, etc.

- Object oriented programming
- Basic Python/Ruby/Ruby on Rails programming
- Programming Assignments

Weeks 5 and 6: Virtualization technology

Virtualization technology is a key enabler of cloud computing technology. In particular, rapid elasticity would be infeasible without virtualization. The goal of these 2 weeks is to introduce virtualization to students using standard tools available such as Xen, KVM or VirtualBox. The projects can include simple set up of a Linux based operating system such as Ubuntu as a virtual machine within the regular workstation computer that students use. This is also a good place to introduce security issues in virtualization since multiple virtual machines can be hosted on a single hypervisor such as Xen and hence the virtual machines share the same physical resources. Vulnerabilities in the hypervisor can be potentially exploited by one virtual machine to steal data from other virtual machines.

- Virtualization overview and concept
- VMWare/Xen/KVM/VirtualBox
- Security and policy issues
- Assignment

The next important concept to introduce to students is architecture of a cloud. We can use open-source products such as OpenStack [14] or Eucalyptus [4] to introduce cloud architectures.

Weeks 7 and 8: Cloud architecture

- Basic architecture for cloud management
- Configuration options
- Experimentation using Eucalyptus
- Assignment

Weeks 9, 10 and 11 can focus on various cloud service models (IaaS, PaaS and SaaS). OpenStack is an excellent candidate that can be used to understand and experiment IaaS. For PaaS, services such as Heroku [7] provides a minimal free version to experiment web application development using frameworks such as Ruby on Rails on cloud platforms. Other examples include Google's App Engine [6] for developing

Google application and Microsoft Azure [11] for developing .NET applications.

Week 9: IaaS

- IaaS using OpenStack
- Experimentation using OpenStack
- IaaS using Joint
- Experimentation using Joint
- Assignment

Week 10: PaaS

- PaaS using Eucalyptus architecture and OpenStack
- Deployment of various platforms for application development
- Assignment

Week 11: SaaS

- SaaS using Eucalyptus architecture and OpenStack
- Deployment of various platforms for application development
- Assignment

An important characteristic of cloud is its elasticity, i.e, teh ability to program scaling-up and scaling-down as needed. IaaS software such as OpenStack by itself does not provide such a functionality. It only enables elasticity. It is upto the customer to engineering this feature. The last week (or more) in the course can focus on cloud elasticity. One option is to experiment with Redhat's free auto-scaling PaaS called OpenShift [13].

Week 12: Cloud Elasticity

- Programming applications for dynamic capacity scaling
- Experimentation using OpenStack
- Assignment

IV. Conclusion

Cloud computing is a rapidly growing area and it is important to recognize this emerging technology in academia and train students so they are well-versed with this technology when they graduate. In this paper, we provided a brief overview of a design of a curriculum on cloud computing based on our experience at UTSA. We provided an overview of cloud computing based on NIST's characterization of the cloud and outlined a set of tasks that needed to be carried out for designing a cloud curriculum. We then provided a sample content of a cloud computing course that could be offered over a period of a semester. This was based on a course that we developed at UTSA. Our hope is that by sharing our work, education institutions across the world can adapt and enhance our work to their particular needs. Further, this can lead to a standardized curriculum on cloud computing that will prepare the students well-versed in this technology at the time they graduate.

References

[1] Michael Armbrust et al. "A view of cloud computing". In: *Communications of the ACM* 53.4 (2010), pp. 50–58.

[2] Rajkumar Buyya, Chee Shin Yeo, and Srikumar Venugopal. "Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities". In: *High Performance Computing and Communications, 2008. HPCC'08. 10th IEEE International Conference on*. Ieee. 2008, pp. 5–13.

[3] Rajkumar Buyya et al. "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility". In: *Future Generation computer systems* 25.6 (2009), pp. 599–616.

[4] *Eucalyptus*. 2013 (accessed April 20, 2013). URL: http://www.eucalyptus.com/eucalyptus-cloud.

[5] Ian Foster et al. "Cloud computing and grid computing 360-degree compared". In: *Grid Computing Environments Workshop, 2008. GCE'08*. Ieee. 2008, pp. 1–10.

[6] *Google App Engine*. 2013 (accessed April 20, 2013). URL: https://developers.google.com/appengine/.

[7] *Heroku*. 2013 (accessed April 20, 2013). URL: http://www.heroku.com/.

[8] Meiko Jensen et al. "On technical security issues in cloud computing". In: *Cloud Computing, 2009. CLOUD'09. IEEE International Conference on*. IEEE. 2009, pp. 109–116.

[9] Tim Mather, Subra Kumaraswamy, and Shahed Latif. *Cloud security and privacy: an enterprise perspective on risks and compliance*. O'Reilly Media, 2009.

[10] Peter Mell and Timothy Grance. "The NIST definition of cloud computing (draft)". In: *NIST special publication* 800 (2011), p. 145.

[11] *Microsoft Azure*. 2013 (accessed April 20, 2013). URL: www.windowsazure.com/.

[12] Daniel Nurmi et al. "The eucalyptus open-source cloud-computing system". In: *Cluster Computing and the Grid, 2009. CCGRID'09. 9th IEEE/ACM International Symposium on*. IEEE. 2009, pp. 124–131.

[13] *OpenShift*. 2013 (accessed April 20, 2013). URL: https://www.openshift.com/.

[14] *OpenStack*. 2013 (accessed April 20, 2013). URL: http://www.openstack.org/.

[15] *Oracle VirtualBox*. 2013 (accessed April 20, 2013). URL: https://www.virtualbox.org/.

[16] Thomas Ristenpart et al. "Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds". In: *Proceedings of the 16th ACM conference on Computer and communications security*. ACM. 2009, pp. 199–212.

[17] Cong Wang et al. "Privacy-preserving public auditing for data storage security in cloud computing". In: *INFOCOM, 2010 Proceedings IEEE*. IEEE. 2010, pp. 1–9.

Teaching Mobile Computing—Curriculum Design and Strategies Applied

Xiaoyuan Suo
Department of Math and Computer Science, Webster University, Saint Louis, MO, USA

Abstract--*Mobile learning represents exciting new frontiers in education and pedagogy. Webster University has recently started offering courses for Mobile Computing. The designed curriculum focuses on practical experiences, which includes designing, creating, and testing mobile applications on mobile devices. In this paper, we report the curriculum designs, teaching strategies and tools applied; Major feedback also showed students were able to learn programming skills necessary to become more proficient in developing mobile applications.*

Keyword: Mobile Computing, Computer Science Curriculum, Teaching Mobile Computing, Curriculum Design

1 Introduction

Mobile learning represents exciting new frontiers in education and pedagogy. Mobile devices are ubiquitous: they encompass portable audio and video players, digital cameras, tablet PCs and PDAs, as well as cell phones and smart phones. A Sept. 2006 *Cellular News* story [1] estimated that there are more than 2.5 billion mobile phones worldwide. Although the mobile systems market is large and the career opportunities for students are excellent, relatively few universities offer mobile computing courses, much less teach mobile systems programming.[2]

Many schools have begun, or are considering, offering courses for programming mobile devices. A simple search on job recruiting website showed a dramatic increased demand in employment in mobile app development, mobile architecture, mobile design, or other mobile technology related field. A demand of education in mobile technology area has never been higher.

The Department of Mathematics and Computer Science at our university has recently proposed and started to offer a new B.S. degree in Mobile Computing. Students in our Mobile Computing major are focused on applying their technical skills to design, create, and test mobile applications. They will learn programming skills necessary to become proficient in developing mobile applications. These individuals are at the center of mobile development. They will be qualified to analyze, design, implement and test mobile applications as well as develop the required skills to maintain and update existing mobile applications. Graduates from this program will be able to apply their computing knowledge and technical understanding to move an organization into the mobile computing arena.

2 Background

Elon University and Appalachian University initiated a new Mobile program. [3] The report is rather comprehensive including both using mobile devices as teaching tools and teaching programming skills on Mobile devices. Course offered at Elon University used e-book, as supplemental reading material, along with Android Developers SDK. Students developed applications using Eclipse IDE and Android Development Tools plug-in. The course starts by lecture on the programming topics, and requires students to accomplish programming projects to fulfill the learning objectives. [3]

Appalachian state University offered a summer course with a combination of Android programming (80%) and iPhone programming (20%). The course carries out in a programming studio fashion and focuses on teaching problem-solving strategies [3]. Many other institutions have applied or used mobile technologies to enhance their teaching and students' learning experiences [4-9]. Work by Queens University of Technology in Australia [7] studied mobile learning outcomes, challenges and proposed guidelines for future mobile learning experience.

A few other institutions practiced using existing programming environment for teaching students building applications for mobile devices without programming background, such as App Inventor for Android (AIA) from MIT [10, 11]. In addition, educators have also converted traditional lab assignments on using mobile devices. For example, practices from Wentworth Institute of Technology has applied graphics programming on Android platform [12]. Android platform has shown its special advantage: it supports OpenGL, it uses open source development environments and it is easy to publish apps. Wellesley college also adopted AIA as a teaching tool for their courses for CS students [13].

West New England University focuses on development and maintenance of real-world mobile applications [13], since mobile applications development faces the challenge of developing for a rapidly evolving technology. Central Connecticut State University offered a mobile games course, which aims on providing students with a more exciting learning approach than pure coding. [13]

The mobile application development course offered at Nassau Community College in 2011. This is an effort of encouraging students at community college to complete their CS degree. [13]

Hampshire College offered a mobile computing course but with a focus on interface and application design rather than implementation [14]. The course used Cabana, which allows rapid interface design and application development, as platform. Students from various levels of programming background found Cabana as a easy adoptable software to help them rapidly develop and change mobile application designs. Technological limitations [15], such as lacking of mobile learning devices and resources, also makes mobile computing programs from carrying out world-wide.

Currently, there is a number of institutions offer a mobile computing related course, which can be classified into three categories [16]. First, mobile application development classes, students learn about building applications[17]. Second, mobile computing classes [18], students study mobile computing related issues and algorithms. Third, mobile computing client classes [19], students learn traditional CS concepts using mobile computing as a context for examples.

Earlier work summarized by Trifonova [20] showed limited capabilities, such as short time (5-10 minutes), simple functionalities, content domain specific, etc. Earlier works [21] also summarized mobile learning as an educational provision where the sole or dominant technologies are handheld or palmtop device. However, little work, which focused on teaching students how to program or learning technologies behind mobile devices, was found.

Based on ARCS learning model and mobile technologies' characteristics in promoting and enhancing human interactions, a variation to the ARCS model, the *Shih's Mobile Learning Model* (see Figure 2), was created to support instructional design for mobile learning. The learning cycle in the Shih's model includes: [22]

1. Sending a multimedia message to mobile phones to trigger and motivate learners
2. Searching the Web for relating information by using embedded hyperlinks (URLs) in the message received in the phone
3. Discussing with learning peers by text, voice, picture, or video messaging
4. Producing a digital story telling of what they learn by audio or video diary (mobblogging journal)
5. Applying what they learn in the simulated environment, such as online educational gaming

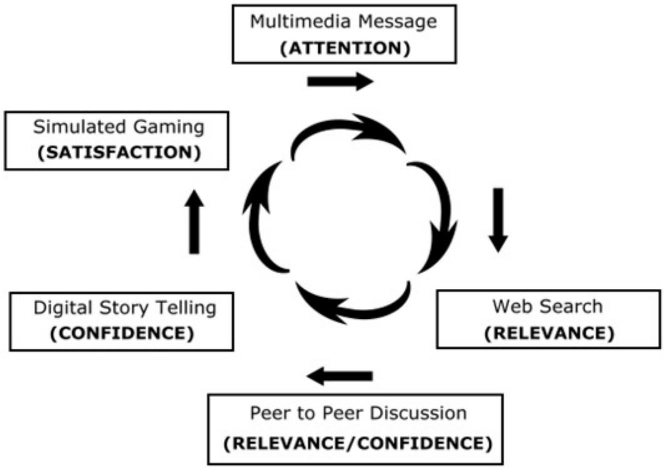


Figure: Learning Cycle in Shih's Mobile Learning Model [22]

Work by Kulik [2] developed a hands on lab which is designed for students to understand and develop programs for mobile devices. The lab covered many important aspects of mobile computing technologies, include: mobile development environments, interface design, mobile PANs, wireless infrastructure and other mobile network related techniques. The audiences are graduates anticipating a Master's degree of engineering in Distributed Computing.

3 Program Framework

Previous literatures studied the different subjects that should be included into the new mobile learning paradigm [23]. Mobile learning applications; mobile user infrastructures (such as browser, hand-held devices); Mobile protocol (adoption of content with WAP); and mobile network infrastructure (cellular systems, satellites, etc.) are the four proposed learning frameworks.

Our unique Mobile Computing program focuses on problem solving techniques, analyzing system component using mobile computing techniques, as well as building an in-depth knowledge of advanced mobile computing and development techniques.

Some of the required courses are listed in the table below.

Course Number	Course Name	Course Content
COSC 1550	Programming I with C++	Basic C++ programming foundation
COSC 1560	Programming II with C++	
COSC 1570	Math for Computer Science	Basic logics, and mathematical skills for computer science
COSC 2050	Programming with Java	Basic Java programming foundation
COSC 2060	Advanced Java	
COSC 2070	Introduction to Mobile Technology	Mobile technologies, mobile networks, basic mobile computing techniques
COSC 3340	Mobile Computing I	Programming skills on mobile platforms. Currently the content is programming for iOS devices.
COSC 3350	Mobile Computing II	Advanced programming skills on mobile platforms. Currently the content is programming for Android devices.
COSC 4510	Mobile Development I	Comprehensive understandings on mobile platform development
COSC 4520	Mobile Development II	Applications of comprehensive skills on mobile platforms
Other required courses for B.S. in Computer Science		

Table 1, required courses for our new Mobile Computing program

Specifically, each of the major mobile courses is discussed below:

COSC 2070 Introduction to Mobile Technology

This course studies the fundamentals of mobile technology. It focuses on emerging mobile technology, the potential of the mobile application market, and the technological and marketing challenges that make mobile applications difficult to commercialize. This course will also discuss the various tools available to build powerful mobile applications. Some of the tools we used in class include: PhoneGap[24], MIT App Inventor [25].

COSC 3340 Mobile Computing I

This course will study the leading-edge mobile computing technologies for professional software developers. This course is hands-on and project-based. The central focus of the course is to enable the understanding and critical evaluation of mobile applications.

Currently, the major component of the course is on designing and developing applications on the iOS platform. The course is primarily a programming course using Objective-C.

COSC 3350 Mobile Computing II

This course will focus on more advanced mobile computing techniques and mobile application development schemes. The central focus of this course is to further enhance the knowledge and critical evaluation of mobile applications and the mobile development process. We carry out the content by teaching programming with Java on Android platforms.

COSC 4510 Mobile Development I

This course aims to provide a greater depth of knowledge by studying the analysis and design process of mobile device computing. Topics include the available development tools, mobile development paradigms, device limitations, mobile application feasibility and economics, and the future trends of mobile computing.

COSC 4520 Mobile Development II

This course studies mobile development from three perspectives: mobile technology, application development, and user interaction. This course overviews various mobile applications, technologies and wireless communications. Students will learn about common paradigms in mobile development such as computing in environments with limited resources. This course will also study current research in mobile development.

4 Preliminary Feedbacks

The university curriculum committee originally approved the program in the spring of 2012. Three of new courses, Introduction to Mobile Technologies, Mobile Computing I, and Mobile Computing II were offered for the first time in the following academic year of 2012-2013. In addition, Introduction to Mobile Technologies was offered twice in the consecutive semesters; the class (18 seats) was fully enrolled for each time it was offered.

The responses we received were enormous. Most students reported they enjoyed the courses to a great extent. Some were able to study advanced mobile programming and other skills in addition to the course content. More specifically, among all students who attended the classes, about 20% of them have non-computer-science related majors. Majority of the students walked in with little or no knowledge with how to program or develop an app on mobile devices. At the end of the semester, we are pleased with most of students' work. In fact, four of the students were able to receive job offers from local mobile technology related businesses.

Some responses and suggestions from the students were incredibly interesting. Some students reported they walked into the classroom expecting to be "playing with an iPad", but ends up learning advanced programming languages such as Objective-C. Freshmen or students who had little knowledge in programming enjoyed learning MIT App Inventor [25], which they described as "an interesting way of learning programming skills behind interface designs". Some students suggested other topics, such as C#, Windows 8, and blackberry, to be included into the curriculum.

Nevertheless, we are already facing a few challenges even though the program is still at its infant stage.

1. Many students are more interested in learning the "cool" aspects of mobile technology, such as interface design, tools, and game designs. Fewer paid attention to fundamental skills such as debugging or testing.
2. The program is high maintenance. With the fast growing mobile world, it is not only costly in the getting new devices and software periodically, but also in learning new technologies and programming skills for both the instructors and students.
3. Many students, who are currently enrolled in our Computer Science program, questioned the necessity of switching their majors to Mobile Computing. While the new Mobile Computing program is attractive and innovative, we still need long-term studies to prove that it is as sustainable as a traditional Computer Science degree.

We believe, in a long term, this work will advance the understanding of engaging students into new degree programs, in order to prepare students for today's rapidly growing and changing computer technological jobs. The teaching materials and methodologies developed will help improve the students' problem solving skills. The project and the research will generate data and materials that can be beneficial to other universities with a similar degree program.

5 Conclusion

We are positively looking forward for another great semester of Mobile Computing offerings. Although there are currently 5 new courses in the program, we are thinking about adding other topics into the curriculum. We are also conducting long-term studies on the students' learning outcomes to further assess the sustainability of the program.

6 References

[1] http://www.cellular-news.com/story/19223.php, "2.5 Billion Mobile Phones In Use," in *Cellular News*, 2006.

[2] L. Kulik, "Mobile Computing Systems Programming: A Graduate Distributed Computing Course," *IEEE Distributed Systems Online*, vol. 8, p. 5, 2007.

[3] J. Fenwick, B. Kurtz, and J. Hollingsworth, "Teaching Mobile Computing and Developing Software to Support Computer Science Education," in *SIGCSE*, Dallas, Texas, 2011.

[4] J. Barbosa, R. Hahn, D. Barbosa, and C. Geyer, "Mobile and Ubiquitous Computing in an Innovative Undergraduate Course," in *SIGCSE*, Covington, Kentucky, 2007.

[5] D. Lim, "Ubiquitous Mobile Computing: UMC's Model and Success," *Educational Technology & Society*, vol. 2, 1999.

[6] S. L. Miertschin and C. L. Willis, "Mobile Computing in the Freshman Computer Literacy Course: What Impact?," in *CITC5 '04 5th conference on Information technology education*, Salt Lake City, Utah, 2004.

[7] R. Cobcroft, S. Towers, J. Smith, and A. Bruns, "Mobile Learning in Review: Opportunities and Challenges for Learners, Teachers, and Institutions," in *Online Learning and Teaching (OLT)*, 2006.

[8] R. Rieger and G. Gay, "Using Mobile Computing to Enhance Field Study," in *2nd International Conference on Computer Support for Collaborative Learning*, 1997.

[9] M. Berry and M. Hamilton, "Mobile Computing, Visual diaries, Learning and Communication: Changes to the Communicative Ecology of Design Students Through Mobile Computing," in *ACE '06 8th Australasian Conference on Computing Education*, Darlinghurst, Australia, 2006.

[10] H. Abelson, "Teaching CS0 with Mobile Appls Using App Inventor for Android," in *Seventeenth Annual CCSC Northeastern Conference*, Quinnipiac University, Hamden, Connecticut, 2012.

[11] H. Abelson, E. Mustafaraj, F. Turbak, R. Morelli, and C. Uche, "Lessons Learned from Teaching App Inventor," in *Seventeenth Annual CCSC Northeastern Conference*, Quinnipiac University, Hamden, Connecticut, 2012.

[12] M. Werner, "Graphics Programming on Android," in *Seventeenth Annual CCSC Northeastern Conference*, Quinnipiac University, Hamden, Connecticut, 2012.

[13] S. Jackson, H. Ellis, L. Postner, S. Kirkovsky, and E. Mustafaraj, "Mobile Application Development in Computing Curricula," in *Seventeenth Annual CCSC Northeastern Conference*, Quinnipiac University, Hamden, Connecticut, 2012.

[14] P. Dickson, "Teaching Mobile Computing Using Cabana," in *Seventeenth Annual CCSC Northeastern*

Conference, Quinnipiac University, Hamden, Connecticut, 2012.

[15] K. Gao, Z. Liang, and X. Ou, "Technical Analysis of Mobile Learning Application in High Education," in *The 6th International Conference on Computer Science & Education (ICCSE)*, SuperStar Virgo, Singapore, 2011.

[16] K. Sung and K. Foster, "Mobile Computing, Smartphones, and Existing Computer Science Classes," in *The First International SMArtphones in the Curriculum workshop (SMACK)*, Honolulu, Hawaii, 2011.

[17] V. Matos and R. Grasser, "Building Applications for the Android OS Mobile Platform: a Primer and Course Materials," in *Journal of Computing Sciences in Colleges*, Consortium for Computing Sciences in Colleges, 2010.

[18] F. Hu and A. Teredesai, "A Pervasive Computing Curriculum for Engineering and Science Students," *IEEE Pervasive Computing*, vol. 6, 2007.

[19] Q. H. Mahmoud, T. Ngo, R. Niazi, P. Popowicz, R. Sydoryshyn, M. Wilks, and D. Dietz, "An Academic Kit for Integrating Mobile Devices into the CS Curriculum," in *(ITiCSE)14th annual ACM SIGCSE conference on Innovation and technology in computer science education*, 2009.

[20] A. Trifonova, "Mobile Learning - Review of the Literature," University of Trento09 Apr 2003 2003.

[21] J. Traxler, "Defining Mobile Learning," in *IADIS International Conference Mobile Learning*, 2005.

[22] Y. E. Shih and D. Mills, "Setting the New Standard with Mobile Computing in Online Learning," *International Review of Research in Open and Distance Learning*, vol. 8, p. 16, June 2007.

[23] C.-H. Leung and Y.-Y. Chan, "Mobile Learning: A New Paradigm in Electronic Learning," in *IEEE International Conference on Advanced Learning Technologies*, 2003.

[24] PhoneGap, "http://phonegap.com/."

[25] MITAppInventor, "http://appinventor.mit.edu/."

INTEGRATING GAMES INTO THE COMPUTER SCIENCE CURRICULUM: FROM GENERAL EDUCATION TO THE GRADUATE LEVEL

Robert E. Marmelstein, Eun-Joo Lee
Department of Computer Science,
East Stroudsburg University
East Stroudsburg, Pennsylvania, USA

***ABSTRACT* - The use of game-based projects in the computer science curriculum has the potential to make the teaching of numerous computing courses more effective and interesting. In this paper, approaches are presented for using computer games to teach computer science at different educational stages: from general education through the graduate level. In each case, we examine how the game project supports the goals of the course. Potential pitfalls and lessons learned from these teaching experiences are also addressed.**

***KEY WORDS* - Computer Games, Artificial Intelligence, Computer Science Education and Curriculum**

I. Why Teach Computer Games?

For many of us, teaching computer game programming is not just a pedagogical exercise—it's also personal because we love recreating the games we play. Likewise, many students become motivated to become computer scientist as a result of the computer games they play as children and teenagers. Beyond the challenge of playing the game, the underlying technology that makes these games work fascinates many students. In short, playing these games provides the motivation to learn how to program them. While most students may never become professional game programmers, it's important for computer science educators to appreciate the power of the computer game as a mechanism for teaching many facets of the discipline. The following bullets illustrate the rationale for this assertion:

- Students may have experience playing a certain game—this personal connection increases their motivation
- The end product showcases the quality of work students are capable of producing. It can be readily demonstrated to parents, peers, or potential employers.
- The design, coding, and testing of most games can be very challenging. The experience forces the students to develop, utilize, and refine many different skills and processes.
- Incorporation of "play offs" can increase the motivation of students to innovate and excel.
- The process of writing a computer game often has a significant interdisciplinary component. It may require the student to employ skills in art, music, math, physics, and storytelling to varying degrees.
- Increasingly, games can incorporate technologies beyond graphics. In particular, they can make use of new input devices and sensors (such as Microsoft Kinect) which can recognize poses, movements, and speech. Additionally, game development may require web programming for multiplayer games, as well as, cross-platform development for mobile devices and game-specific platforms like the Xbox.

Obviously, there are many compelling reasons to integrate computer games into our curriculum. Often, this is accomplished by creating one or more classes specifically for computer game design. While this is the most direct approach, there are many other opportunities to integrate games into the curriculum at various levels. The remainder of this paper will explore how games might be incorporated at the general education, introductory, advanced undergraduate and graduate levels. Additionally, we will look at common pitfalls associated with teaching computer games and how avoid them.

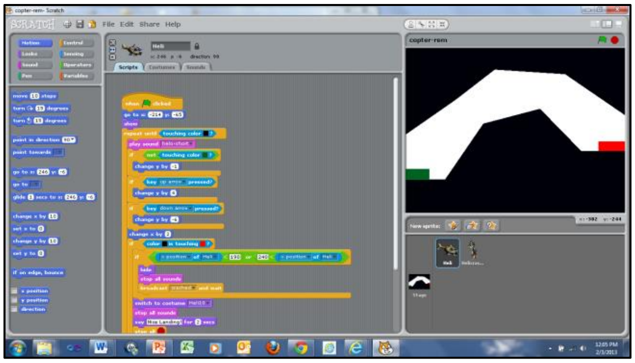
II. General Education Level

Convincing young people to study computer science is difficult for several reasons. Those who grew up with computers often come to think of them as mere appliances. Conversely, many students also avoid the subject due to the perceived difficulty of programming. If those students take a General Education (GE) course in computer science, it will probably have something to do with Microsoft Office or survey courses on information technology in general. However, incorporating computer games into a course can be one way to entice these students to try programming.

Recently, our Department launched a new GE course entitled “Games, Robots, and Intelligence”. The purpose of this course is to provide non-majors an introduction to using computers to solve problems that are both fun and interesting. As the title suggests, the course is divided up into three parts. In the robots part of the course, students use the Lego Mindstorms kits to build and program robots for simple tasks. The intelligence portion examines the nature of intelligence, both biological and artificial. Some of the issues and impacts associated with advanced in machine intelligence are presented and discussed.

For the games part of the course, students are taught to build simple games using the Scratch programming environment. **Scratch** is an interactive development environment which enables novices to quickly prototype games without having to learn syntactically correct code writing first. Created by the MIT Media Lab, it is intended to motivate further learning through playfully experimenting and creating projects, such as interactive animations, games, etc. [1]. Scratch scripts are constructed using functional building blocks that can be fit together to build larger, more complex structures. Each of these blocks represents different types of functions including: control, sensing, motion, looks, sound, pens, variables, and operators. These scripts are created for each individual sprite in the game. Each sprite is an object (character) that interacts with other sprites on a common stage; this makes Scratch an inherently object oriented environment. One of the most positive aspects of Scratch is that it enables students to quickly get a visual, interactive program working.

While Scratch makes programming easier and more intuitive, it does not make it effortless. In this course, students are guided, step by step through a sample Scratch program (see Figure 1) that makes use of all the relevant features. This “Copter” game is built up in stages: first the state is created, and then the sprite is imported. The initial sprite script allows the copter to take off and fly. Next sprite animation and sound are added. The next steps are adding the logic to detect either a successful landing or a crash. Lastly, the animation associated with a crash situation is added. There are many excellent books on Scratch for beginning programmers; the one currently used is “Scratch Programming for Teens” by Jerry Lee Ford. The Scratch.mit.edu website has a wealth of free resources, including help pages, video tutorials, and a downloadable gallery of over 3 million Scratch programs.



```
when [flag] clicked
go to x: -214 y: -65
show
repeat until <touching color [ ]?>
  play sound helo-short
  if <not <touching color [ ]?>>
    change y by -1
  if <key up arrow pressed?>
    change y by 6
  if <key down arrow pressed?>
    change y by -6
  change x by 2
  if <color [ ] is touching [ ]?>
    if <<(x position of Heli) < 190> or <240 < (x position of Heli)>>
      hide
      stop all sounds
      broadcast crashed and wait
    switch to costume Heli10
    stop all sounds
    say Nice Landing! for 2 secs
    stop all
```

Figure 1 - Scratch Copter Game

III. Introductory Programming

For many years, both the fundamentals of procedural programming and Objected Oriented Programming (OOP) were covered in our first class in our introductory programming sequence. Even though the OOP topics have since been moved to a follow-on course, we still teach Java as our first high level programming language. While Java can be used to do procedural programming, it is an inherently OOP language. Given this, the challenge is finding an introductory project which develops both the student’s procedural and OOP skills.

In order to develop these skills in our students, we had them develop a Blackjack game. Use of the Blackjack game project to teach OOP is an approach that other computer science educators have advocated [2]. Note that the game is not developed all at once; rather it is completed in stages over the course of the semester. The nominal sequence of stages is as follows:

- Stage 1 – Validate the string representing the value of a given card (e.g., “J-S” for Jack of Spades). Report an error if the string does not represent a valid card.
- Stage 2 – Instantiate a Card object from a validated string.
- Stage 3 – Create a Hand object to keep track of a series of dealt cards; as such, the Hand object is composed of Card objects. As each card is added, update and report the value of the hand (including if a bust has occurred). Note that the Hand may have a primary and an alternate value if it contains an ace.
- Stage 4 – Create a mechanism to deal the cards from a virtual Shoe object. The Shoe can hold four decks of cards and must be shuffled at the start of the game and when it runs low. Additionally, two classes are created for different types of game participants: a Dealer class and a Player class. For each game, the program will determine the winner based on the value of the players’ hands.

- Stage 5 – At the conclusion of a game, output the cards of each hand in sorted order.

As you can see from the above sequence, the first stage of the project largely involves procedural programming skills. However, with each stage, more and more OOP skills are introduced to the student. The last two stages of the project introduce the concept of arrays and sorting. Figure 2 shows a typical Java class diagram for the final version of Blackjack.

Unlike the Scratch game, this game need not make use of the GUI or other graphics/multimedia capabilities. Instead, input/output is text-based because the emphasis is on developing the student's procedural and OOP skills. The fact that this project is a card game that many of them are familiar with makes it fun, challenging, and rewarding. At the end of the class, the students have a readily familiar accomplishment they can show their parents and friends.

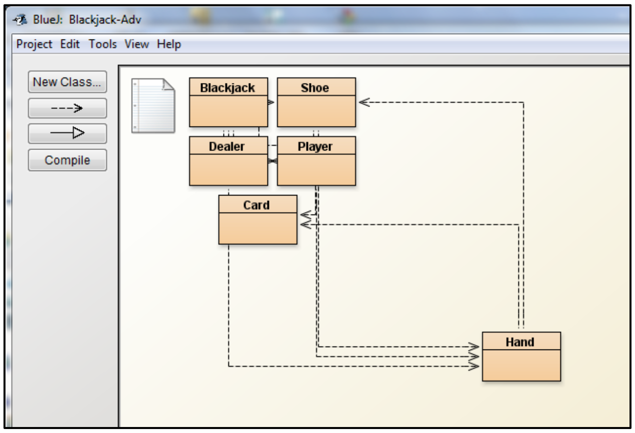


Figure 2 - Java Class Diagram for Blackjack Game

IV. Advanced Undergraduate

The next major opportunity to write computer games comes in our Computer Game Development course, which is an advanced undergraduate elective. In the latest iteration of this class, students are required to write two full scale game projects in C# using XNA game development library. They also complete several minor programming assignments which build up their skills in preparation for the major projects. Both major projects are 2D, arcade type games. While 3D graphics techniques are covered later in the course, they are not required for these projects.

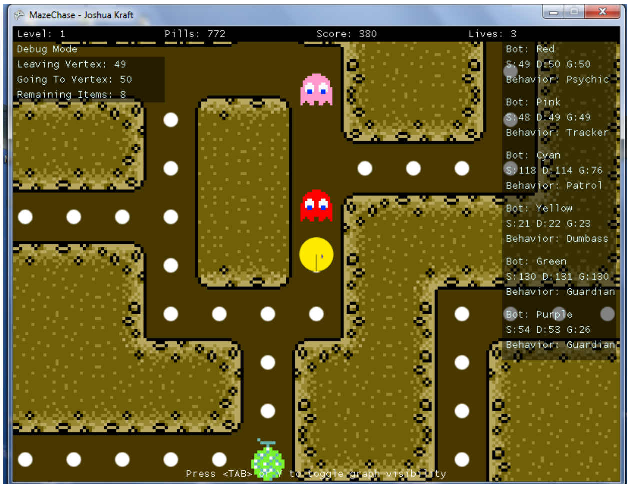
The first project is an "Asteroids" style game. This game is developed incrementally and due at midterm. The purpose of this project is to help the students master the basics of an arcade type game. The "Asteroids" game itself is challenging from a real-time programming perspective. The game software must create, move, and animate many different object types and their related interactions; some of the objects, like asteroids and missiles, have large populations which must be managed. Examples of the "Asteroid" game objects include: the player ship, asteroids (of varying size), missiles, explosions, and an autonomous alien ship. Likewise, inputs from the user must be processed without a noticeable delay and appropriate sounds must be played when certain events occur (e.g., explosions). The game also incorporates a fair amount of physics, including velocity, acceleration, and credible collision detection-and-response effects (e.g., conservation of momentum). To round out the experience, students are also required to implement a title screen and a high score screen as part of the game application.

The second project is a "Pacman" style game, due by the end of the term. This is likewise an arcade style game; however, the emphasis here is on constructing the maze environment, scrolling, navigation, and implementing the artificial intelligence (AI) for the bots chasing the player. The background maze is required to be much larger than the viewport (see Figure 3). When the player moves within the maze, the background must scroll to keep the player centered in the viewport, if possible.

The most elaborate part of designing the game, however, is the AI, including the behaviors of the bots [3]. The player can use the controls to guide his character through the maze; the bots must chase the player convincingly, without making it too difficult for the player to evade them. As a result, each bot must track the player in a unique way. For example, one bot may track the player by column; another bot may track the player based on the last intersection visited. Bots may also share information periodically in order to appear more intelligent and less predictable.

Once a given bot has an estimate of the player's current position, the bot must then plan a path through the maze from where it currently is to the player's potential location. Of course, over even a few game cycles, the player's position will change, so the bot must continually re-plan its route based on the most current estimate. Like "Asteroids", the "Pacman" game must run in real-time, so the continual planning that goes on cannot be allowed to slow down the game. As you can see, both games are challenging, self-contained projects that teach the students complementary sets of skills for real-time programming.

Additionally, we are currently supervising an independent study course where students are developing a "Battle Tetris" application that runs on Android mobile phones. This app will be distributed, such that students can compete against each other in real time when connected to the cellular phone network. If this course goes well, we will considering incorporating this material into a cross-listed undergraduate/graduate advanced games course.

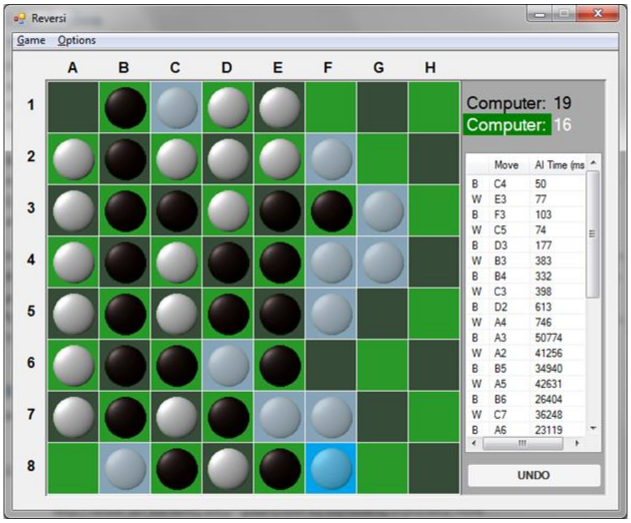


```
Level: 1          Pills: 772          Score: 380          Lives: 3
Debug Mode
Leaving Vertex: 49
Going To Vertex: 50
Remaining Items: 8

Bot: Red
S:49 D:50 G:50
Behavior: Psychic

Bot: Pink
S:48 D:49 G:49
Behavior: Tracker

Bot: Cyan
S:118 D:114 G:76
Behavior: Patrol

Bot: Yellow
S:21 D:22 G:23
Behavior: Dumbass

Bot: Green
S:130 D:131 G:130
Behavior: Guardian

Bot: Purple
S:54 D:53 G:26
Behavior: Guardian

Press <TAB> to toggle graph visibility
```

Figure 3 - Student Version of Scrolling "Pacman" Game

V. Graduate Level

At the graduate level, students build a Reversi game as the main project for the Data Structures and Algorithms course. In this game, the human is pitted against a computer (AI) opponent. Because a human is in-the-loop, the focus of the project is searching for the best move in a reasonable period of time (under 3 minutes). Due to the time constraint, the student must develop effective heuristic search strategies and limit search depth to maintain effective performance within the allowed time.



Computer: 19
Computer: 16

	Move	AI Time (ms
B	C4	50
W	E3	77
B	F3	103
W	C5	74
B	D3	177
W	B3	383
B	B4	332
W	C3	398
B	D2	613
W	A4	746
B	A3	50774
W	A2	41256
B	B5	34940
W	A5	42631
B	B6	26404
W	C7	36248
B	A6	23119

Figure 4 - Student Implementation of Reversi Game

The graphical user interface for a student-built Reversi game is shown in Figure 4. Most students implement the Reversi AI using a Minmax search algorithm coupled with alpha-beta pruning, as this is a well-documented approach [4] [5]. As a result, the key to designing the most effective computer player is the search heuristic. While some board positions are obviously more valuable than others (e.g., corners are considered the most valuable), the most successful students are those who use multiple factors to create a composite heuristic. Typically these factors include mobility (e.g., choosing a move that limits the mobility of the follow-on moves of the opponent), frontier (e.g., restricting the move choices the opponent has on the frontier), and stability (e.g., choosing moves that cannot be easily flipped).

Additionally, students are required to do a performance analysis of the time required to select a move based on the tree depth and move number (starting with the first move and ending with the last move). Figure 5 shows the chart generated from one such student analysis. Once the projects are complete, students participate in a "play off" tournament held on the last day of class. The student whose implementation wins the tournament gets extra credit—even more so if their implementation beats that of the professor. Of course, the biggest payoff for the student is the invaluable experience of building an AI algorithm that can beat a person at a strategic board game.

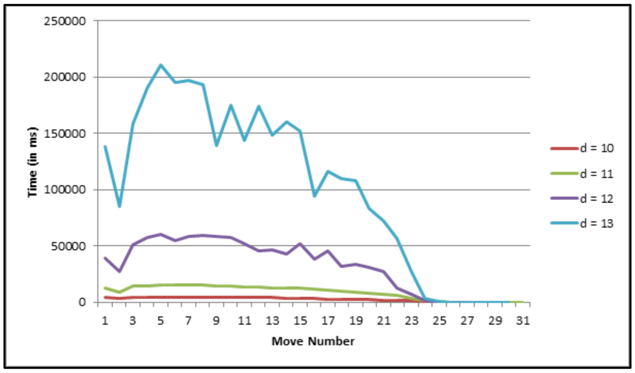


Figure 5 - Performance Analysis Chart for Reversi Game

In the computer graphics course, developing a game program has been utilized to enhance students' fluency and understanding of the course material. Two major learning goals are established for the students: acquiring fundamental concepts and skills, and developing various problem-solving strategies. Obtaining the fundamentals is essential to a students' learning process. In the class, students are expected to understand and implement all of the pieces of the graphics pipeline - model building, transformations, and rendering - in order to create an image. To develop students' problem-solving strategies, a final project of creating a game program has been employed, in which students were requested to solve various types of problems based on their knowledge and skills in computer graphics. Based on our experience, the process of developing a game program often helps to provide students with a deeper and more practical understanding of the subject.

Figure 6 shows a 3D Tetris game, developed by a graduate student in the previous class. The student expected it to represent a fully functional playable game. Also, some key concepts of computer graphics were demonstrated such as transparency, user interaction, orthographic views, double buffering, nested transformations and sub windows with window specific event handlers.

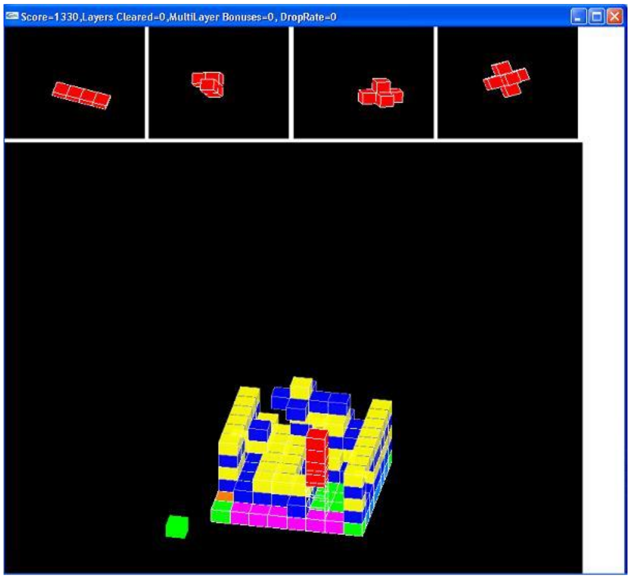

Score=1330,Layers Cleared=0,MultiLayer Bonuses=0, DropRate=0

Figure 6 - 3D Tetris Game

VI. Pitfalls

Despite the many positives of assigning game-related projects, there are also significant pitfalls as well. Most of these stem from student misconceptions about the nature, and difficulty, of the task. A fair number of students who sign up for a "computer games" course may expect to do more game playing than game writing—this is true even for programming courses. Even those that expect to work are often surprised at the level of difficulty associated with fully implementing an arcade type game. On some level, they may expect computer games to be as easy to write as they were to play. Because of this, it is good practice to make clear early on that the course will involve a significant amount of work; this helps weed out those who may be looking for entertainment vs. education.

Another issue with game programming is shifting standards and tool obsolescence. Our first Computer Game Development course used Allegro, a cross-platform game library, for C++ game development. While Allegro was well supported, there were issues with some of the support tools, especially the tile-map editor (Mappy) needed for maze design and scrolling [6]. The Mappy editor came with a playback engine compatible with Allegro version 4. However, when Allegro version 5 was released, there was no Mappy upgrade for the new version. This left me with three undesirable choices: find a replacement for Mappy compatible with Allegro 5, continue to use the outdated version of Allegro, drop my Pacman clone project, or migrate to another game library.

Ultimately, Allegro was abandoned in favor of XNA [7], a cross-platform, .Net game library for C# developed by Microsoft. However, that switch was rather painful as all my Allegro C++ code had to be converted to XNA C#. Additionally, with several scrolling tools had to be tested before finding one that worked in an acceptable manner; and even then, the selected application (TIDE) only had a fraction of the functionality that Mappy possessed. While the transition to XNA was successfully completed, Microsoft recently announced they will no longer support XNA. So, at some point, the whole process will have to be repeated for some new Game library. The moral of the story here is: be very careful with standards and the tools that you are depending on. Whenever possible, choose libraries and tools with the best support and greatest probability of longevity.

Another pitfall is the sheer complexity of the many game design tools. There are many different tools that can be used to develop games. While some of these are easy to use, many of the most powerful and flexible tools (like 3DSMAX and Maya) have very long learning curves. Even some of the simpler graphic editors, like Gimp, can take a while to master. For this reason, if the focus of the course is game programming, it's best to choose tools that deliver basic functionality without the learning curve.

Another challenge is trying to cover all essential topics in a single semester. There are many aspects of game programming that can be taught with 2D game applications. However, 3D games are increasingly the norm. The problem is that it's tough to combine both 2D and 3D game techniques in a single course without overloading the student. In the computer games course, the focus is on 2D scrolling and AI, vs. going into 3D techniques. The process for building 3D games is a conceptual sea change from 2D, especially in how objects are designed and rendered on the screen. 3D game programming inherently involves matrix creation and manipulation, making it far more mathematically complex than 2D games. While many game libraries have features to facilitate 3D graphics programming, it is still a non-trivial exercise. For this reason, it is preferable to separate 3D game development into a follow-on course. In our case, the computer graphics course serves as the follow-on course for learning 3D graphics.

Because of the difficulty of writing games, it is very important that the instructor write a functioning version of each game assigned. Even if it's a game that you've written before using another language or game library, it is always a good idea to convert your work to match the currently assigned configuration. Having a self-written version of every assignment becomes invaluable in guiding the students through the game development process step by step. Remember that any game implementation, no matter have seemingly trivial, comes with at least one “gotcha” characteristic. Without being able to assist them with the hard-to-implement features, many of your students may succumb to the frustration factor.

Lastly, it is essential to keep the students’ expectations of the course realistic. It’s important that they understand that a single game development class won’t make them a game writing superstar; rather, it is only a starting point. Like professional athletes, only a select few programmers ever get hired by top-flight game companies like Electronic Arts. The key to success is getting experience on your own writing even more complex games or involving yourself in open source game development. Successful game developers build up a visible, accessible portfolio which will eventually cause someone to take notice.

VII. Conclusion

Computer games are an excellent vehicle for teaching a wide variety of concepts within the computer science curriculum. However, they can also be very challenging; as a result, it is important to select the right game project and development environment for a given course objective. Luckily, with the number of game tools and environments continuing to expand, this is becoming easier than ever. At the introductory level, the introduction of the Scratch environment for teaching novice programmers is extremely exciting; it is also a potential vehicle for getting many non-programmers interested in computer science. In this paper, we’ve related our experiences with injecting computer games into the computer science curriculum at all levels. Hopefully, this can be used as inspiration or a guide for readers to do likewise.

References

[1] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, J. Silver, K. Brennan, E. Rosenbaum, B. Silverman, A. Milner and Y. Kafai, "Scratch: Programming for All," *Communications of the ACM*, vol. 52, no. 11, pp. 60-67, 2009.

[2] S. V. Kouznetsova, "Using BlueJ and Blackjack to teach object-oriented design concepts in CS1," *Journal of Computing Sciences in Colleges*, vol. 22, no. 4, pp. 49-55, 2007.

[3] M. Overmars, *Computer*, vol. 37, no. 4, pp. 81-83, 2004.

[4] J. L. Nielsen, "Reversi in Java," 2009. [Online]. Available: www.jonnielsen.net. [Accessed 2011].

[5] J. Sikes, "Creating a Reversi-Playing Agent Using a Combination of Minimax and a Value Array," Neon Zen, 2012. [Online]. Available: www.neonzen.com/Reversi_Paper.html.

[6] J. S. Harbor, Game Programming All in One - Third Edition, Course Technology PTR, 2006, pp. 477-585.

[7] J. Harris, "Teaching game programming using XNA: what works and what doesn't," *Journal of Computing Sciences in Colleges* , vol. 27, no. 2, pp. 174-181, 2011.

SESSION

RECRUITMENT AND MENTORING + METHODS FOR ENHANCING EDUCATIONAL ENVIRONMENT + TOOLS AND CASE STUDIES

Chair(s)

TBA

Android Control Application for Nao Humanoid Robot

Roslyn L. Brown, Heather L. Helton, Allison C. Williams, Michael T. Shrove, Mladen Milošević, Emil Jovanov
David Coe, and Jeff Kulick
University of Alabama in Huntsville,
301 Sparkman Drive, Huntsville, AL 35899

Abstract— *In this paper we present the Nao Controller, a first of its kind application to control the Aldebaran Robotics Nao Humanoid Robot. Nao Controller is the only Android application that has been successful in controlling the commands sent to a Nao Robot without using any sort of middleware for communication. Using packet capturing and executing replay attacks on the Linux machine inside the Nao robot allowed us to build unique movements into our Android application that allows a user to easily control the Nao robot by simply using Wi-Fi and an Android phone. The Nao Controller features many controlling aspects of the robot including Battery Status, Battery Safe Mode, Video Feedback from Nao's camera, Tilt Screen Mode, Manual Movement Mode, Saved Sequences, and a demonstration mode.*

Keywords: Nao Robot Controller, Android Smartphone, humanoid robot, Replay Attack, Nao Battery Monitor

1. Introduction

Smartphone applications are becoming increasingly popular means of use for general application purposes. The University of Alabama in Huntsville (UAH) purchased a popular humanoid robot known as Nao. To conduct research and demonstrations, Nao had great potential in making an impact on anyone who came to demonstrations. Throughout the research of Nao, one thing had been noted. The Nao robot could only be controlled with a computer. Any movement had to be programmed using Choregraphe or the NAOqi file [1]. This stemmed the idea of creating an Android application that would eliminate the use of a computer and make the robot more mobile and agile with creating and controlling movements performed by Nao. Once Nao became mobile, the possibilities of using him for demonstration and recruitment purposes seemed endless.

For this particular paper, we are going to outline the entire process that we used in the creation of our Android application, Nao Controller. First of all, we will explain features of the Nao robot and all of its capabilities. From there, we present other Android projects that are already in existence and being used to interface with the Nao robot.

Then, we describe the Nao development software, including Naosim and Choregraphe. We describe our approach to accomplish our Nao controller task. We had to discover how to use the NAOqi to coordinate with the robot and create our own movement packets sent to Nao. Next, we describe our application and capabilities that have been integrated into our final product. Finally, we discuss how the application has been used in accordance with our university, the University of Alabama in Huntsville.

2. Nao Robot

Aldebaran Robotics, a French-based company, came on the scene in 2004 with a new project known as Nao [1]. This humanoid robot developed into a product that rivaled upon the most sophisticated robots including Aibo, the robot dog developed by Sony [2]. In August of 2007, Nao won the Robot Soccer World Cup Standard Platform League beating Aibo [3]. Nao includes a Linux powered system that makes him very powerful. Along with his internal system, Nao comes with a suite of software used to control it on a user's PC including Choregraphe, which is a graphical programming tool [1]. Also, NAO came with a NAOqi file, which is used as the framework or API for the robot and can be used to program movements of the robot. Mainly used for research and educational purposes, universities have begun developing different capabilities of the Nao robot, such as to promote social engagement interaction with children who have Autism [4].

The twenty-three inch Nao robot is the first generation robot, Nao V3.2, featuring 25 degrees of freedom (DoF), and includes the following sensors: Ultra Sonic, Tactile\Touch, Microphones, Camera, FSR - Force Sensing Resistors, Gyro, and Accelerometer. For this version, the user can connect to Nao via Ethernet or Wi-Fi [5]. However, in the newest version, Nao V4 (NextGen), the user can even connect to NAO via Bluetooth, USB, or Infrared, in addition to the previously mentioned methods. All versions of the robot claim a 45 minutes battery life span when active, which we found was actually the case. [5]

3. Other Smartphone Control Applications

With Nao being so expensive and used mostly for research and academia, there are not very many Smartphone applications on the market that will control the robot. The two applications below were the only applications found that are used to control the Nao robot. These applications can be found on Robot App Store[5]. Even in these applications, it seems that each project only focuses on one or two highlighted features of Nao.

NAO Control [6] is an iPhone application that controls the NAO robot's movements using an on screen joystick. If the joystick is pushed up, Nao will move forward. The same act follows for backward. If the joy stick is pushed left or right, the Nao robot will shuffle left or shuffle right accordingly. In addition to movements, the application controls Nao robot's speech. The user can select from a list of predefined speech phrases or type text to send to Nao to speak. This application's user interface is very user friendly and the connection of the robot is automated meaning the user does not have to input anything, and the robot tells the user once he is connected to the phone.

NAO Server- Remote Control From Android is an Android Application that controls the Nao robot's movements using the tilt of the phone (accelerometer and magnetic field sensors) [5]. The biggest disadvantage of this application is that it has to use a "middle man" server to communicate between the Nao robot and the Android phone. The server is used because this particular application needs access to the NAOqi file to operate. The NAOqi is not able to be loaded to the phone, which means that it has to be accessed through the server. The NAOqi file is an APK that allows a developer to easily interface with Nao's movements [1]. To connect to Nao with this application, Nao states his IP address by pressing his center chest button. The user enters the IP address into the server application. Then, the user connects the phone to the server. From there, the user can choose to stand or sit the robot. If the robot is standing and the dead switch is pressed, the operator can move the robot forward, backward, turn left, or turn right according to the tilt of the screen.

4. Development Environment with Nao

Nao robot supports many applications including voice recognition, balance, video feed, face recognition, bumper sensors, and text to Speech [1]. Because of the vast features that are included in Nao, we needed to tailor our project to address just a few specific needs for our purpose as oppose to addressing all of the features Nao has to offer. As mentioned above in Section 3, most if not all other Smartphone applications only focused on one to two features of the robot. We have expanded this in our project to focus on many more aspects including the following: battery status, battery safe mode, manual mode, demonstration mode, tilt control with video feed, and saved sequences. After discovering the two projects described above in section 3, we determined that we wanted an application that actually receives information from the robot and combines needed functionality for quickly assembled demonstrations of Nao's capabilities. Before we could begin implementation of the Android application, we had to first get familiar with Nao and the software packages that came with it that were meant to control it.

4.1. Choregraphe

Choregraphe is the programming software that will allow Nao users to edit and create movements in a simple user interface (UI) [1]. The user can create a series of behaviors by dragging and dropping the predefined behaviors from the library, NAOqi. These behavior boxes are easily configurable allowing a user to develop a new movement not currently held in the library. In the application, the user can view the robot's position as they are giving him each movement. They can choose to connect to a Nao robot or to connect to a Nao simulation robot (more to be explained in section 4.2 below). If the user is using a Nao robot, the video feed of the robot can also be seen in the application. This application was used in our development to understand the behavior and interfaces of the Nao robot as well as learning to see how the movements were sent to the robot. Figure 1 is an example screenshot from the Choregraphe application.

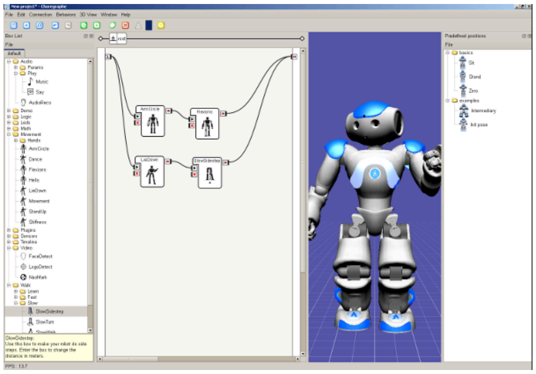


Figure 1: Choregraphe Nao Robot Software

4.2. NaoSim

NaoSim [1] is a simulation tool that was also part of the software package Aldebaran developed to support the Nao robot. NaoSim allows the user to launch a virtual world for the simulated Nao robot to navigate around as well as a virtual version of the Nao robot.

The robot in this environment mirrors the actual Nao robot, which means that if a new move is developed, the simulation Nao can perform that movement, and the user will see the reaction to that movement from the simulation. This protects the actual very expensive Nao robot from damage if the movement programmed does not go as planned. By watching the movement of the simulation, the user can determine if this was the intended movement and if the Nao robot will be able to execute that movement. Choregraphe and Naosim also connect with each other. This is how Choregraphe can use a simulation robot. Figure 2 shows an example of such a connection between Choregraphe and NAOsim.

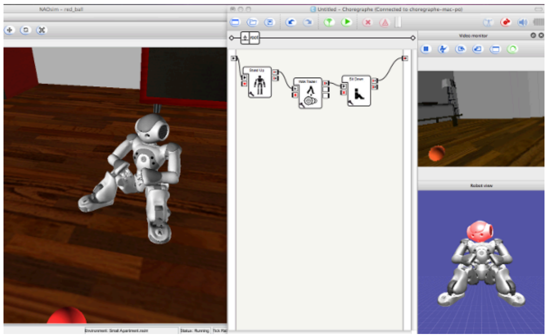


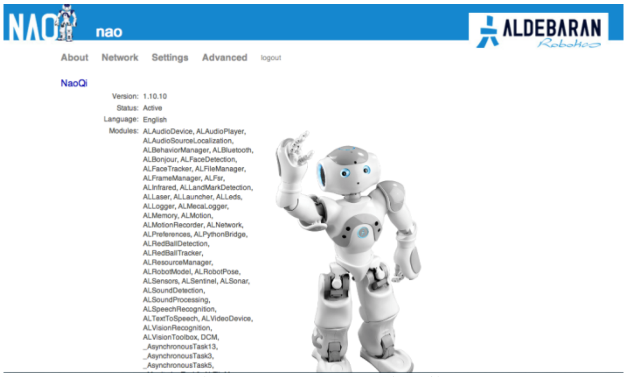
Figure 2: NAOsim and Choregraphe Connection [1]

4.3. NAOqi

NAOqi is the library file that is the API to the whole Nao robot. It runs on the robot's Linux machine and controls the commands of Nao. Every command the robot can execute will be in this library unless the developer creates an entirely new command sequence. For our project, we are currently using the library version 1.10.10. Although an older version, our robot is the first generation robot, and we felt the risk of upgrading him to possibly receive more functionality from a newer version of the NAOqi was too high. The developer needs to know the "language" of the robot, and this is the file that will allow the developer to communicate. The NAOqi Framework is the programming framework for the developer. The framework is cross-platform and cross-language [1]. In order for the user to run a simulation robot, this file must be installed on the user's computer. This file, as described in Section 3, was a large part in determining our design of NAO controller.

4.4. Nao Web Client

In addition to all of the software packages that have been described, Nao also has a web client as shown in Figure 3. This web client shows the user virtually anything they might need to know about the Nao robot. This is where the robot's version number, Ethernet and Wi-Fi address, joint positions, networks, and many other features can be discovered [1]. This web client became a necessity when we began our implementation of the application as will be described later on in Section 6.



Version: 1.10.10
Status: Active
Language: English
Modules: ALAudioDevice, ALAudioPlayer, ALAudioSourceLocalization, ALBehaviorManager, ALBluetooth, ALBonjour, ALFaceDetection, ALFaceTracker, ALFileManager, ALFrameManager, ALFsr, ALInfrared, ALLandMarkDetection, ALLaser, ALLauncher, ALLeds, ALLogger, ALMecaLogger, ALMemory, ALMotion, ALMotionRecorder, ALNetwork, ALPreferences, ALPythonBridge, ALRedBallDetection, ALRedBallTracker, ALResourceManager, ALRobotModel, ALRobotPose, ALSensors, ALSentinel, ALSonar, ALSoundDetection, ALSoundProcessing, ALSpeechRecognition, ALTextToSpeech, ALVideoDevice, ALVisionRecognition, ALVisionToolbox, DCM, _AsynchronousTask13, _AsynchronousTask3, _AsynchronousTask5,

Figure 3: NAO Web Client

5. Our Approach

After dealing with the user interfaces provided by Aldebaran and discovering that the possibility of having the NAOqi file on the phone to access was infeasible, we were faced with having to figure out how to send commands to the robot with just the Android phone, which had never been done before. Finally, it was determined that the best way to tackle this issue was to see what was actually being sent across from the computer to the Nao robot. We set up our environment with Choregraphe and a packet analyzer known as Wireshark [7]. This environment is described below.

5.1. Using NAOqi

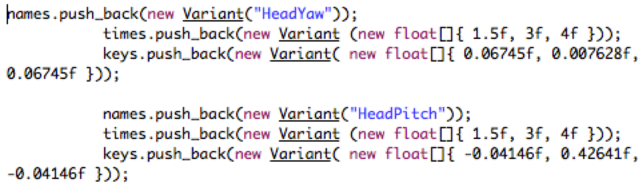
NAOqi had a library with it known as the jNAOqi.jar file that was used for the API. NAOqi is available in eight languages including java, and this was the Java library file [1]. Unfortunately, the NAOqi library for java was not compiled to run on a specialized virtual machine designed specifically for android devices, Dalvik [8], so we could not use the API to communicate to the robot through the phone. Since they do not provide the API specifically for the android platform, we created our own Java project and used the NAOqi library file to send command packets over TCP/IP to the robot with a PC. Then we sent the coded commands to the robot while monitoring the TCP/IP connection using Wireshark. Basically, we had to create a separate Java project that was specifically used for generating movements with the NAOqi file. The packets that are being sent are transferred over a TCP/IP connection using the Simple Object Access Protocol [9]. The Wireshark program was used to watch, capture, and strip the raw bytes we needed to resend in our Android application over in order to make the robot perform certain tasks. The following list contains the basic steps we used to create each unique movement sequence:

- Code what movement that needs to be created using NAOqi
- Run code while running Wireshark and observe the bytes sent
- Capture the bytes.
- If the packet size is too large to put directly into the android java files, serialize the bytes.
- Put the generated serialized file into the assets folder of the android program
- Reference the serialized file to pull in the bytes and send them over to the robot.

This process was repeated for each movement programmed into the Android phone.

5.2. Creating Unique Movements

It very quickly became apparent that we would need to create our own unique movements as well as the predefined movements already in NAOqi. Therefore, we used the NAO web client quite frequently to generate these movements. The web client has a Hardware section under the Advanced drop down menu. Here, the web client displays all 25 degrees of freedom and the position they are in at any given time. Consequently, we could hold Nao in the position we wanted to capture, take the values from the web client, and use them in our NAOqi Java application. Figure 4 is a code snippet of two joints, the HeadPitch and HeadYaw of Nao (as illustrated in Figure 5). As shown in the code of Figure 4, the keys represent the floating-point position of movement for that particular joint. Also, the times represent the time at which the Nao robot would be in that position. For this example, HeadYaw and HeadPitch were in key one at 1.5 seconds, key two at 3 seconds, and key 3 at 4 seconds. As shown, this could be quite labor intensive, especially when there are 25 degrees of freedom on the Nao robot. That means that we have to repeat these steps for each of the 25 degrees of freedom of the Nao robot. This contained a lot of trial and error for each unique movement, but our goal for the project was accomplished: create unique movements and send them from the Android phone.



```
names.push_back(new Variant("HeadYaw"));
            times.push_back(new Variant (new float[]{ 1.5f, 3f, 4f }));
            keys.push_back(new Variant( new float[]{ 0.06745f, 0.007628f,
0.06745f }));

            names.push_back(new Variant("HeadPitch"));
            times.push_back(new Variant (new float[]{ 1.5f, 3f, 4f }));
            keys.push_back(new Variant( new float[]{ -0.04146f, 0.42641f,
-0.04146f }));
```

Figure 4: Code Snippet of Joint Movements, Headpitch and HeadYaw

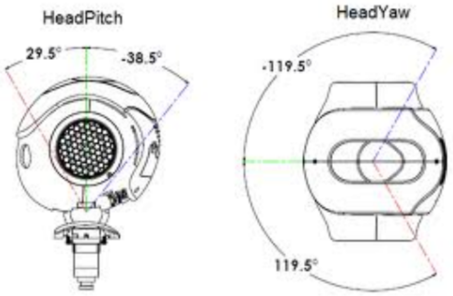


Figure 5: Example of Joint Movement [1]

5.3. Capturing and Sending Movement Packets

The most influential aspect of our application was the capturing and sending of movement packets. We could generate the movements with the NAOqi file all day, but if we did not know what was being sent or how to capture it, then our project would have failed. To capture the packets, we listened on Wireshark [7] for IP addresses communicating with each other (one from the computer and the other the Nao Robot). Each time we sent Nao a movement, we saw the packet go across Wireshark from our computer to Nao's IP address. Therefore, for any movement we programmed, we ran the movement and captured the packet. The captured packets would then be put into our Android application. We created a TCP socket to communicate with the Nao robot and send the captured packet over to Nao via TCP/IP protocol.

The captured packet was then saved. We used a replay attack [10] method where we resent the exact captured packets in the form of raw bytes through our PC using a TCP socket connection to the robot to make sure that he performed the same action.

Once we verified that the replay attack method would work, we then proceeded to add the packet bytes (either in raw form or serialized form if too large) used in our method to our Android application.

We created several different classes pertaining to the different parts of the body (NAOhead, NAOhand, etc). Then, we created several methods within each class that would be responsible for sending each movement. For example, the NAOhead class had a different function for moving the head up, down, left, right, and centering horizontally and vertically.

Within each function we stored the packet bytes into byte arrays. The arrays were either embedded into a function that was used to send the packet if small enough, or if the arrays were too big due to java method size limits of 65535 bytes [11], the arrays were serialized and read in.

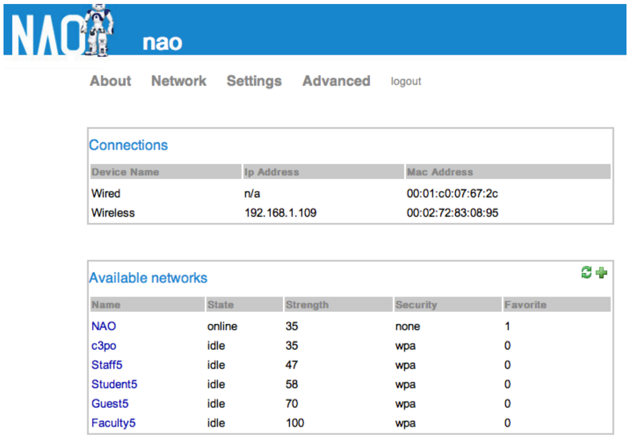
Then, we proceeded to send the stored packets over the network. For each packet we send, we had to wait for an acknowledgement from the robot before we proceeded to make sure that the command was actually sent and received correctly. First, we write to the server. Then, we wait for an acknowledgement to come back and read the first sentence of that acknowledgement. For some movements we are reading the entire message, but in this case we are just reading the first line of the response to make sure the robot received our message.

6. Nao Controller Application

As described previously, our Nao controller application contains many features that have never been developed for the Nao robot before. One of the biggest perks to the Nao Controller application is the ability to communicate with Nao through an Android phone without a server/computer setup to be the "middle man". No middleman is needed for our application. Before this application, the only android application to control Nao had to use the server setup on user's computer to communicate the movements to the robot. This setup almost made the android application completely obsolete since one of the main advantages to having an application on your phone is the ease of use and the portability of the application. The reason why the other Android application was in need of the server is due to the NAOqi file. The NAOqi as described above contains all the possible commands that can be send to the robot. It is the API that is responsible for every command pertaining to the robot. Early on, it was discovered that it was not possible to load the NAOqi file onto the Android phone and have our program access it which lead us to the approach we took in Section 5. Using this approach revolutionized our application and made our app the very first of its kind: the first Android application to solely control and interface with the Nao robot thus keeping the application portable to help with demonstration ventures for which the application was originally intended. In addition to only using an Android phone to control the robot, the application went a step further and decided to tackle many different features of the Nao robot, something that the preceding applications did not do. The following features of the application were developed keeping in mind the limitations of applications previously developed for Nao.

6.1. Connection Screen

The connection screen is used to enter the IP address of the Nao robot. The IP address is obtained by pressing the center chest button of the Nao robot. Nao will then say the IP address to which it is connected. If for some reason Nao is not connected to the Wi-Fi, the user can go to Nao's web client while it is plugged into the router with an Ethernet cord. From there, the user can select the available network for Nao and assign Nao a wireless IP address. See Figure 6 as an example.



NAO nao

About Network Settings Advanced logout

Connections

Device Name	Ip Address	Mac Address
Wired	n/a	00:01:c0:07:67:2c
Wireless	192.168.1.109	00:02:72:83:08:95

Available networks

Name	State	Strength	Security	Favorite
NAO	online	35	none	1
c3po	idle	35	wpa	0
Staff5	idle	47	wpa	0
Student5	idle	58	wpa	0
Guest5	idle	70	wpa	0
Faculty5	idle	100	wpa	0

Figure 6: Nao Web Client Available Networks Tab

The EditText itself that is found on the Connection Screen has an IP Address Regex [12], which is a Pattern Detection algorithm, behind the scenes. This is to ensure that whatever the user types into the EditText is actually an IP address before the program tries to make a connection to the robot. Also, if it is in fact an IP address, it will then need to be determined if it is Nao's IP address.

To accomplish this task, we send a ping out to the robot using the IP address entered by the user. If we receive a battery status back, we know that the IP address is in fact that of Nao, and the user of the application will proceed to the Main Menu screen shown in Figure 7 which contains all the functionality of the application. If we do not receive anything back, we have a time out, and the user receives a Connection Unsuccessful pop-up box.

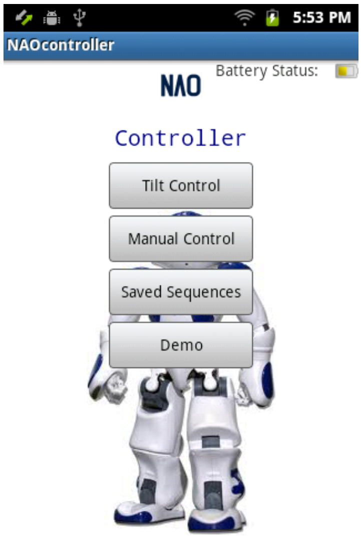


Figure 7: Nao Controller Main Menu

6.2. Tilt Screen

The Tilt Screen functionality was very similar to the Android project described above in section 3. The user can control the robot using the Gyroscope and Accelerometer sensors on the Android phone. The user must have one of the two dead switch button held down for the Nao robot to register the movement. As the user uses tilt to execute movements, the movements are displayed to the user via a Marquee bar. The battery status of the robot can also be seen in this screen, so the user of the application will always have a battery status of the robot while in Tilt operation. The battery status is updated every ten seconds. Therefore, the control aspect of the Tilt is much like that of the previous Android project with the addition of the Battery Status. However, our application went a step further by pulling the video feed from the NAO robot and displaying it to the user on the Tilt screen. Now, while the user is controlling the robot, they can also get up to date video feed from the Nao robot's camera.

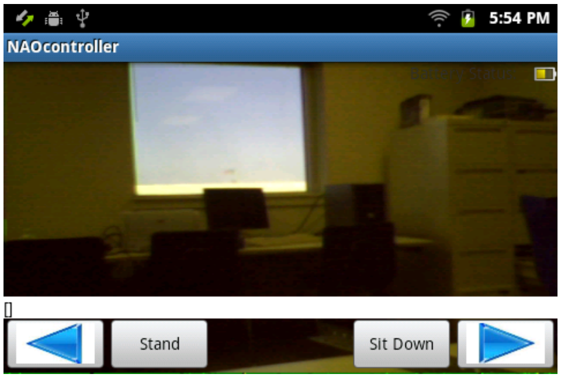


Figure 8: Nao Controller Tilt Screen

6.3. Manual Movement Screen

The manual movement screen was originally created just to test the functionality of each movement of the robot. However, it quickly became apparent that this Manual Movement screen would be useful if left in the application as well. In this screen, each of the Android's possible movements is displayed to the user in an expandable dropdown list. The top-level functions include Head, Arms, Legs, and Misc. The head function includes the following lower level functions: head tilt up, head tilt down, head turn left, head turn right, turn center vertical, and turn center horizontal. Arms has the following functions: close left hand, open left hand, close right hand, open right hand, right arm up, right arm down, right arm right (arm extended away from body), right arm left (into the body), right arm front, left arm up, left arm down, left arm right (into the body), left arm left (arm extended away from body), and left arm front. Legs have the following functions: walk one step forward, walk one step backward, walk backward eight steps, walk forward eight steps, crouch, turn left, turn right, sit, stand from sit, and stand from crouch. Finally, miscellaneous has the function to relax or stiffen the robot (cut on and off the servos of the NAO robot) and dance, which is the Gangnam style dance with the Nao robot playing the song through his speakers. Once the user selects a movement, the robot will perform the movement in near real time.

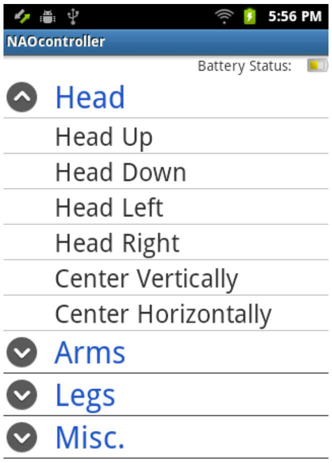
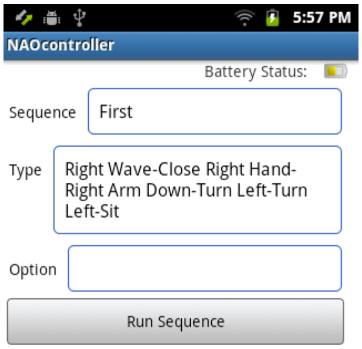


Figure 9: Nao Controller Manual Movement Screen

6.4. Saved Sequence Screen

In the Saved Sequence screen, the user has the option of creating their very own unique sequence using all of the possible manual movement controls described above. The user can choose to add a sequence, select all the movements they would like, and then save the sequence. From there, the sequence will show up in the list of Saved Sequences on the application. The user can choose to run the sequence, in which case the Nao Robot would perform each movement in series that the user has selected in that particular sequence. The user also has the option to delete or edit a sequence. This feature was added because of the feasibility of being able to quickly piece together sequences when needed. This would become great importance as a demonstration aspect. The user can queue up a list of sequences pertaining to that particular setting and execute those movements without ever having to use a laptop. There are some safety precautions put into place so that whatever sequence is entered, it will not allow a dangerous sequence to be created that could harm the robot. For example, the application will not allow the user to make the robot walk forward if the robot is sitting down.



Sequence: First

Type: Right Wave-Close Right Hand-Right Arm Down-Turn Left-Turn Left-Sit

Option:

Figure 10: Nao Controller View Saved Sequence Screen

6.5. Battery Status and Battery Safe Mode

As described earlier, the battery status of the robot is available on every screen where the user can control the robot's movements. The battery status symbol changes from green to yellow to orange to red depending on the battery level of the robot. Once the battery reads 10% to 20%, the user is alerted via a pop-up dialog box that the battery is getting low for the Nao robot. Now, when the battery is at 10% or below, the robot will go into Battery Safe Mode. In Battery Safe Mode, the robot will automatically sit down and will no longer receive any commands from the Android application. This prevents damage to the Nao robot if a sudden decrease in battery life could cause the Nao robot to crash during a movement. From there, the user must go and plug Nao back in to begin the charging process.

6.6. Demo Mode

Demo Mode was created as a "wow" factor to show a range of motions that the robot could perform. We wanted to create a fun sequence of steps that would be entertaining for potential new UAH recruits and visitors of UAH. The most familiar dance at this point would be something known as Gangnam style, a YouTube video gone viral. Therefore, we took the steps we described in section 5 and began mapping out the movements of the dance. The end result is a forty-five second dance of Nao performing Gangnam style while the song plays from his speakers [13].

7. Lessons Learned

The Nao application was created to provide an easier way to set up and control the Nao robot for demonstrative purposes. Our purpose for the application has already been proven to be a success at the Board of Trustees meeting [13] with the President of UAH including members of the University of Alabama, the University of Alabama in Birmingham, and of course, UAH. Nao performed his dance and then took a humble bow to conclude the demonstration. Most of the members of the universities present claimed they would love to have something like this demonstration at their school. In addition to this meeting, Nao made another appearance with the President at the UAH Foundation Committee meeting. With all of the praise and hype of using the Android application with Nao, we even have recruitment opportunities to take Nao to local high schools to promote Engineering and recruitment possibility at the University of Alabama in Huntsville.

8. Future Work

One aspect that could be improved upon is automatic connection to the Nao robot. Instead of the user having to press the center chest button of the robot and manually entering into a connection screen, the phone could get this information from the Nao robot itself.

Also, the Nao robot has text to speech functionality. This would be an incredible new feature to add to our application that would help with demonstration impacts.

The robot could always have more movements and demonstration modes added. We limited the Nao robot's actual range of movements due to time constraints and limited his demonstration functionality to just one dance. For the future, adding more interesting demonstrations and movements would be an overall improvement to the project.

In addition to these recommendations, since our project focused a good amount of maintaining the safety of the robot, a new feature could be a fall sequence that is designed to minimize the impact of a potential fall if Nao is in the middle of an action. A fall initialization algorithm could be put into place to accommodate such an accident to prevent further injury to the Nao robot [14].

Finally, using an updated version of the NAOqi on the robot could help with building newer functionality that is already included in the newer version of the file as well as testing our application with newer versions of Nao.

9. Conclusion

We developed a unique Android project that can be used to control the Nao robot directly, without any special equipment. The user only needs our application on the Android phone, Wi-Fi, and the Nao robot. Our application has accomplished so much more than any of its predecessors. With a lot more functionality, the users can do almost anything to Nao with our application including receiving video feed, battery status, battery safe mode, demonstration mode, and manual movements as well as creating custom movement sequences on the fly. Also, our application is the first of its kind to be able to control Nao's movements on an Android phone without the use of any type of computer or hardware "middle man" equipment. The elimination of such PC-based middleware equipment created a portable demonstration system with our Nao robot and Nao controller. With the increased amount of portability for demonstrations, we have discovered that not only is our application and the Nao robot impressive in itself, but the application is being used as a recruitment mechanism and demonstration robot to bring attention to the University of Alabama in Huntsville and the Electrical and Computer Engineering department.

Safety features were set into place with each aspect of our Nao Controller to prevent damage to the robot. With the tilt based movement control, the user receives video streaming from Nao's camera to view what the robot is seeing. The dead switches on tilt control allow the user to control Nao's movements only when these are pressed, creating a safety barrier for the robot. Another safety net is the battery monitoring that commands the Nao robot to sit before the battery drops to an unsafe level.

Our breakthrough with Nao Controller is the new way we capture movements through Wireshark. Using this capability, we are able to perform replay attacks on the Nao robot to repeat the movements sent which allows the ability to easily add more movements and demonstration capabilities to our application.

10. Acknowledgments

We would like to thank Andrew Cecil for his work with setting up the Nao environment and insight into his experience on working with the Nao robot. He created the initial development environment that served as a starting point for our project.

We would also like to thank Mr. Ray Garner, Director of Government Relations and Public Affairs of UAH, and Dr. Robert Altenkirch, President of the University of Alabama in Huntsville. These two individuals made our concept of using Nao robot as recruitment tool a reality. By having Nao perform for the Board of Trustees and the UAH Foundation Meeting, we promoted Engineering at UAH and the capabilities students could develop while they are being taught here at the University of Alabama in Huntsville.

11. References

[1] "Nao" Aldebaran Robotics. Last accessed 22 April 13. http://www.aldebaran-robotics.com/en/

[2] Aibo. Last Accessed 28 April 13. http://www.sonyaibo.net/aibostory.htm

[3] RoboCup. Last Accessed 28 April 13. http://www.robocup.org

[4] Daniel O. David, et al. "Children With Autism Social Engagement In Interaction With Nao, An Imitative Robot." *Interaction Studies* 13.3 (2012): 315-347. *Communication & Mass Media Complete*. Web. 29 Apr. 2013.

[5] Robot App Store. Last accessed 22 April 13. http://www.robotappstore.com

[6] "Nao Store". NAO My Apps- Help- Contact. Last Accessed 22 April 13. https://store.aldebaran-robotics.com/category/applications/

[7] Wireshark. Last accessed 28 April 13. http://www.Wireshark.org

[8] "dalviik.system." Android Developers. Last Accessed 28 April 13. http://developer.android.com/reference/dalvik/system/package-summary.html

[9] Web3C Recommendation. Last Accessed 28 April 13. http://www.w3.org/TR/2007/REC-soap12-part1-20070427/

[10] Replay Attacks. Last accessed 28 April 13. http://msdn.microsoft.com/en-us/library/aa738652.aspx

[11] Prashant Dava. "Method Size Limit in Java." DZone JavaLobby. 13 June 11. Last Accessed 28 April 13. http://java.dzone.com/articles/method-size-limit-java

[12] Regular-Expressions.Info. Last accessed 28 April 13. http://www.regular-expressions.info/examples.html

[13] Gattis, Paul. "'Gangnam Style': UAH students create app to teach robot the moves (video)". AL.com. 25 April 2013. Last accessed 28 April 2013. http://blog.al.com/breaking/2013/04/gangnam_style_uah_students_cre.html

[14] J. Ruiz-del-Solar, R. Palma-Amestoy, R. Marchant, I. Parra-Tsunekawa, P. Zegers, "Learning to fall: Designing low damage fall sequences for humanoid soccer robots", *Robotics and Autonomous Systems*, Volume 57, Issue 8, 31 July 2009, Pages 796-807, ISSN 0921-8890, 10.1016/j.robot.2009.03.011.

Evaluating a Prospective Student Mentoring Program

Amber Settle, Sarah Pieczynski, Liz Friedman, and Nathan Kizior
DePaul University, Chicago, Illinois, USA

Abstract - *Although student enrollment in computing programs is rebounding, there are still not enough students in the pipeline to meet future demand for computing professionals. Mentoring programs play a role in the retention of existing students, but mentoring of prospective students can help to bridge the gap between secondary and post-secondary experiences. We describe a long-standing prospective student mentoring program at a Midwestern university in the United States and share the results of a two-year evaluation of the program. We found that the program was more likely to influence the enrollment decisions of female students, that a majority of students were interested in continuing contact with their assigned faculty mentor, and that contact with a faculty mentor was correlated with positive enrollment decisions.*

Keywords: Student recruitment, student enrollment, student mentoring, student-faculty interactions

1 Introduction

Effective student recruitment has long been of interest to educators [5], and such interest is not restricted to U.S.-based institutions [3, 7]. This is particularly true in computing. Despite favorable job projects [9], the numbers of students enrolling in programs in computer science, information systems, and information technology programs is only beginning to recover [9]. Because at many institutions fewer students enroll in computing programs than would be desirable [9], retaining students takes on an increased importance. One approach to retaining students is to use mentoring programs. But mentoring programs are often not implemented until a student has enrolled at an institution [2, 4, 6, 8, 12] and then often to improve the retention of underrepresented groups [6] or of students in particular courses such as introductory programming [8]. This leaves a potential hole for prospective students, who may not have access to resources while they are making crucial decisions about college enrollment. One approach to this issue has been to design virtual environments for prospective students, but this approach has not been evaluated for its effectiveness [1].

The College of Computing and Digital Media (CDM) at DePaul University is one of the largest information technology institutions in the United States with approximately 1800 undergraduates enrolled in 14 Bachelors degree programs. CDM is unusual in its breadth of degree programs, and advising is crucial both for matriculated and prospective students. Matriculated students have long had access to both faculty and staff advisors throughout their career, and recent changes have been made to improve their experience [10]. Programs designed for prospective students are more challenging, in part because of the logistical issues. Since 2004, CDM has had a mentoring program that pairs faculty members with prospective students applying to CDM. The goal of the prospective student mentoring program is to allow students early access to faculty, with the hope that the information faculty provide about academic programs will help the students make good decisions about college enrollment and CDM in particular.

However, creating a prospective mentoring program is only helpful if that program is achieving the goals that motivated its implementation. Although CDM had a prospective mentoring program for many years, an evaluation of the program had never been conducted. In 2009 a project was begun to understand the effectiveness of the program, with a two-fold focus. First, we wanted to understand whether the program was impacting the decisions of prospective students. A secondary goal was to understand whether faculty were engaged and effective as mentors, since that is a prerequisite for useful mentoring program.

The evaluation project consisted of a series of surveys for three audiences: prospective students, current CDM students, and faculty participating in the prospective student mentoring program. Prospective and current students were asked whether the program had affected their enrollment decisions. Faculty were asked about their participation in and engagement with the program. The first year of evaluation resulted in poor response rates and quality of data from both prospective students and current students, although useful information was obtained from faculty participants [11]. The evaluation process and the program itself were modified slightly based on the faculty feedback during the 2010 – 2011 academic year, and surveys of prospective students and faculty were implemented again. The response rates and quality of data were better. We found that while student recall about the program was spotty and students reported less contact than faculty did, the program was more effective at influencing enrollment decisions for female students. A significant minority of students indicated that the program had influenced their enrollment decisions, and a majority of students were interested in continuing contact with their faculty mentors.

In the remainder of the paper we describe the CDM prospective student mentoring program, provide details about the surveys of students and faculty, analyze the data gathered, and discuss the results.

2 Mentoring program

As mentioned previously CDM has had a mentoring program that pairs faculty members with prospective students applying to CDM since 2004. The goal of the program is to provide prospective students with academic information that they would not otherwise receive from contacts with advising staff and also to improve the enrollment rate among prospective students.

The CDM undergraduate services staff assign students to volunteer faculty within a few weeks of their admittance, typically beginning in January of each year. Students are sent the contact information for faculty in a letter congratulating them on their acceptance to the university and in an e-mail following the letter two weeks later. Faculty receive a spreadsheet, typically twice a month, with mentees' information. Faculty are expected to initiate at least one contact with the student and to answer any questions the students may ask. Faculty are not required to contact the students in any particular way and are provided with each student's address, phone number, and e-mail address. In March there is an Admitted Student Day in which students and their parents/guardians are invited to come to DePaul for an open house. Faculty participating in the mentoring program are encouraged to come to the event, where they can meet with students and answer any questions they or their parents/guardians may have. The program is typically somewhat dormant after the March event, although students do still have the opportunity to contact faculty. May 1st is the national deadline for students to declare their intentions for university enrollment, and student-faculty contact associated with the mentoring program typically ends at that point.

While the mentoring program has existed in various forms since 2004, no systematic evaluation of the program was done until 2009. In the next section we describe the process used to evaluate the program over the two years of this project.

3 Evaluation

When approaching the evaluation of the prospective student mentoring program, we believed there were two major issues. The most important consideration is the effectiveness of the program with respect to students. Since a strong reason for the existence of the program is to recruit prospective students, our main measure of effectiveness is an increased likelihood of participating students attending DePaul. But there were other impacts of the program we considered, such as the possibility that the faculty mentor was a poor fit for the student. In particular, to evaluate the effectiveness of the program with respect to students we considered the following questions:

1. Are students with a particular major who participate in the program more likely to enroll at DePaul?
2. Are male or female students more likely to enroll at DePaul after participating in the program?
3. Are students who were contacted by faculty mentors more likely to enroll at DePaul?
4. Are students who contacted their faculty mentor more likely to enroll at DePaul?
5. Is a student who met with a faculty member other than their mentor more likely to enroll at DePaul?

To evaluate the impact of the program on students we surveyed newly enrolled students who had participated in the program. We also surveyed students who had participated in the program during previous years. Response rates for each of the surveys varied enormously and the results are discussed in the following section.

A prospective student mentoring program will not be effective without faculty who are engaged in the program. Therefore a secondary evaluation approach was through faculty surveys. There we wanted to understand how often faculty contacted students, how faculty contacted students, the frequency with which faculty answered student questions, and what topics were discussed during faculty-student interactions. The information from the faculty surveys is provided in the final part of this section.

3.1 Student analysis

The first year of the evaluation of the prospective student mentoring program was 2009-2010 and during that year there were two populations of students who were surveyed: newly enrolled freshman who had participated in the prospective student mentoring program in the six months prior to their enrollment and all relevant enrolled CDM students. Prospective students were surveyed after they had enrolled at DePaul, either in their first or second quarter. Only current CDM students who had enrolled since 2004 as a traditional freshman, and thus had taken part in the prospective student mentoring program, were included in the survey.

Unfortunately the response rates for both student groups were lower than expected in the first year of the evaluation project. The response rate for current students was 18% (64 out of 355). Further among the 64 responses, only a small percentage (18% or 12 out of 64) indicated that they had participated in the mentoring program, something that simply was not true. Our conclusion was that current students were unable to recall their participation in the program, a hypothesis supported by the comments provided by students responding to the survey. The response rate among prospective students during the 2009-2010 year was also low at 20% (66 out of 325). Although there were more useful responses than the current student survey produced, we felt that improving the response rate would be crucial in order to gain sufficiently reliable data.

During the 2010-2011 evaluation process current students were not surveyed at all since their recall of the program was deemed to be too weak to provide useful data. Further, the process for prospective students was changed. Students were surveyed earlier, at their orientation rather than during their first or second quarter after enrollment. Students completed an anonymous paper survey during orientation. As hoped this improved the response rate for the surveys, with 191 responses received. There were 211 students over the age of 18, for a response rate of 91%.

The following questions were included in the 2010-2011 prospective student survey:

1. Sex: Male/Female/Prefer to not specify
2. Age: [fill in the blank]
3. My intended major is: [selection from available items]
4. My assigned faculty mentor has contacted me: yes/no
5. I have contacted my assigned faculty mentor: yes/no
6. If you answered yes to question 4 or 5, indicate how often you have been in contact with your faculty mentor, regardless of who initiated the contact: weekly/monthly/at least once
7. I have been in contact with another CDM faculty member other than my assigned faculty mentor: yes/no
8. If you answered yes to the previous question, please indicate how often you have been in contact with the other CDM faculty member: weekly/monthly/at least once
9. I would be interested in having my faculty mentor be assigned as my faculty advisor: yes/no
10. The contact I had with my faculty mentor positively impacted my decision to enroll at DePaul: yes/no
11. The contact I had with my faculty mentor positively impacted my decision to choose a major at CDM: yes/no
12. The best things about being assigned a faculty mentor are: [fill in the blank]
13. The faculty mentoring initiative could be improved by: [fill in the blank]

In the remainder of this section we provide descriptive statistics for the responses from the prospective students and analyze correlations between various survey responses to answer our previously-mentioned evaluative questions. All data analysis was done using SPSS.

3.1.1 Responses

A vast majority of prospective students were male (82.2%) with the remainder female (17.8%), which is comparable to the overall CDM student population. The overwhelming majority of students were traditional age, with 94.8% indicating that they were 18 years of age, 4.7% 19 years, and 0.5% 20 years. The majors of the students participating in the program were proportional to the overall enrolled student population. The data about majors is not included here due to space constraints.

The following table indicates the responses to the questions about contact with the faculty mentor and the frequency of that contact:

Table 1. Contact with faculty mentor

Question	Yes	No	No answer
Contacted by faculty mentor	28.8%	69.1%	2.1%
Student contacted faculty mentor	17.8%	79.6%	2.6%

A total of 56 students answered the follow-up question, with 1 (1.8%) indicating contact weekly, 7 (12.5%) indicating contact monthly, and 48 (85.7%) indicating contact at least once.

When asked about contact with a faculty member other than their assigned mentor, 16.8% of students responded yes, 74.9% responded no, and 8.4% declined to answer. In the follow-up question, 0.5% indicated weekly contact, 2.1% monthly contact, 17.8% reported contact at least once, and 79.6% declined to answer.

When asked if they would like to have their assigned faculty mentor as their CDM advisor, 50.8% indicated yes, 18.8% indicated no, and 30.4% declined to answer.

When asked whether the faculty mentoring program had any impact on their enrollment decisions, the results were nearly evenly split between each response (yes, no, unanswered). The table below lists the responses for both questions:

Table 2. Mentoring impacted enrollment decision

Question	Yes	Maybe	No	No answer
Enrollment at DePaul	34.0%	0.5%	33.5%	31.9%
Choice of a major at CDM	28.8%	0.5%	36.1%	34.6%

The final two questions on the survey were open-ended. The first asked students to list the best thing(s) about the mentoring program. 47.65% did not answer the question and 13.61% reported that the benefits were unknown. Among those students indicating a benefit, the most common were having help/guidance (10.99%), help with classes/curriculum (8.9%), a source of information/answers to questions (8.38%), a personal connection/relationship (3.66%), communication (2.62%), and feeling welcomed/connected (2.09%). When only those students who reported some contact with faculty are considered, the percentage who reported unknown dropped to less than 2%, and the percentage of each common answer increased. The ordering of the common answers remained the same.

In response to the open-ended question about ways the prospective student mentoring program could be improved, 67.54% of students did not answer the question and 4.19% indicated unknown. The largest percentage of students providing a response indicated no suggestions for improvement (10.47%), followed by more contact with faculty (9.95%), and better responses by faculty (5.24%). If only the students who reported contact with faculty are considered, the percentage not answering the question dropped to 16.84% and the number indicating unknown dropped to 1.58%. Among these students the top two responses are more contact (5.26%) and nothing (4.74%).

3.1.2 Correlations

To further investigate the impact of the mentoring program on the participating students we considered correlations between responses to various questions. The first correlation we considered was that between a student's major and a response indicating that the mentoring program had an impact on their decision to come to DePaul. Unfortunately, almost all of the expected counts were less than 5, meaning that the data set was too small for reliable statistical analysis.

Gender was the next factor considered, with correlations between a student's gender and a response indicating that the mentoring program had an impact on their decision to come to DePaul. Here the results were significant at χ^2 (1, N=209) = 211.24, p=0.0. The program had a high effectiveness for female students when making a decision to come to DePaul.

We next considered the relationship between faculty-student contact and the student's decision to come to DePaul. The first significant result was a correlation between the student reporting contact with a faculty member and the student indicating that the program had an impact on their decision to enroll at DePaul with χ^2 (1, N=209) = 264.187, p=0.0. If a student's assigned faculty mentor contacted them it was reflected in a positive response regarding their decision to attend DePaul. Also, there was a high correlation between a student indicating that the faculty mentor did not contact them and that the program did not impact their decision to attend DePaul. As a follow up to this, the relationship between the reported frequency of contact with the faculty mentor and the DePaul enrollment decision was checked. Here there was no significant correlation found.

If the student contacted the faculty mentor there was also a significant correlation with the reported impact of the program on the DePaul enrollment decision at at χ^2 (1, N=191) = 44.246, p=0.0. The students reporting that they contacted their faculty mentor were also more likely to report that the program impacted their enrollment decision, and students who reported that they did not contact their faculty mentor were more likely to report that the program did not impact their enrollment decision. We also checked the correlation between a student reporting contact with a faculty member other than their faculty mentor and their DePaul enrollment decision. Here there was no statistically significant result.

3.2 Faculty analysis

Faculty participants in the mentoring program were surveyed during the 2009-2010 and 2010-2011 academic years. In April 2010 24 (18 male and 6 female) faculty were surveyed and 13 faculty completed the survey for a response rate of 54%. In February 2011 23 (17 male and 6 female) faculty were surveyed and 11 faculty completed the survey for a response rate of 48%. There were some common questions between the two surveys, and the responses from those common questions are presented in this section. First, the gender of faculty responding, what percentage that count represents among all responses, and response rates by gender are provided in the table below.

Table 3. Gender of faculty respondents

Year	**Female**			**Male**		
	#	**%**	**Rate**	**#**	**%**	**Rate**
2009-2010	3	23.1%	50%	10	76.9%	56%
2010 - 2011	2	18.2%	33%	9	81.8%	53%

Faculty were asked how often they contacted their assigned students. They were allowed to select more than one option.

Table 4. Frequency of contact with students

Frequency	**2009-2010**		**2010-2011**	
	Count	%	Count	%
Within 30 days of assignment	10	76.9%	10	90.9%
Weekly	0	0%	0	0%
Monthly	3	23.1%	1	9.1%
I wait for them to contact me	Not an option on this survey		0	0%
Never	1	7.7%	0	0%

Faculty were also asked how they contacted prospective students and were allowed to select more than one option.

Table 5. Methods for contacting students

Method	**2009-2010**		**2010-2011**	
	Count	%	Count	%
E-mail	12	92.3%	10	90.9%
Phone	2	15.4%	1	9.1%
Regular mail	0	0%	0	0%
Other	3	23.1%	0	0%
Not applicable	1	7.7%	0	0%

The faculty members in the 2009-2010 program who selected other for the method of contact indicated that the student services staff send information with the faculty member's name on it and mentioned the event where faculty can meet with prospective students and their families. One respondent indicated that she meets with students and their families upon request.

One question appeared only on the 2009-2010 survey asking faculty how often they answered questions from their assigned students.

Table 6. 2009-2010 Frequency for answering student questions

	Weekly	**Monthly**	**At least once**	**Never**	**No questions**
#	5	3	5	0	1
%	38.5%	23.1%	38.5 %	0%	7.7%

One question appeared only on the 2010-2011 survey asking faculty how frequently they were contacted by students. None indicated weekly contact, 2 (18.2%) indicated monthly contact, 9 (81.8%) indicated contact at least once, and 1 (9.1%) indicated that contact never occurred.

Faculty in 2010-2011 were also asked about the reasons for student contact.

Table 7. 2010-2011 Reasons for student contact

	Count	**%**
Curriculum in selected program	7	63.6%
Curriculum in other program	2	18.2%
Career and job prospects	4	36.4%
In-person visit	6	54.5%
Admitted Student Day	2	18.2%
Logistics (deposits, housing, etc.)	5	45.5%
Other	2	18.2%

Faculty who indicated contact about another issue both mentioned questions about hardware and software available at DePaul.

Faculty in the 2010-2011 program were asked about a positive interaction with students in the program. Three mentioned seeing students in the program much later in their time at DePaul and that it was rewarding to see them progress. Another mentioned how satisfying it was to direct students to the correct major.

When asked about benefits of the program faculty in the 2009-2010 program five indicated that early contact with faculty was helpful, one mentioned that it welcomed students, and two said that the benefits were unclear. In the 2010-2011 program five faculty indicated that contact with professors was beneficial, one said that the program was good for those who use it, and one indicated that having experts in the curriculum was helpful.

The question about improvements to the program produced five suggestions from faculty in the 2009-2010 survey. The responses indicated students should be encouraged to contact faculty, students should be surveyed to determine if it is impacted enrollment, it should be more important in faculty merit review, canned material should be provided by student services, and the mentoring program should be eliminated. Faculty in the 2011-2012 program suggested that students be prompted with sample questions so that they knew what to ask or that faculty mentors should be informed regarding what approaches were successful previously in reaching students.

4 Discussion

There are several themes that emerged from this work. In this section we discuss the results and provide context for them, concluding with an identification of some limits of the study.

4.1 Understanding the results

The first interesting result from the 2010-2011 prospective student survey is the gap between the contact reported by students and faculty. Faculty during that year overwhelmingly indicated (90.9%) that they had contacted students within 30 days of assignment, yet only a minority of students (28.8%) reported that they had been contacted by a faculty member. There was also a gap between students reporting that they had contacted a faculty member (17.8%) and faculty who reported such contact (81.8%). It is certainly possible that the actual amount of contact between the faculty and students lies between the two sets of reported numbers, but it is difficult to say with certainty which is more accurate. One would suspect that faculty would have more accurate recall, but that is simply conjecture. It is, however, somewhat supported by the response of students to the follow-up question about faculty-student interactions, where 85.7% of that smaller group indicated at least one round of contact.

In a surprising result given the relatively small number reporting contact, a majority of student respondents (50.8%) indicated that they would be interested in having their faculty mentor assigned as their advisor. The results from the open-ended questions indicated a combined 28.27% of respondents felt that helpful advice or information was a benefit of the program and 10.47% of students reported that they had no suggestions for improving the mentoring program. This data provides support for the idea that the prospective student mentoring program is beneficial for some of the students. While one would hope that a larger percentage of students would report direct benefit from the program, it is encouraging that those students who recalled participating in the program feel positively about it.

The results regarding the impact of the program on enrollments is more mixed. When directly asked about it, only a minority indicated that it had impacted their decision to enroll at DePaul (34.0%) or to choose a major within CDM (28.8%). More students (36.1%) indicated that it had not influenced their decision to choose a major within CDM, and close to an equal number (33.5%) indicated that it had not impacted their DePaul enrollment decision. However, the analysis of the correlation between responses to some questions suggests that the program may be more effective for some subpopulations. For example, the program was shown to impact female students more strongly. While the analysis also showed that students who contacted their faculty mentors were more likely to have indicated that the program impacted their enrollment decision, this may simply be due to an increased interest or curiosity about DePaul that pre-existed their participation in the mentoring program. More interesting is the result that shows students who were contacted by their faculty mentor were more likely to indicate that the program had an impact on their DePaul enrollment decision.

An important characteristic of prospective student mentoring when interpreting faculty results is the optional nature of the program. Faculty are solicited to participate, and as one faculty member noted the work is only one of many service opportunities in CDM. In the 2009-2010 academic year an opt-out method was used for recruitment, meaning that faculty who had participated before needed to indicate that they no longer wished to continue. Based on the relatively high number who mentioned that the benefits of the program were unclear (2 or 15%) or that it should be eliminated (2 or 15%), an opt-in procedure for recruitment was adopted in 2010-2011. This change in policy and an orientation session for faculty regarding the program improved the percentage of faculty who contacted their assigned students within 30 days of assignment from 76.9% to 90.9%. It also eliminated the suggestions that the program be discontinued, indicating that faculty who were not convinced of the worth of the initiative were no longer participating.

The questions added in the 2010-2011 program show that the faculty believe the program is being utilized by students, with slightly more than 80% of mentors being contacted at least once. The new questions also reveal that faculty field many questions that should be handled by other staff. Faculty typically are unable to arrange in-person visits, have no role except as a participant in the Admitted Student Day, and do not have detailed logistical information. Given that the materials distributed to students have clear statements on whom to contact for this information, strategies for improvement of this aspect of the program are not obvious.

4.2 Study limitations

The first limitation of this work is that the survey instruments developed were not validated. There were no similar studies of prospective student mentoring programs for technically-focused post-secondary institutions upon which to base our work. It would be helpful to consider how the surveys could be more effectively designed. Another significant limitation is the self-reported nature of the data. While care was taken when designing the survey, the participants may have misinterpreted the survey questions. Participant recall is also an issue with a survey discussing a program that happened many months previously, as was seen in the student data from the 2009-2010 evaluation year.

During both years of the evaluation project the only group of prospective students who were surveyed were those who ultimately chose DePaul as their institution. Since only students who chose DePaul are surveyed, this may artificially raise the apparent effectiveness of the prospective student mentoring program. However, legal obligations do not allow us to survey students who do not enroll after they have indicated their decision making this situation difficult to avoid. One possibility is to survey students before they have made a decision one way or another, during which time we are still legally allowed to contact them. There are two drawbacks to this idea. First, many students will be under the age of 18, and surveying them will require a more complex type of human-subjects protocol. The second is that students finishing their last year of secondary school have many obligations and the response rate of the surveys may be low.

5 Conclusion

Between 2009 and 2011 we evaluated a prospective student mentoring program, with the two-fold goal of understanding the impact on student recruitment and the extent of faculty engagement. As a result of the first year of evaluation [11], we made changes to faculty recruitment procedures that produced an improvement in faculty engagement with the program. Although the student data gathered during the first year of the evaluation project was not usable, the changes in the survey timing produced excellent response rates during the second year. In analyzing the responses, we found a gap between reported rates of contact between faculty and student respondents, with faculty reporting higher levels of contact than students. Students did report satisfaction with the program as seen in the relatively high number who indicated an interest in having their faculty mentor assigned as an advisor and in the comments shared on the open-ended questions. There were also several significant correlations that suggest the program is more effective for female students and that students contacted by faculty mentors are more likely to indicate that the program influenced their enrollment decision.

Attracting and retaining students in computing programs is important, and mentoring of prospective students is an important tool for university educators. There are difficulties in measuring success of such programs, most saliently effective student recall. But engaging students with faculty shows promise, and it would be interesting to see if and how prospective student mentoring programs could be implemented at other institutions.

6 Acknowledgements

Our thanks to Mary Jo Davidson for her help in analyzing the faculty and student responses from the 2009-2010 academic year.

7 References

[1] Alkadi, G. et al. 2011. Virtualization of our University for the Recruitment and Orientation of new Students. *Journal of Computing Sciences in Colleges*, 26:4, pp. 71-77.

[2] Binkerd, C. and Fernandez, J.D. 2004. *Journal of Computing Sciences in Colleges*, 19:4, pp. 218-224.

[3] Binsardi, A. and Ekwulugo F. 2003. International Marketing of British Education: Research on the Students' Perception and the UK Market Penetration. In *Marketing Intelligence & Planning*, 21:5, pp. 318-237.

[4] Boyer, K.E. et al. 2010. Increasing Technical Excellence, Leadership and Commitment of Computing Students through Identity-Based Mentoring. In Proceedings of the 41st ACM Technical Symposium on Computer Science Education, Milwaukee, Wisconsin.

[5] Chapman, D. 1981. A Model of Student College Choice. In *Journal of Higher Education*, 52:5, pp. 490-505.

[6] Craig, A. 1998. Peer Mentoring Female Computing Students – Does It Make a Difference? In Proceedings of the *3rd Australasian Conference on Computer Science Education*, Brisbane, Australia.

[7] Cubillo, J.M., Sánchez, J., and Cervino, J. 2006. International Students' Decision-Making Process. In *International Journal of Educational Management*, 20:2, pp. 101-115.

[8] D'Souza, D. et al. 2008. In Proceedings of the *10th Australasian Computing Education Conference*, Wollongong, Australia.

[9] Roberts, E.S. 2011. Meeting the Challenges of Rising Enrollments. *ACM Inroads*, 2:3, September 2011.

[10] A. Settle and J. Glatz. 2011. Rethinking Advising: Developing a Proactive Culture to Improve Retention. In the Proceedings of the *Special Interest Group for Information Technology Education Conference (SIGITE 2011)*, West Point, New York.

[11] Settle, A., Pieczynski, S., Friedman, L., and Davidson, M.J 2011. An Initial Look at Prospective Student Mentoring, In the Proceedings of *the 16th Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE 2011)*, Darmstadt, Germany.

[12] Tashakkori R. et al. 2005. A Systemic Mentoring Model in Computer Science. In *43rd ACM Southeast Conference*, Kennesaw, Georgia.

Transformative Pedagogy for Integrative Amelioration

Mr. Bandi S Herold
Associate Professor,
Dept. of Management Studies
Swarnandhra College of Engineering
and Technology
Narsapur, INDIA
bsherold25@rediffmail.com

Dr. M Muralidhara Rao
Professor,
Dept. of Mechanical Engineering
Swarnandhra College of Engineering
and Technology
Narsapur, INDIA
drmmrao54@gmail.com

Dr. S. Ramesh Babu
Professor,
Dept. of Electrical & Electronics
Swarnandhra College of Engineering
and Technology
Narsapur, INDIA
rameshbabusatrasala@gmail.com

FECS' 13

Abstract— **In the changing global environment, there is a need to assess present learning outcomes. The current education system mainly focuses on academic instruction and not on personal development with learning. For a successful complete educational expertise, transformative education paves the way in acquiring academic learning and student development with increased cognitive understanding, personal maturity and interpersonal effectiveness. Transformative education is not only about acquiring knowledge and skills and socializing students to become professionals but also focuses on developing leadership attributes. By integrating several strands of analysis, a conceptual framework is developed on 'transformative pedagogy', which associates several properties to make the education system more pragmatic and bring forth a sustainable integrative human development. To support today's learning outcome, the focus of education must shift from 'information transfer' to 'identity development'. The core concept of this paper is that transformative education promotes integrative development for students as wholesome individuals.**

Keywords — transformation; pedagogy, diversified heritage; learning paradigm; global higher education;

I. Purpose

The objective of the paper is to imply that learning is a comprehensive, holistic, transformative activity that integrates academic learning and student development which enhances the student's amelioration by cognitive understanding and personal maturity.

II. Introduction

It is apparent to consider the ideological fact that 'Education leads to changes – changes in the amount of knowledge people possess, changes in skills and competencies, changes in the way people communicate and understand each other, changes in the sense of self, and changes in the social world'. The present education system does not really mean a complete education because it is solely and primarily informative and academic in nature. It does not increase the students' cognitive understanding, a sense of personal maturity and interpersonal effectiveness. Therefore, the current learning outcomes of education must shift from information transfer to identity development (transformation). To understand what a true and complete education might mean, it is worth considering the idea of 'Transformative Education'. A transformative paradigm requires considering the University as "*an integrated system for learning*" and breaking down the divisions between student learning and student support/development. Presently, there is a wide gap existing between the type of education imparted in colleges and real time situations. In order to make education more realistic and useful for the organizations and society, certain key areas need to be identified.

III. Key areas of transformative Pedagogy

A. Components of contemporary pedagogy

The components of contemporary teaching and learning models should be defined in terms of – core and secondary. Core being , peer teaching, , student centered debate, interdisciplinary projects, problem solving, team-working, autonomous study, critical appraisal, empowerment, self-confidence, seeking input from external sources, teacher as guide and facilitator, deep analytical thinking, validity and appraisal of sources, internet search skills, and inclusivity. Secondary being, exploratory learning, self-initiative, absorbing, analyzing and synthesizing the concepts, consolidation and application in shifting concepts, role plays, learning support and community liaison.

B. Constructivist Teaching and Learning Paradigm

Teaching learning paradigm can be arrived by abetting the students to have academic shift from rote memorization of prior knowledge to more active learning of a contemporary social constructivist epistemology. Further, the students should be encouraged to move from constructivist philosophy, psychology and epistemology to the characterization of constructivist learning environment which presents the students the challenge of synthesizing a large spectrum of diverse concepts.

C. Incorporating Active Learning Principles

The main principles of active learning that should be incorporated in teaching methods and course objectives are;

- Authentic problem solving – both autonomous and in groups (collaborative enquiry)
- Critical appraisal - such as the evaluation of website reference quality and constructive appraisal of peers.
- Empowerment - the positive spiral of self-confidence.
- Blended learning – online support, adding value to what and how we teach.
- Learning support – workshops and individual support to passionate and struggling students.
- Expanded learning – inviting external stakeholders and professionals.
- Community liaison – where students group together both in class and online to expand and consolidate their learning.
- Team working – in class or online.

D. Competent Faculty members, Infrastructural facilities, Updating of Curriculum

- **Qualified & Competent Faculty**: The institutions offering various courses must have core faculty in main areas of specialization. Faculty must meet at least one of the three criteria viz. Industry experience, Academic & Research experience, knowledge on the latest trends of global higher education.
- **Infrastructure and facilities**: This includes learning venues with multimedia and projection facilities, computers and peripherals, e-books, e-journals and a good physical library with collection of materials for academic knowledge and transformation.
- **Forecasting Global Requirements**: Corporate/ Industry personnel/researchers should forecast the future global requirements of the upcoming generations and re-count to universities during exchange programs for updating the curriculum at regular intervals.
- **Updating of academics with research innovation**: The courses and programmes conducted in the institutions must be in line with the requirements of the global innovative research methodologies on current/ latest technologies.

E. Faculty Development in Education

Faculty development is one of the important activities that institutions use to renew faculty members for their multiple roles. Faculty development activities include programs to enhance teaching, education, research, scholarly activity, academic leadership and management. The intent of these activities is to assist the faculty members for their roles as teachers, educators, leaders, administrators and researchers. The objectives of Faculty Development are;

- To help the faculty members in performing various job roles.
- To give awareness about new pedagogical methods.
- To update their knowledge.
- To assist in solving the real time problems with application of the concepts.
- To give awareness regarding results of various research areas.
- To improve industry interface.
- To bring corporate and academicians on to a common platform for developing an empathy of future generations.
- To develop the students' skills needed for global requirements.
- To encourage publications and research work in domain areas.

F. Fostering digital generation needs

Instructional design approach of traditional pedagogy must come up with boundless opportunities through information technology. Education faces a transformative period in which traditional in-person pedagogy interact with those of internet-based digital learning. Pedagogy design in higher education currently exhibits a hybrid personality in which traditional practices interact with technology on a regular basis for majority of students. The highly personalized redesign of digital age pedagogy can unlock the potential of technology for a new digital generation of students and build a foundation for the future. Online component of technologically integrated pedagogy which includes both in-person and internet-based elements meets the needs and preferences of the digital generation.

G. Providing a Safe, Supportive and Sustainable environment

The outcome of a transformative education needs a safe and supportive environment in which a learning space can be created to enable learners to confront chaos, pain, fears and bewilderedness. Within a safe environment, learners are free from judgment, bullying, negative conflicts and intimidation. They can be courageous enough to be pushed to the boundaries of their realities. Hence learners are more likely to discover multiple aspects and develop as wholesome persons. A sustainable learning environment involves;

- Autonomous capital for the physical environment.
- Effective training for all educational staff.
- Human relations.

H. Transformed liaison of learners, teachers and institutions

In transformative education, teachers are perceived as role models, facilitators, mentors, co-inquirers, critical friends, experienced co-learners, respectful guides and compassionate helpers in the educational process.

This underlines the importance of reciprocal human relations in the educational context. It assumes that learning is in itself, a journey of inquest. The teachers' role is to accompany the learners' journey and help them to overcome life challenges. In this process, teachers will learn and grow.

Present vision also reinforces the need for the nature of educational institutions to center on learners' growth and transformation. Therefore, the notion of institutions as learning communities is fundamental in constructing a new relationship between the learner, the teacher and the institution. Such relationship promotes leadership rather than management, and calls for full constituency dialogue rather than hierarchy and bureaucracy.

I. The primacy of University – Industry Knowledge Transfer

The concept designates a wide range of interactions, at different levels and involving different activities mostly aimed at the exchange of knowledge and technology between universities and technical Industries, which embraces:

- The performance of collaborative research between technical Industries and academic institutions.
- Contract research and academic consultancy commissioned by Industry.
- The development and commercialization of intellectual property rights on the part of universities.
- Co-operation in education.
- Advanced training for enterprise staff.
- Exchange of researchers between technical Industries and universities.

J. Collaboration with foreign universities and global education

The academic collaboration with reputable foreign universities/ institutions will be able to enrich the quality of their programs by twinning the teaching expertise and latest curricular developments. Further, institutions will be able to provide their students with first-hand knowledge of the latest developments, trends, and techniques available at the foreign universities. As a consequence, institutions will have a marketing edge for attracting a wider pool of good quality students from home institutions and abroad.

i. *Faculty exchange programs*: These provide an opportunity to teach or conduct research for a semester or an academic year at foreign university. Faculty members would benefit from exposure to a culturally varied and diversified heritage to exchange ideas and observe a variety of styles. This would benefit the students with an opportunity to learn concepts and ideas bestowed in an exclusively new and distinctive manner. The faculty exchange programs present a unique opportunity for interaction between foreign universities and home institutions.

ii. *Academic partnership programs*: These programs benefit in improving the academic quality of the home institutions and prepare the students for entry into foreign universities. Partnership programs provide a measure of certainty to students in home institutions so that they will be accepted in good foreign universities for global exposure.

iii. *Distance Education programs*: In the current era, with greater usage of internet and web, a phenomenal growth of distance education has cropped up. Home companies and organizations are spending huge amounts on in-house training and education programs through distance and blended mode. Blended learning is the process of incorporating many different learning styles that can be accomplished through the use of blended virtual and physical resources. Home institutions including companies and organizations, in academic collaboration with foreign universities, would offer an array of degree, diploma, and certificate programs in a wide range of fields, thus making the students strategically fit into the global environment.

iv. *Vocational Education programs*: Vocational education intends to develop skilled manpower through diversified courses to meet the requirements of organized and unorganized sectors and to instill self-employment skills in people through a large number of self-employment oriented courses.

K. Designing Effective Learning Experiences

In a typical technical and professional program, the practicing and learning may be completed on an average of six hours per week per course. If a course lasts for 15 weeks and a degree program has a total of 40 courses, then the total time spent on application and practice would be about 3,600 hours. The number of hours spent by a typical student in truly active engagement, however, is likely to be far less than 3,600 hours. Thus, to maximize the impact of time spent in practice and application, it is imperative that one should design the most effective learning experience. The use of a systematic curriculum design process can assist in increasing the chances of successful curriculum-level integration with learning experiences.

Current understanding of expertise and the learning processes that are developed will indicate that the education system should encompass a set of learning experiences that allow the students to have deep conceptual knowledge for developing the key technical and professional skills. Technical/Engineering curricula and teaching methods are often not well aligned with these goals. Curriculum-level instructional design processes should be used to design and implement changes that will improve alignment.

L. Transformative learning in multicultural education

A cross cultural reality differs remarkably in terms of race, ethnicity, gender, class, language, etc., There is a need to develop the knowledge, skills and dispositions for successful outcome of multicultural education. Therefore, the course should be designed for challenging the students to;

i. Explore issues of equality, social justice, and socially just teaching.

ii. Engage in deconstructing the educational system within a historical, cultural and political context.

iii. Examine self in terms of their assumptions, beliefs, values and knowledge.

iv. Construct new lenses and perspectives about self, others, and praxis,

v. Cultivate vision of hope and possibilities for transformative practice and changes.

The pedagogies that can foster students' transformative learning include;

i. Establishment of dialogical relationship in learning community for creation of comfortable learning climate for the students to share the ideas.

ii. Encouragement of writing pre-post narrative inquiries to make the students move from ignorant stage to a new understanding.

iii. Engagement of students in experiential activities for engendered deep thinking of oneself, society and praxis.

iv. Critical textual disclosure (course materials, text books, articles, video materials) would be thought provoking and insightful. This would positively impact the learning transformation.

v. Engagement of students in structured class room and online threaded discussions would influence learning transformation.

vi. Humanization of teaching and learning process can influence the learning transformation.

Successful learning in multicultural education courses, professors should have the mind and habits of transformative practice and they must be proactive and engage in pedagogies that are empowering, humanizing and liberating to foster meaningful knowledge construction.

M. Institutional Training & Placement to meet business entity needs.

It is a fact that the business entity needs to continuously evolve and re-invent itself because the rapid advancement of new and breakthrough technologies keep shaping and reshaping the contours of the business on an ongoing basis. Such issues call for varied knowledge and skill set. Thus it's about time that all the educational institutions recast their system of training and placement of students to meet the needs of the organizations. It is a very vital area which has become important parameter to assess the strength, value and reputation of the educational institutions. The institute needs to make their syllabus more vocationally oriented so as to groom, nurture and develop the talent in a proper fashion, catering to the needs of the organisations. The students should be encouraged to attend technical seminars, workshops, comprehensive training programmes for the placements and knowledge of the latest developments in technologies and its impact on business. Equal importance should be given to the communication skills of students for clear expression of their ideas.

IV. Conclusion

The education system has to be transformed to meet the needs of the organizations and society. It has to undergo a paradigm shift from success to excellence. To bring about the paradigm shift in education, an attitudinal change, a process oriented approach and an emphasis on tacit knowledge is required. Transformative education paves the way for the development of total human potential including physical, moral, creative, emotional, intellectual and spiritual. Transformative education stimulates personal change and offer opportunities for the students to learn new ways of making a meaning, both subjective and objective. It intends to help students to cultivate their rational capabilities. Transformative education seeks for a safe, supportive and sustainable learning environment with a transformed view of relationship between learners, teachers and institutions. Transformative education helps to promote integrative development of students as wholesome individuals. Transformation can bring forth a vital effect on overall human development, social transformation and ultimately building a better world.

Acknowledgment

The authors acknowledge the ideas/reviews from various senior academicians, affiliating university authorities.

We also acknowledge the moral support given by the management for this activity.

References

1. Fontana, R, Geuna, A. and Matt, M, (2006), “Factors affecting university-Industry R&D projects
2. Lawton smith, H. (2007), “universities, innovation and territorial development; a review of evidence”.
3. Ambrose, S. A., Bridges, M. W., DiPietro, M., Lovett, M. C, 8c Norman M. K (2010). How learning works: Seven research-based principles for smart teaching. San Francisco, CA: Jossey-Bass.
4. Atman, C. J., Sheppard, S. D., Turns, J., Adams, R S., Fleming, L. N., Stevens, R ..... Lund, D. (2010). Enabling engineering student success: The final report for the Center for the Advancement of Engineering Education. Retrieved from http://www.engr.washington.edu/ caee/CAEE final report 20100909.pdf
5. Baillie, C. (2010,JuIy). Waste for life. Retrieved from http://wasteforlife.org/
6. Biggs, J. (2003). Teaching for quality learning at university: What the student does (2nd ed.). Berkshire, UK The Society for Research into Higher Education and Open University Press.
7. Boshuizen, H. P. A. (2010). Teaching for expertise: Problem-based methods in medicine and other professional domains. In K Anders Ericsson (Ed.), Development of professional expertise: Toward measurement of expert performance and design of optimal learning environments (pp. 379-404). New York NY: Cambridge Press.
8. Bransford, J. D., Brown, A. L., 8c Cocking, R. R (Eds.). (1999). How people learn: Brain, mind, experience, and school. Washington, DC: National Academies Press.
9. Brint, S., Cantwell, A. M., 8cHanneman R. A. (2007). The two cultures of undergraduate academic engagement. Research in Higher Education, 49(5), 383-402.
10. Brodie, L. (2007, July). Problem based learning for distance education students of engineering and surveying. Paper presented at the Conference on Design Education, Sydney, Australia. Retrieved from http://eprints.usq.edu.aU/2450/2/Brodie_L_2007_Connect ED_ final.pdf

11. Catalano, G., Baillie, C, Riley, D., Nieusma, D., Byrne, C, Bailey, M., 8c Haralampides, K (2010). Integrating social justice ideas into a numerical methods course in bioengineering. Proceedings of the American Society for Engineering Education. Retrieved from http://soa.asee.org/paper/conference/paper-view.cfm?id=24339

12. Case, J., 8c Light, G. (2011). Emerging methodologies in engineering education research, fournal of Engineering Education, 100(1), 186-210.

13. The Lancet Commissions : transforming education. Lancet. Published

14. online, 29 November 2010, DOI:10.1016/S1040-6736(10)61854-5.

15. The National Association of Student Personnel Administrators: http://www. naspa.org/membership/leader_ex_pdf/lr_long.pdf

16. The Association of American Colleges and Universities: http://www.greaterexpectations.org/

17. Learning Reconsidered: A Campus-Wide Focus on the Student Experience (NASPA, 2004).

18. Greater Expectations: A new vision for learning as a nation goes to college (AAC&U, 20022).

19. Cranton, P. (1997). Editor's notes. In P. Cranton (Ed.), Transformative learning in action. [Monograph]. New Directions for Adult and Continuing Education, 74, 1 – 3.

20. Grabove, V. (1997). The many facets of transformative learning theory and practice. In P. Cranton (Ed.), Transformative learning in action. [Monograph]. New Directions for Adult and Continuing Education, 74, 89–95.

21. Brookfield, S. D. (1986). Understanding and facilitating adult learning.San Francisco: Jossey-Bass.

22. Mezirow, J. (1978). Perspective transformation. Adult Education Quarterly, 28(2), 100-110.

23. Mezirow, J. (1997). Transformative learning: Theory to practice. In P.Cranton (Ed.), Transformative learning in action. [Monograph].New Directions for Adult and Continuing Education, 74, 5 – 12.

24. Abell M (2006) individualized learning using intelligent technology and universally designed curriculum. The journal of Technology, Learning and Assessment.

25. Pedagogies that foster transformative learning in a multicultural education – The Berkeley Electronic Press, 2009

Exploring Computer Science and a High School Program of Study in Computing

D. Lewis[1] **and R. Davis**[1]
[1]Computer Engineering Department, Santa Clara University, Santa Clara, California, USA

Abstract - *Starting from the Exploring Computer Science course, originally developed for high schools in Los Angeles, we have attacked the problem of spreading CS curricula into the high school level. In this paper we review the state of computing at the high school level; describe our project, the partnerships that make it possible, and the impact on students; then report the lessons learned and plans for developing a high school CS program of study beneficial to both students continuing to higher education and to those immediately entering the workforce.*

Keywords: ECS, K-12, Pathway, Course Sequence.

1 Introduction

The Bureau of Labor Statistics predicts that computing will create more jobs between 2010 and 2020 than all other science and engineering fields combined [1]. Although undergraduate enrollments in computer science (CS) have increased in each of the past five years, including a 29% increase from 2011 to 2012 [2], only 30% of the 1.4 million jobs for computer specialists projected by 2020 are expected to be filled by U.S. computing graduates [3], and only 2% of ACT-tested 2012 high school graduates expressed a career interest in computer and information specialties [4].

The problems with high school CS curricula are well known. Since 2005, the number of high schools offering an introductory CS course decreased by 17% and those offering an Advanced Placement (AP) CS course decreased by 33% [5]. In 2012, only 2,978 of the more than 42,000 U.S. high schools offered the AP CS exam [6]. Between 2002 and 2012, while the number of students who took AP exams in all fields of math and science more than doubled, the number who took the AP exam in CS remained relatively unchanged [7].

In our own state of California, the dot-com bust of 2000 and news of off-shoring had a significant personal impact on many families and likely influenced students against pursuing a career in computing. While the total number of California high school students increased 20% from 2.48 to 2.98 million from academic year 1998-99 to 2008-09, enrollments in CS courses fell by 34% from 322 to 212 thousand [8] and made CS classes some of the first to go when the state's education budgets were slashed. California now has one of the lowest CS AP participation rates in the nation; despite being a high-tech center for computing and the home of Silicon Valley, only 3,101 California high school students took the CS AP test in 2011 [9].

There's a perfect storm building now around high school CS – both in California and across the nation. Even President Obama has said that he thinks "it makes sense" to have a computer programming language graduation requirement in high schools [10]. He stated that it is important for those students who will go to college as well as for those who plan to immediately enter the workforce. Several independent but related efforts have developed new CS courses that are now spreading throughout our nation's high schools, creating opportunities to integrate these courses into an appropriate sequence referred to as a program of study.

1.1 Curriculum development efforts

In 2006, Jeannette Wing introduced her well-known article on Computational Thinking [11], which influenced Google's creation of the Computer Science for High School (CS4HS) initiative to promote CS and Computational Thinking. The CS4HS project has now reached more than 1,800 teachers and 130,000 students in the United States alone [12]. More recently, the International Baccalaureate (IB) Program announced a revision of their CS courses, reducing the emphasis on Java programming and instead focusing on Computational Thinking [13]. In her role at the National Science Foundation (NSF), Jan Cuny has been leading the development of the CS10K project that seeks to have 10,000 well-trained CS teachers in 10,000 high schools across the United States by 2015 [14]. As part of that effort, two new high school CS courses have been developed to supplement the existing CS AP course: an introductory pre-AP course called "Exploring Computer Science" (ECS) and an intermediate course to be offered for AP credit called "Computer Science Principles" (CSP).

ECS was originally created "to increase and enhance the computer science learning opportunities in the Los Angeles Unified School District (LAUSD) and to broaden the participation of African-American, Latinos, and female students in learning computer science" [15]. Because of its success, ECS has now spread to more than 80 schools in five states [16]. To facilitate its adoption, SRI International recently released draft curriculum alignments of ECS to the California [17] and Illinois [18] state standards, including correlation to the Common Core Standards, the Next Generation Science Standards, the ISTE National Educational Technology Standards (NETS) and the K-12 Computer Science Standards developed by the Computer Science Teachers Association (CSTA). These curriculum alignments were prepared by one of our ECS teachers while working at

SRI during a summer fellowship awarded by the Industry Initiatives for Science and Math Education (IISME).

CSP was developed in conjunction with the College Board, which plans for the first AP test for CSP to be offered during the 2016-17 academic year. NSF-sponsored pilots of CSP have now been offered at 20 universities and high schools in 12 states [19]. Project Lead The Way (PLTW) will pilot its own version of CSP called "Computer Science and Software Engineering" (CSE) in 50 high schools in 2013-14, with plans to implement the course at more than 500 schools the following year [20].

In California, the Mid-Pacific Information and Communication Technologies Center (MPICT) is an NSF-funded effort aimed at improving ICT education in its region, which encompasses California, Nevada, Hawaii, and the Pacific Territories. While the emphasis of MPICT is on two-year community colleges, they also work with K-12 networks, other NSF-funded Centers (through the Advanced Technological Education Program), the California Community Colleges ICT Collaborative, four-year institutions, and industry employers to identify best practices, leverage and improve ICT education in the region to create a diverse technologically prepared workforce to meet the economic demands of the region. One of their goals is to identify student learning outcomes that may serve as the basis for educational pathways.

The Alliance for California Computing Education for Students and Schools (ACCESS)[1] is a collaborative effort bringing together key stakeholders in CS education (K-12 administration, CS post-secondary education, schools of education, teacher training programs, industry, and both regional and national non-profits/policy organizations) to provide equitable access to high-quality K-16 CS education for all California students.

ConnectEd, the California Center for College and Career, is a partnership of several organizations focused on bringing a Linked Learning approach to California high schools. In this approach, students follow a multiyear sequence of integrated academic and Career Technical Education (CTE) courses organized around a broad theme, interest area, or industry sector. Used in schools throughout California, this approach has been shown to lead to higher graduation rates, increased postsecondary enrollments, higher earning potential, and greater civic engagement.

2 Our approach

Our project expanded the reach of ECS to Silicon Valley. Since the summer of 2010, we have been working to establish ECS as the foundation for a sequence of CS courses for students in San Jose area high schools. We began by identifying high-need districts and high schools, meeting with district superintendents, principals, counselors and teachers to obtain their support, and applying for funding from NSF. The following sections describe how we selected our partner districts and schools, how our NSF grants support our work, and the content and pedagogy of the ECS curriculum.

2.1 Selection of schools and districts

We chose to work with school districts that had a significant percentage of students who are under-represented in computing and are located in lower-income areas of San Jose. We selected schools from three different school districts - the East Side Union High School District (ESUHSD), the San Jose Unified School District (SJUSD), and the Santa Clara Unified School District (SCUSD). In each district, more than half of the students were neither white nor Asian, and 43% were enrolled in the free or reduced price meal program.

ESUHSD is one of California's largest high school districts, with an enrollment of over 25,000 students in 22 high schools encompassing an area of 180 square miles. Fifty native languages are spoken by East Side students and there is a wide range of socioeconomic levels in the district. Five high schools in ESUHSD were selected for the project: Evergreen Valley, Independence, Oak Grove, Santa Teresa, and Silver Creek.

SJUSD covers the major portion of the city of San Jose, and a geographic area of over 100 square miles. As one of the largest urban school districts in California, SJUSD has 27 elementary schools, six middle schools and seven high schools. The District's high school student population totals almost 10,000 students. Three high schools in SJUSD were selected: Pioneer, San Jose, and Willow Glen.

SCUSD serves over 14,700 K-12 students, and an additional 14,000 students in alternative programs offered from Preschool through Adult Education. Neighborhoods in the cities of Santa Clara, Sunnyvale and San Jose comprise the District's 56 square-mile area. The District includes 16 elementary schools, one K-8 school, three middle schools, and two comprehensive high schools. Both high schools in SCUSD were selected: Santa Clara and Adrian Wilcox.

When we began our project in the summer of 2010, there were very few CS courses offered at our partner schools. When the principals were asked about CS courses that were available, many reported courses with titles such as "Computer Applications" (i.e., Microsoft Office), "Computer Fundamentals", "Digital Electronics", "Web Design", "Multimedia", "Technology Literature", or "CAD Drafting". Out of the ten schools, only two reported offering both an "Introduction to Computer Science" programming course and an AP CS course. A second school offered only the introductory course, and a third only the AP CS course. The equipment situation wasn't much better: When asked to offer ECS, all but two of the schools said that they would need a

[1] ACCESS was previously known as the California Computing Education Advocacy Network (CCEAN).

computer lab – either because they didn't have one, or because it was already committed to other courses.

2.2 Funding and equipment

Our project has been supported by two NSF grants and a donation of 350 desktop computers from Santa Clara University (SCU) through its three-year replacement cycle of office and laboratory computers.

An initial grant from NSF's Broadening Participation in Computing (BPC) program funded workshops for teachers on the ECS curriculum and pedagogy, and provided LEGO Mindstorm™ robotics kits and funds for administrative support in each of the schools. We held four weeks of summer professional development (PD) in 2010, three in 2011, and two in 2012, with some teachers attending more than one summer. Each summer PD was supplemented by three one-day follow-up PD sessions on Saturdays in the fall and a reflection workshop in the spring. To date, we have trained a total of 17 ECS teachers; of these, ten are teaching ECS in our partner schools, some have not yet taught ECS, some have retired or been reassigned to other courses, and two were from schools outside our partner districts.

A second grant from NSF's Graduate STEM Fellows in K-12 Education (GK-12) program funded graduate engineering students to support the ECS teachers in their classrooms, and provided stipends to the teachers for helping to strengthen the organizational, leadership and presentation skills of the graduate students. We gave preference to graduate students who could serve as role models for students typically under-represented in computing. They each received a week of PD on the technical content of ECS and two days on classroom management, and were then assigned to two ECS class sections. They work in each ECS classroom twice a week and meet with the teachers after class to reflect and plan. They are expected to help prepare lesson plans, assignments and assessment instruments, to provide technical support to the teachers and students, and to prepare and give a minimum of six lectures in each classroom throughout the year.

2.3 Description of the ECS curriculum

The ECS curriculum is designed to engage high school students in computational thinking and make them aware of the importance of computing in a huge variety of disciplines and careers. The students learn about problem-solving, algorithm development, programming, and the use of computing applications in a variety of contexts that make sense in their lives. The curriculum also addresses the societal context and ethical issues that arise with the ubiquitous use of computers today.

As we believe was intended by the developers of the original ECS curriculum [21], we have modified some of the lessons to make them relevant to our population of students. The specific computing tools and programming languages that are used in each unit are less important than the inclusive and engaging pedagogy that encourages students to make computational thinking concepts their "own."

The pedagogy outlined in the curriculum is inquiry-based. It uses the "5 E Model" by R. Bybee [22], detailing strategies that enable the students to Engage, Explore, Explain, Elaborate, and finally Evaluate their own progress and developing expertise in each of the topics. The students learn to analyze problems, ask questions, work in teams, reflect on their own thinking processes and communicate such, and to be creative in their quest for solutions to problems or projects in which they have a personal interest.

The year-long course consists of six modules: Human Computer Interaction, Problem Solving, Web Design, Introduction to Programming, Computing and Data Analysis, and Robotics. While the teachers at each school have some flexibility in organizing their own classes and the time spent on each unit, we expect that they all cover each of the units. To help engage the students in mastering these topics we've introduced a contest at the end of the academic year, with awards for entries in each of the units on Web Design, Programming, Data Analysis, and Robotics. Within each school, students compete to represent their school as finalists in the project-wide competition held on the SCU campus in May.

3 Results and lessons learned

Our success with ECS has only been possible because of partnerships throughout the K-12 community. Besides the teachers who taught the ECS curriculum, we could not have succeeded without the district superintendents who endorsed the project and were instrumental in the selection of our partner schools, curriculum coordinators who pursued approval for ECS to count towards the University of California admission requirements as a "g" elective and helped to navigate course prerequisite and scheduling constraints, principals who supported the ECS teachers and encouraged the development of new course sequences, and counselors who helped recruit students into the ECS courses.

After a modest start, our partner schools are now experiencing increased student demand for ECS. Many schools have added additional sections, with at least one reporting that they could fill as many sections as they are able to offer. The ECS course is currently running in nine of these schools, with 365 students enrolled in a total of 14 sections. Since the project began, 30 sections of ECS have been taught to more than 785 students.

3.1 Student demographics

Unfortunately, while we are offering the courses in schools with a high percentage of underrepresented students, we have yet to match the demographics of the schools with the enrollment in the ECS course. The majority of the students enrolled in ECS is male and not of ethnicities considered to be under-represented in STEM. In the current year, 55% of the ECS students identified themselves as either Caucasian or Asian, and only 21% of the students are female. We are attempting to increase the number of under-represented students through recruitment efforts mentioned in section 3.3, below.

3.2 Use of graduate students

The graduate students have been an asset much appreciated by the high school teachers and students. Initially (2011-12 academic year), the graduate students were only used as helpers in most of the high school classrooms, although exceptions in which the graduate student taught several classes did occur. For the 2012-13 academic year, we have provided more specific guidance on the use of the graduate students in order to ensure that they are developing communication skills and sharing more of their own work with the students, as well as providing deeper expertise in the subject matter of the course in the classrooms.

The following comments are representative of the feedback that we received from students and teachers about the added benefit of having the graduate students in the classrooms:

ECS Teacher: *"I feel less on my own having the graduate student to help. I feel like I have support. Just having anyone knowledgeable to talk to is a great help. This makes me more confident about giving assignments to the students."*

High School Student: *"My positive attitude towards computer science has increased, especially that [the graduate student] has talked about not only the skills you need, but also the attitude you need (like patience) when it comes to computer science. He made us realize that going to college and learning about this will be fun, but at the same time, not everything is easy, and patience is a must because some activities take a LONG time to accomplish."*

The graduate assistants have expressed an appreciation for the impact they have as role models and mentors to the high school students:

Graduate Assistant: *"I think the students are impressed that someone like me is in engineering and knows the material as well as the teacher. This gives them confidence and motivation. In addition, the only female student in one class is motivated to continue in engineering, as I am also a female, and sees that I have made it as well."*

3.3 Student recruitment

We have increased our efforts at recruitment, paying closer attention to the timing for fall registration choices in each high school, and having undergraduate students give presentations to high school students about the various career paths in computing. We believe that these efforts, and word of mouth among the students, have contributed to the 33% increase of ECS enrollments from 2011-12 to 2012-13. However, we still find that it is difficult to get the right mix of students enrolled in the classes, and need to improve our efforts to recruit the most underrepresented students. To this end we have supplied materials to the guidance counselors within each school, ensuring that they are aware of the benefits of learning about computing, and encouraging them to recommend the course to students currently underrepresented in the course at their school.

3.4 Teacher certification issues

California, like most states, does not offer teacher certification in CS. Instead, who is allowed to teach ECS is determined by whether each school district classifies the course as either primarily academic or vocational. If a CS course is not a CTE course, the teacher must either hold a credential in business, mathematics or industrial technology education (ITE), or a supplementary authorization in computer concepts and applications. However, if ECS were considered to be primarily vocational, then it could be taught by a CTE teacher. All of the teachers in our partner schools hold a credential in math.

3.5 How does ECS fit student graduation needs?

In California, students who plan to continue on to college are guided in their choice of courses by the University of California's (UC) "A through G" admission requirements. Neither the UC requirements nor the high school graduation requirements include a course in CS. The ECS course has been approved by UC to satisfy the one year of coursework required from their "G-elective" category, but ECS competes within this category with a large number of other courses. (UC does require two, and recommends three, years of coursework in their "D-laboratory science" category, but historically this has been restricted to courses in the natural sciences, such as biology, chemistry and physics.) The ECS course also qualifies for CTE credit, which is useful for those students planning to enter the workforce directly after high school.

In order to make it clear how the ECS course can fit into both academic and career plans, we are currently cooperating in the effort to create a "Pathway" or sequence of computing courses, described in the next section.

3.6 How does ECS fit within schools' existing curricula offerings?

With budgets as they are, in California as well as elsewhere, schools cannot add new courses just because it seems like a good idea for the students. Some schools already have courses in "computing" – these courses can be from a wide variety of offerings such as keyboarding, using document editing tools, spreadsheets, photo editing tools, robotics, or programming classes. All of these and the ECS course are electives, and thus are the first to be eliminated in favor of required courses when budgets are reduced. A solid case needs to be made if the school is to replace a current offering, or add another course that complements it.

4 Next steps

The introduction of ECS into San Jose area high schools has been enthusiastically received by students, teachers, and administration. Students are asking what they can take next in computing, teachers are excited by the level of student interest, and school and district officials have started to think about how best to develop appropriate and coherent CS sequences for both CTE and college bound students.

4.1 Developing a CS Program of Study

A “Program of Study” is a prescribed multi-year sequence of courses to prepare students for advanced study or a career in a particular discipline. A typical high school sequence would consist of three or four courses from the specific discipline that include a number of required foundation courses and a possible set of electives.

ECS has proven to be a successful and stable curriculum, and has been recommended by ACCESS to be the first foundation course in any CS Program of Study. However, ECS would be insufficient as the only prerequisite for the existing AP CS course and Programs of Study that include the latter will need an intermediate level course such as CSP. A standard CSP curriculum, however, is likely to continue evolving until the 2016-17 introduction of its AP exam. Thus although CSP may ultimately become the pertinent second foundation requirement, schools that want to establish a Program of Study now may effectively use an introductory programming course as a temporary alternative.

A Program of Study for students who intend to pursue advanced study in CS should include the existing AP CS course, possibly followed by a fourth (elective) course such as Web Design, Database Systems, Networking, or System Administration. A Program of Study for CTE students might appropriately replace AP CS by a second elective.

4.2 Sustaining a CS curriculum

Once our NSF project is over, the continued success of ECS and any Programs of Study in CS requires that the K-12 community take ownership of curriculum development, training and support. As a first step, this summer we will be training some of our current ECS teachers to become teacher-trainers in a new ECS workshop to be held at the Santa Clara County Office of Education (SCCOE). Statewide interest in ECS is high, as evidenced by the fact that within a week of opening registration, 30 teachers from outside our partner districts had registered for the first SCCOE workshop to be held this August.

5 Conclusions

This is a critical time for high school CS. The ECS and CSP courses have been enthusiastically adopted at several schools across the nation, and there is a renewed interest among school administrators to expand their course offerings in CS. We have described how these courses form the foundation of a high school Program of Study in CS and our efforts to move to a sustainable model in which the K-12 community takes ownership of CS curriculum development and support.

6 Acknowledgements

This material is partially based upon work supported by the National Science Foundation under Grant Numbers 1019217 and 1045434. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

7 References

[1] U.S. Bureau of Labor Statistics, "Employment by major occupational group," [Online]. Available: http://www.bls.gov/emp/ep_table_101.htm. [Accessed 27 Feb 2013].

[2] S. Zweben, "Computing Degree and Enrollment Trends," Computer Research Association, Washington, D.C., 2011-12.

[3] National Center for Women in Information Technology, "By the Numbers," [Online]. Available: http://www.ncwit.org/sites/default/files/resources/2012bythenumbers_web.pdf. [Accessed 27 Feb 2013].

[4] ACT Research, "The Condition of College & Career Readiness," 2012. [Online]. Available: http://media.act.org/documents/CCCR12-NationalReadinessRpt.pdf. [Accessed 27 Feb 2013].

[5] Computing in the Core, "Facts and Resources," [Online]. Available: http://www.computinginthecore.org/facts-resources. [Accessed 27 Feb 2013].

[6] College Board, "Number of Schools Offering AP Exams (by subject)," [Online]. Available: http://media.collegeboard.com/digitalServices/pdf/research/2012_number_of_schools_offering_ap.pdf. [Accessed 27 Feb 2013].

[7] College Board, "AP Program Participation and Performance Data 2012," [Online]. Available: http://media.collegeboard.com/digitalServices/pdf/research/2012_exam_volume_change.pdf. [Accessed 27 Feb 2013].

[8] Lloyd McCabe, CA Dept. of Ed., Compiled from Data Quest data.

[9] College Board, "AP: Exam Grades: Summary Reports: 2011," [Online]. Available: http://www.collegeboard.com/prod_downloads/student/testing/ap/sumrpts/2011/CALIFORNIA_Summary.xls. [Accessed 27 Feb 2013].

[10] TheOfficalACM, "Computer Science, Zuckerberg and Video Games," 15 Feb 2013. [Online]. Available: http://www.youtube.com/watch?v=XlOySsg7oxY. [Accessed 27 Feb 2013].

[11] J. M. Wing, "Computational Thinking," Communications of the ACM, vol. 49, no. 3, pp. 33-35, 2006.

[12] Google, "Google Computer Science for High School," [Online]. Available: http://www.cs4hs.com. [Accessed 27 Feb 2013].

[13] International Baccalaureate, "Changes to the IB Diploma Programme computer science courses: A guide for universities," International Baccalaureate, 2011.

[14] J. Cuny, "Transforming Computer Science Education In High Schools," IEEE Computer, pp. 107-109, June 2011.

[15] "Our Mission - Exploring Computer Science," [Online]. Available: http://www.exploringcs.org/about/mission. [Accessed 4 Mar 2013].

[16] Exploring Computer Science, "A National Program," [Online]. Available: http://www.exploringcs.org/about/ecs-now. [Accessed 27 Feb 2013].

[17] SRI International, "Exploring Computer Science Curriculum Alignment to Learning Standards - California Teacher Edition, Draft version 0.1," 2013.

[18] SRI International, "Exploring Computer Science Curriculum Alignment to Learning Standards - Illinois Teacher Edition, Draft version 0.1," 2013.

[19] CS Principles, "Pilot Sites," [Online]. Available: http://www.csprinciples.org/home/pilot-sites. [Accessed 27 Feb 2013].

[20] Project Lead The Way, "Computer Science and Software Engineering (CSE) Course Information," Indianapolis, 2012.

[21] G. Chapman and J. Goode, "Exploring Computer Science, Version 4.0," [Online]. Available: www.exploringcs.org/curriculum. [Accessed 8 March 2013].

[22] R. W. Bybee, "Designing Curriculum for Student Learning," in Learning Science and the Science of Learning, National Science Teachers Association Press, 2002, pp. 25-36.

Coaching Robotics Competitions with Tekkotsu

Xuejun Liang
Department of Computer Science, Jackson State University, Jackson, MS, USA

Abstract - *The robotics competitions held along with the Annual ARTSI Students Research Conferences were designed to encourage undergraduate students from non-traditional backgrounds in the study of robotics in areas that are relevant to society. The tasks in these competitions were challenging, including robot searching and localizing itself inside a maze, finding canisters and pushing them into pen with paddles, and finding canisters, pickup them up with grippers, and transport and drop them to a goal location. These tasks are involved in robot sensing, navigation, manipulation, localization, and etc. This paper will breakdown these tasks into pieces and present detailed steps to complete each task so that making these tasks accomplishable by our robotics team students and suitable for being major robot programming lab projects in an upper-level undergraduate robotics course for underrepresented students. Meanwhile, the Tekkotsu robot programming tool will be also introduced for programming the robot to accomplish these competition tasks.*

Keywords: Robotics Education, Robotics Competition, Robot Programming, Tekkotsu

1 Introduction

Teaching an upper-level undergraduate robotics course at Historically Black Colleges and Universities (HBCUs) is challenging because of the lack of suitable teaching materials, especially the robot programming lab projects. Thanks to the ARTSI (Advancing Robotics Technology for Societal Impact) Alliance [1]. It provided a family of unified robot platforms called Calliope [2-4] to each HBCU schools in the alliance, and the technical support to the robot programming tool called Tekkotsu [5-6]. More important and helpful are robotics competitions held each year along with the Annual ARTSI Students Research Conferences since 2010 [7-10]. The tasks in these competitions include robot searching and localizing itself inside a maze, finding canisters and pushing them into pen with paddles, and finding canisters, pickup them up with grippers, and transport and drop them to a goal location. These tasks are involved in robot sensing, navigation, manipulation, localization, and so on. These tasks can be excellent hands-on learning materials for our upper-level undergraduate robotics course. However, these tasks are quite challenging to our students. A detailed guidance to lead to solve the problem is desirable.

This paper will breakdown these tasks into pieces and present detailed steps to complete each task so that making these tasks accomplishable by our robotics team students and suitable for being major robot programming lab projects in an upper-level undergraduate robotics course for underrepresented students. In fact, three of the four competition tasks have been already transformed to our robot programming lab projects [13].

It should be pointed out that all the tasks mentioned early are related to the robot vision in a sense that robots need to use their visions (cameras) to sense their environment to act to achieve their goals. These robotics applications require that a robot can be programmed to be able to know what it wants to see, to recognize what it observed, to compute the orientation, the size, and the location of objects it detected, to reason about the spatial relationship among observed objects, to figure out the geographical layout of the objects, and to build the map. These applications are apparently very interesting to computer science students, and able to broaden their understanding of computer science concept and encourage them to learn more. However, programming a robot in these applications from scratch can be very difficult. Fortunately, the Tekkotsu robot programming development environment provides users with a set of high level supports, including vision processing for shape extraction, shape predicates, and mapmaking. Hence, the Tekkotsu robot programming tool will be introduced for programming the robot to accomplish these competition tasks in this paper.

The Calliope robot family used in our robotics course and competitions include Calliope basic model, formerly called the Create/ASUS robot, and Calliope2SP which added 2-DOF arm, Sony Eye webcan, and pan/tilt on the basic model [2-4]. The iRobot Create uses differential drive and equips buttons for power, play, advance, and wheel drops (front, left, and right), bumps (left and right), IR sensor, wall sensor, cliffs sensors (left, front left, right, front right), encoders (distance, angle), and Leds. An ASUS netbook computer sitting on top of iRobot Create is functioned as the brain of the Calliope robot. The Tekkotsu are installed on the ASUS computer.

In the rest of this paper, the robot programming with using the Tekkotsu will be introduced first in Section 2. Four robotics competition tasks will be detailed and discussed from Section 3 to Section 6, each with one section, respectively. In the first two tasks, a robot needs to navigate and localize itself within a maze, and to announce detected objects and their locations in the maze. But, the objects inside the maze and the navigation markers on the walls of the maze are different in these two tasks. In the last two tasks, a robot needs to locate canisters and move them to a goal location. But, one task will use two attached paddles to push canisters and the goal location is a square region marked with colored tapes on the ground. The other task will use the gripper to pickup canisters and the goal location is marked by an AprilTag on the wall. Note that the discussions for each task will focus on two aspects: 1. task and detailed steps to guide students to accomplish it, and 2. necessary knowledge for completing the tasks, including programming skills, mathematical formulas, algorithms, and issues regarding to failures and uncertainty. Finally, a short discussion and conclusion is presented in Section 7.

2 Introduction to Tekkotsu

Tekkotsu is an application development framework for mobile robots. It provides (1) lower level primitives for sensory processing, smooth control of effectors, and event-based communication, (2) higher level facilities, including an hierarchical state machine formalism for managing control flow in the application, a vision system, and an automatically maintained world map, (3) housekeeping and utility functions, such as timers and profilers, and (4) the newly added Tekkotsu crew [12], including Lookout, MapBuilder, Pilot, and Grasper, which enables programmers to use the built-in higher level robotic functions such as map-making, localization, and path planning.

Tekkotsu is object-oriented, making extensive use of C++ templates, multiple inheritance, and polymorphism (operator overloading). To write a robot control program, users need to define subclasses that inherit from the Tekkotsu base classes, and override any member functions requiring customization. Two types of fundamental classes in Tekkotsu are behaviors and events. Users need to know the way to response or act when a behavior is constructed, activated, and deactivated, the way for a behavior to listen to events and to process events, and the ways to construct a state machine in Tekkotsu. Users also need to know the concepts of generator, source, and type of an event. Furthermore, when using the Tekkotsu crew, users need to know how to use different types of maps, how to localize the robot, how to detect and/or move to an object of interest, and how to get the location and shape information of objects of interest.

In a Tekkotsu state machine, each state has an associated action: speak, move, etc. and transitions are triggered by sensory events, timers, or user's signals. State nodes are behaviors. Entering a state is activating it, and leaving a state is deactivating it. Transitions are also behaviors. A transition starts to work whenever its source state node becomes active. Transitions listen to sensors, timer, or other events, and when their conditions are met, they fire. When a transition fires, it deactivates its source node(s) and then activates its destination node(s).

A shorthand notation is used instead of C++ code to build state machines. The shorthand notation includes the state node definition, the transition definition, and the state node class definition. The shorthand is turned into C++ code by a state machine compiler.

Example 1 lists a Tekkotsu state machine code that describes a behavior similar to a 2-bit free-running counter circuit. It says zero, one, two, and three and then backs to zero again. This application defines a state machine with four state nodes and four timer transitions, each with 1 second period. SpeechNode is a built-in node class in Tekkotsu. Note that there are many built-in node classes and transitions in Tekkotsu. Users can also define their own node classes, but rarely need to define their own transition classes.

```
#include "Behaviors/StateMachine.h
$nodeclass Counter4 : StateNode {
  $setupmachine {
    zero: SpeechNode("zero")
    one: SpeechNode("One")
    two: SpeechNode("two")
    three: SpeechNode("three")

    zero =T(1000)=> one =T(1000)=> two
    =T(1000)=> three =T(1000)=> zero
  }
}
REGISTER_BEHAVIOR(Counter4);
```

Example 1: State Machine Code of a 2-bit Counter

In addition to the concepts mentioned above, the following three basic skills of programming with the Tekkotsu are very important: (1) how to transit from one state to multiple states simultaneously so as to support parallel actions or behaviors, (2) how to transit from one state to one of multiple states based on different conditions so as to make a conditional transition, and (3) how to pass and/or share data among states so as to provide approaches of the data flow and the memory.

```
$nodeclass Counter4R : StateNode {
  $provide int b(1);
  enum Next {
    forward,
    backward
  };
  $nodeclass ButtonPressNode : StateNode {
    virtual void doStart() {
      erouter->addListener(this,EventBase::button);
    }

    virtual void doEvent() {
      $reference Counter4R::b;
      b = 1-b;
    }
  }
  $nodeclass NextNode() : StateNode : doStart {
      $reference Counter4R::b;
      if (b==1)
         postStateSignal<Next>(forward);
      else if (b==0)
         postStateSignal<Next>(backward);
  }
```

Example 2: Partial Code of 2-bit Counter with Reverse

Example 2 lists a part of state machine code that describes a behavior of 2-bit counter with a reverse button. The counter will count in a forward order 0, 1, 2, 3 initially. Whenever the button is pressed, the counting order will be reversed. In this example, the ButtonPressNode node class is defined to keep track the button event. Whenever the button is pressed, the shared variable b will change its value. Note that the button press node will be always active. Thus, this node is concurrent with other state nodes. To this end, a parallel transition from the start node like startn =N=> {zero, button} is needed. The NextNode node class is defined to decide the next state based on the current order (forward or backward) via the value of the shared variable b. Thus, one of the two state transition signals is posed according to the value of b.

3 Locate Color Balls Inside a Maze

The project is taken from the robotics competition task held along with the 2nd Annual ARTSI Student Research Conference [7]. This task is to get an iRobot Create/ASUS robot [2] to navigate and localize itself within a maze, to announce detected objects and their locations in the maze.

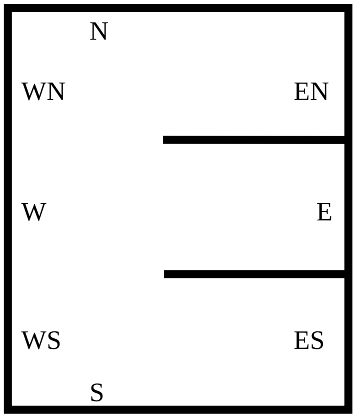


Figure 1: E-Maze with 8 Navigation Markers

The maze is shaped like the letter E as shown in Figure 1: a long corridor with three alcoves branching off from it. There is a bicolor marker at each end of each alcove or corridor, for a total of 8 markers: NE, N, NW, W, SW, S, SE, and E. Objects placed in the alcoves will be red, blue, or green balls, one per alcove. The project is graded on visiting each alcove of the maze, announcing detected objects and their locations, delivering a final report, and the overall run time.

This project is divided into three parts: (1) Travel though the maze by modifying the “following wall” behavior, (2) Travel though the maze and announce the balls the robot detects, and (3) Travel though the maze and report the balls the robot detects and their locations.

In Part 1, students need first to run a sample Tekkotsu’s wall following program. Students will notice that the robot will not able to follow the wall unless its right side is placed near a wall, and the robot will not stop. Then, students are required to modify the sample code so that the robot will perform (a) go to a wall, (b) follow the wall, and (c) stop after a while. So the robot can travel through the maze along the walls regardless where the robot is placed inside the maze.

In Part 2, a sample Tekkotsu state machine code of looking for balls is given to students. This behavior will announce the balls in front of the robot. Students need to test the code with several runs, each run with different sets of color balls and different lighting conditions. Students may need to change the ASUS camera settings through Tekkotsu. Then, students are asked to add the looking for balls behavior in the state machine code in Part 1. So a robot can look for balls while it travels through the maze. Note that looking for balls and traveling through the maze are two parallel behaviors and the former behavior will send a signal to the latter behavior to stop the robot after the robot finds all three balls. A state machine diagram to accomplish this task is given to students.

Please note that the robot may see the same ball several times. So, the robot must have memory to remember which ball it has already seen. Meanwhile, the robot should also remember the ordering in which it sees each of the three balls. To this end, a scheme of encoding the three colors is provided to students. In this scheme, Red is labeled as 1, Blue is 2, and Green is 4. A set of colors, for example, {Red, Green} will be labeled as 5 (1+4). A pair of colors, for example, (Blue, Green) will be labeled as 24 (2×10+4). A 3-tuple of colors, for example, (Blue, Green, Red) will be labeled as 241 (2×100+4×10+1).

In Part 3, a sample Tekkotsu state machine code of visiting markers is given to students. In this behavior, the robot will search for a bicolor marker by turning in its place and then move towards the marker; if the robot cannot find a marker after a while, it will move forward. In both cases, the robot will move forward and hit a wall of the maze. Students need to test this code first with the robot placed in different locations inside the maze and different lighting conditions. Students may need to change the ASUS camera settings.

Then, students are asked to modify the sample state machine code so that the robot will only find the bicolor marker at south, at southwest, at west, or at northwest. This means that the markers at the remaining four locations are ignored. Finally, students are asked to add the modified visiting marker behavior into the state machine code in Part 2. Then, students need to modify the announcement and the final report to include the location information of the detected balls. Note that the first ball the robot sees must be in the south alcove, the second must in the middle alcove, and the third must be in the north alcove, if one of the four bicolor markers at south, southwest, west, or northwest is found.

In addition to the above guidance, students need to know how to deal with uncertainty and failure. For example, bicolor markers are difficult to detect (false negative) and background blobs are easily recognized as a color ball (false positive).

4 Locate AprilTags Inside a Maze

The project is taken from the robotics competition held along with the 3nd Annual ARTSI Student Research Conference [8]. This task is the same as the one in Second 3 except the bicolor makers are replaced by the AprilTags [11] and the three color balls are replaced by three cubes with each covered by different AprilTags. Note that there are 20 navigation markers (AprilTags) on the walls of the E-maze as shown in Figure 2 so as to support a better result of the robot localization.

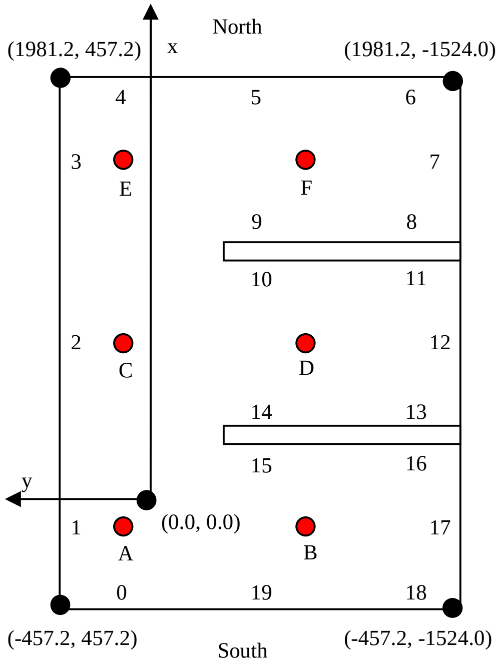


Figure 2: E-Maze, Twenty Navigation Markers (0-19), Six Waypoints (A-F), and World Coordinates of the Four Corners

In Section 3, each bicolor marker is treated as a topological navigation marker and the measurements of the maze are not used at all. On the other hand, in this project, a metric map of the maze as shown in Figure 2 is used in the robot localization. Meanwhile, moving robot to a particular waypoint inside the maze can be done easily by calling a pilot function that is newly added into Tekkotsu.

This project is divided into four parts: (1) Memorize and report which object the robot has observed in each alcove, (2) Look for the object in each alcove, (3) Drive robot to a waypoint in the maze, and (4) Putting it together.

In Part 1, students need first to run a sample Tekkotsu's pilot demonstration program. Students can use commands provided in this program to drive the robot forward or making a turn, check the robot location, localize the robot, and look AprilTags that are facing to the robot and report their IDs.

In this part, students are asked to add a command in this sample code to report which object the robot has observed in each alcove. Here, the robot needs to figure out which object in which alcove. We assume that when the robot is seeing an object in an alcove, it will also see at least one navigation marker on the wall of the same alcove. This may generally not be true. But, users can use commands to make the robot facing to an object and at least one marker at the same time. Note that students need to define three shared variables to remember the objects the robot has observed in each alcove.

In Part 2, students are asked to add a command to look for an object and its location. Now, we assume that the robot is inside an alcove, but it may not be facing an object. This new command allows the robot to look for an object for several times by making several turns until it finds one as well as its location or fails to find.

In Part 3, students are asked to add a goto <x y> command to drive the robot from its current position to location (x, y) inside the maze. Please note that the world coordinates of points in the maze should be used in the goto function. The coordinates of the four corners are shown in Figure 2. Note that the coordinates are using millimeters as measurement unit. Students can use the built-in path planning and execution function through using the Tekkotsu pilot to implement the goto command. Or, students can implement the goto function on their own. In this case, we assume there is no obstacle in between the robot current location and the target location.

Due to the uncertainty, the robot may not be able to be close enough to the target location. In this case, the goto function should redo itself again until the robot is close to the target within a threshold distance (for example, 200 mm). If the robot is still not able to reach the target after redoing 3 times, the goto function should be stopped and return a failure.

Due to the same reason as above, the robot may hit walls before reaching to the target location. In this case, the robot should backup a little bit and then the goto function should redo itself again. If the robot is still not able to reach to the target after redoing 3 times, the goto function should be stopped and return a failure.

In Part 4, students are asked to put them together by creating a node class that takes a role of subtask scheduler. For example, a schedule can be goto A, goto B, look for object, goto A, goto C, goto D, look for object, … and finally report which object the robot has observed in each alcove. It is obvious that this schedule will let the robot find the object at each alcove, if each subtask is successfully executed.

5 Push Canisters into Pen

The project is taken from the robotics competition held along with the 4th Annual ARTSI Student Research Conference [9]. As shown in Figure 3, five red canisters will be scattered around a 2-by-2 meter space. Near the center of the space is a 0.75-by-0.75 meter "pen" drawn with blue masking tape. The task is to get a robot with two added aluminum paddles to locate the canisters and push all of them into the pen as quickly as possible.

This project is divided into five parts: (1) Locate and move to the center of the pen, (2) Locate a canister and move to the canister, (3) Push one canister into the pen when the robot is facing a canister and a corner of the pen, (4) Push all canisters into pen, assuming that the robot is able to see a pen corner and a canister outside the pen at the same time, (5) Push all canisters into the pen with no assumption in (4).

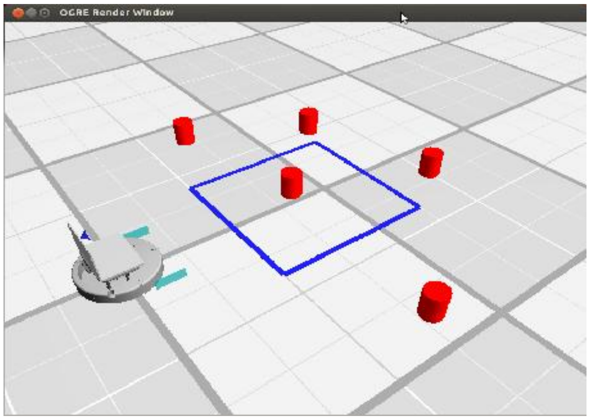


Figure 3: Push Canisters into Pen

In part 1, a sample state machine code is given to students. This code will compute the center of the pen based on a corner of the pen and then drive the robot to the center of the pen, when the robot is facing a pen corner. Students are asked to study the code and run the code in the following settings: (1) the robot facing more than one pen corner, (2) the robot facing only one pen corner, and (3) the robot facing no corner. Then, students are asked to modify the code so that the robot can search for a pen corner if no pen corner is visible to the robot, and then compute the center of the pen and drive the robot to the pen center. In case that there is no pen at all, the robot should report "no pen".

Part 2 is very similar to Part 1. A sample state machine code that drives the robot to a canister that the robot is looking at is given to students. Then, students will add a search function so that the robot will search a canister first and then drive to it.

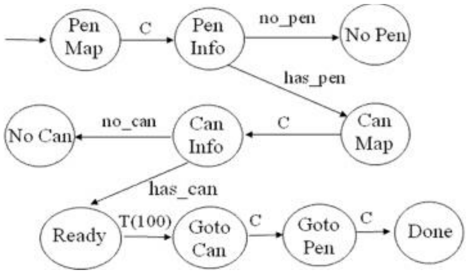


Figure 4: State Machine Diagram in Part 3

In Part 3, students are asked to write a state machine code to push one canister into the pen center according to a given state machine diagram as shown in Figure 4. If the robot can see a pen corner and a canister outside the pen simultaneously, then the robot will move to the canister first and then push it into the pen. If the robot cannot see either any pen corner or any canister, then it reports “No Pen” or “No Can". Meanwhile, students are asked to figure out the formulae to compute the angle and distance to the canister and the angle and distance to the pen center in the robot's local shape space (local map) as shown in Figure 5.

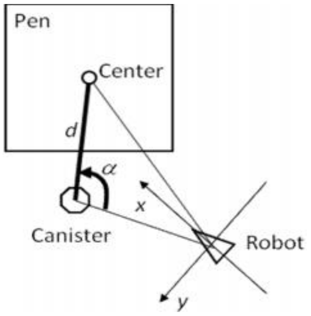


Figure 5: Center of Pen and Canister in Local Map

In Part 4, the code in Part 3 is to add a search function first according to a given state transition diagram shown in Figure 6. If the robot can see a corner of the pen and a canister simultaneously, then the robot will move to the canister first and then push it into the pen and say “Done”. If the robot does not see either a corner or a canister, then the robot will makes a turn a little bit (say, 30 degree) and then continue to look for a corner and a canister. If the robot is still unable to find a corner and a canister simultaneously after making several turns (say, 12), then the robot should stop searching and report “Fail”.

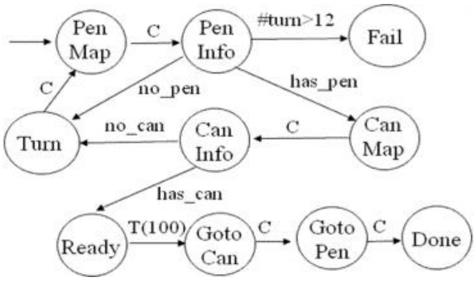


Figure 6: State Machine Diagram in Part 4

Next in Part 4, students are asked to extend the above code further to push all canisters into pen. After one canister is pushed into the pen, the robot should backup and get out of the pen, and then repeat for a next canister until all canisters are inside the pen. Note that the distance between a canister and the pen center can be used to judge if the canister is inside the pen or not.

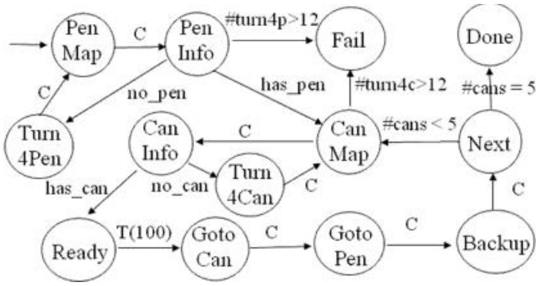


Figure 7: State Machine Diagram in Part 5

In Part 5, the assumption that the robot is able to see a canister and a pen corner simultaneously is removed. Thus, the robot should detect a pen corner and compute and memorize the pen center first. Then, whenever the robot moves, the coordinate transform of the pen center should be performed so as to keep the pen center always inside the current local shape space. Figure 7 shows the state transition diagram that students should implement for their final version of the code.

6 Pickup Canisters

The project is taken from the 5th robotics competition held along with the Fifth Annual ARTSI Student Research Conference [10]. As shown in Figure 8, this task will be done using a Calliope2SP robot in a 1 meter by 2 meter rectangular arena with white walls and an AprilTag on one short wall marking a goal location. Three colored cylinders will be scattered around the arena, far from the goal. The robot must locate each cylinder, pick it up using its gripper, and transport it to the goal location, defined as by a 2x2 foot box below the AprilTag.

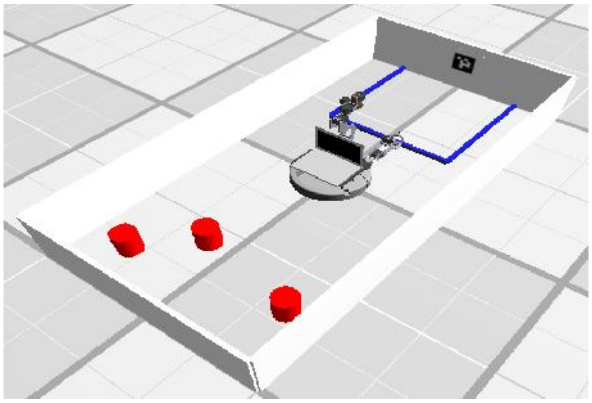


Figure 8: Pickup and Putdown Canisters

This project is divided into five parts: (1) Locate an AprilTag and move close to it, (2) Pickup a canister when it is placed in between the two fingers of the gripper, (3) Locate a canister, move to the canister, and pickup the canister, (4) Locate a canister, move to it, pickup it, transport it to the goal location, and drop it, and (5) Add memory to the robot.

In Part 1, a sample state machine code is given to students. This code will obtain the location of AprilTag in front of the robot and drive the robot to this location. Students are asked to test this code and then modify the code so that the robot can search for an AprilTag, if no tag is visible to the robot. It is also possible that there is no AprilTag at all. In this case, the robot should report that there is no tag.

In Part 2, a state machine code is given to students. This code will make the robot in a getting ready posture by using the Tekkotsu's Dynamic Motion Sequence Node. In this posture, the robot's head looks ahead, the robot's arm points ahead, and the robot's gripper is open and in a low position. Students are asked to study the code and then add two more motion sequences: Pickup and Putdown so that if a canister is placed in between the gripper's two fingers, the robot should be able to pick the canister up. Note that students may need to test the code several times to adjust the distance in between two fingers until the robot can pick the canister up.

In Part 3, a sample state machine code is given to students. This code will detect canisters and drive the robot to the canister that is closest to the robot. If there is no canister, the robot will report there is no can. As shown in Figure 9, the grippers is northeast to the robot center. In order to make the two fingers of the gripper close enough to the canister after the robot moves to the canister, the robot should follow the route with red color in Figure 9. To this end, students need to give the formulae to calculate D, β, and θ. Note that (xg, yg) can be obtained by measuring the Calliope2SP robot, d is a constant which is a little longer than the gripper's finger, and (xc, yc) can be obtained by using Tekkotsu. Note that students need to test this code and their formulae multiple times to adjust their formulae. Finally, students are asked to write a new state machine code so that the robot can locate a canister, move to the canister, and pickup the canister.

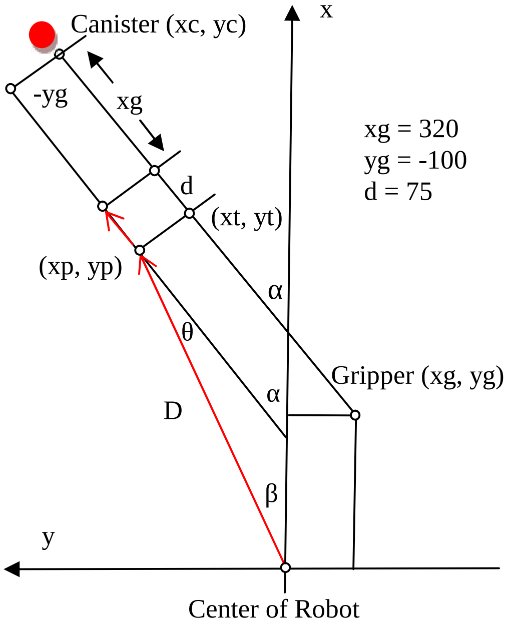


Figure 9: Center of Robot, Gripper, and Canister

In Part 4, students are asked to write a state machine code that simply combine their codes in Part 3 and Part 1 together so that the robot can locate a canister, move to it, pick it up, and transport it to the goal location that is close to the AprilTag and drop it. After testing this code, students will extend it so that the robot will continue its work until it finishes to pickup, transport, and drop all three canisters. Note that students are asked to make sure the robot will not pick up a canister again that has been already transported to the goal location. This may need some assumptions if the robot has no memory.

In Part 5, students are asked to add a shared variable to memorize the AprileTag location. Therefore, the robot just needs to search for the AprilTag once and it will know a canister has been already transported to the goal location, if the canister is close enough to the AprileTag. In this case, whenever the robot moves, the coordinate transform of the AprilTag location should be performed so as to keep it always inside the current local shape space. Figure 10 shows the state transition diagram that students should implement for their final version of the code.

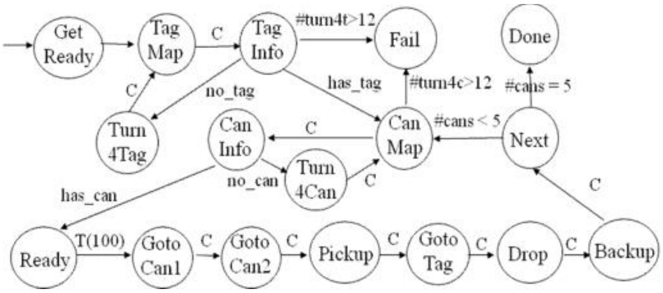


Figure 10: State Machine Diagram in Part 5

7 Discussion and Conclusion

Our robotics team at JSU has participated in the robotics competitions held along with the ARTSI Annual Student Research Conferences four times since 2010. Three of the four competition tasks have already been transformed into our major robot programming lab projects. Our robotics course has been offered in the fall semesters four times starting from 2009 [13]. It covers all major topics on intelligent mobile robots, including robot control architectures, navigation, localization, planning, sensing, and uncertainty.

The ARTSI robotics competitions have provided a unique opportunity to encourage undergraduate students from non-traditional backgrounds in the study of robotics in areas that are relevant to society. They have also provided rich materials for robotics education. Coaching students for preparing for such competitions are challenging and time-consuming, but it also gives coaches an opportunity to learn how to help their students to solving the challenging problems.

The robot programming development tool Tekkotsu builds a set of high-level interacting software components. Thus, it relieves a programmer of the burden of specifying low-level robot behaviors [12]. This makes it possible to accomplish challenging robotic applications and to teach and practice more on robotics rather than programming details. In addition, the 3-D simulation software Mirage can be used along with the Tekkotsu for the simulation. But, there is a large learning curve to master the Tekkotsu fundamentals and its high-level software components. The Tekkotsu tool is under the constant construction and changes a lot over time.

One of the biggest problems we were facing in the robotics competitions is uncertainty. The environments have a huge impact on the robot performance. All our robotics competition tasks are related to the robot vision. The robots are required to recognize bicolor markers, color balls, and color cylinders. However, these jobs are sensitive to the lighting conditions in the competition site. There are a lot false negatives for bicolor makers and a lot false positives for color balls and cylinders. But, the AprilTags detection is very accurate and stable. This makes the robot localization more accurate. So, we applied the Tekkotsu built-in robot localization function in the second robotics competition.

Finally, there could be other alternative ways to solve the robotics competition problems. For examples, the world map could be used in the third and fourth competitions. Therefore, it is easy for robot to memorize the locations of the pen center and the AprilTag. But, maintaining a world map in Tekkotsu makes it slower and adds a little more uncertainty. Thus, we selected to use the local map. This also leave a chance to let users to keep track the objects they need.

Acknowledgments

This work was supported by National Science Foundation award CNS-1042326. The author thanks Dr. David Touretzky for his help on the Tekkotsu.

8 References

[1] ARTSI Alliance, Available at http://artsialliance.org/

[2] D. S. Touretzky, *iRobot Create/ASUS Notebook Platform for Tekkotsu*, Available at http://chiara-robot.org/Create/

[3] D. S. Touretzky, *Calliope Mobile Manipulation Platform for Tekkotsu*, Available at http://chiara-robot.org/Calliope/

[4] D. S. Touretzky, *Calliope Robot*, Available at http://wiki.tekkotsu.org/index.php/Calliope_Robot

[5] E. J. Tira-Thompson, G. V. Nickens, and D. S. Touretzky, *Extending Tekkotsu to new platforms for cognitive robotics*, Proceedings of the 2007 AAAI Mobile Robot Workshop, pp. 47-51, Menlo Park, CA.

[6] D. S. Touretzky, and E. J. Tira-Thompson, *Tekkotsu: A framework for AIBO cognitive robotics*, Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI-05), volume 4, pp. 1741-1742. Menlo Park, CA, 2005.

[7] ARTSI Alliance, *2010 Robotics Competition*, Available at http://artsialliance.org/2010-Robotics-Competition

[8] ARTSI Alliance, *2011 Robotics Competition*, Available at http://artsialliance.org/2011-Robotics-Competition

[9] ARTSI Alliance, *2012 Robotics Competition*, Available at http://artsialliance.org/2012-Robotics-Competition

[10] ARTSI Alliance, *2013 Robotics Competition*, Available at http://artsialliance.org/2013-Robotics-Competition

[11] E. Olson, *AprilTag: A robust and flexible visual fiducial system*, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2011), Shanghai, China, May 2011.

[12] D. S. Touretzky, and E. J. Tira-Thompson, *The Tekkotsu "Crew" Teaching Robot Programming At A Higher Level*, Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10), pp. 1908-1913, Atlanta, GA, 2010.

[13] Xxuejun Liang, *Introduction to Robotics*, Available at http://www.jsums.edu/robotics/

Engaging Underrepresented Students in Computer Science: Examining the Effectiveness of a 5-week Computer Science Course in the SMASH Summer Academy

Alexis Martin[1] **and Allison Scott**[2]
[1]Level Playing Field Institute, Oakland, California, USA
[2]Level Playing Field Institute, Oakland, California, USA

Abstract – *To alleviate barriers facing underrepresented students of color in pursuing computer science, the Level Playing Field Institute implemented a computer science course within its SMASH summer program, serving n=165 low-income, high school students of color. Aiming to examine the effectiveness of the intervention, this study utilized a pre- and post-program survey to measure student growth in computer science knowledge and aspirations. Results indicated that self-reported computer science skills and knowledge, and computer science aspirations significantly increased over the 5-week course. There were no significant differences by gender suggesting that the courses, curriculum, and instruction were effective for both males and females. Given current constraints within K-12 public education, this model for engagement and exposure to computer science in an out-of-school setting among students often marginalized from opportunities in computer science can inform strategies and interventions to increase participation, engagement, and retention in computer science among students of color.*

Keywords: African American, Latino/a, Computer Science Aspirations, Out-of-School Learning, Expanding Computer Science Opportunities

1 Introduction

Despite projections that the fastest-growing and highest-paying occupations of the future are in computing fields [1], underrepresented students of color continue to comprise an alarmingly small percentage of individuals taking computer science courses, pursuing computer science degrees, and entering the computing workforce. From 2008-2018, computer and mathematical occupations are expected to add over 785,000 new jobs, growing twice as fast as the average rate for all occupations (22% growth in 10 years) and being one of the top 10 fastest growing occupational groups [2]. In addition, computer and mathematical occupations are among the highest-paying occupations, with an average mean annual salary of $78,730 [3].

In California, home of Silicon Valley, and the state with the highest numbers of projected annual computing job openings [4], vast disparities by race and gender exist in access, opportunity, and outcomes in computer science from K-12 to the workforce. Across the entire state of California, just 45 African American students and 314 Latino students took the AP Computer Science A exam in 2012. Despite the fact that African American and Latino students comprise a combined 57% of the high-school aged population in California, they accounted for just 9% of all students taking the AP Computer Science A exam in 2012 [5][6]. Further, African American and Latino students combined account for just 17% of all computer science Bachelor's degrees conferred, 7% of all computer science Ph.D.'s conferred and 9% of the computing workforce nationwide [7][8]. These disparate outcomes can be linked to the following barriers impacting the pursuit of computer science among underrepresented students of color: (1) Lack of access to rigorous computer science courses, (2) Lack of engaging and relevant curriculum, and (3) Social and psychological barriers affecting perceptions of computer science.

1.1 Access to Rigorous Computer Science Courses

The Computer Science Teachers' Association recommends that all students participate in a sequence of 3 computer science courses at the high school level [9]. Yet, in many schools attended by students of color, if any computer courses are offered at all, they tend to focus on basic computer applications or typing [10][11]. Access to advanced computer science courses also varies by race/ethnicity, and schools serving high numbers of students of color are much less likely to offer rigorous computer science courses than more affluent schools [12]. Without access to Advanced Placement computer science courses, students are 8 times less likely to pursue computer science studies in higher education [13].

1.2 Curriculum

Within computing classes at the secondary level, curriculum can vary tremendously from courses focusing on typing and basic computer applications (e.g., Microsoft Word, Excel) to creating mobile apps, to a strict focus on programming in Java [10][11]. A major challenge in the rigorous Advanced Placement computer science courses is making the content engaging, interesting, and relevant, especially when attempting to engage underrepresented students of color who have not typically enrolled in such courses [10][14]. Promising practices in revising computer science curriculum to engage a broader group of underrepresented students of color include the development of a new Computer Science Principles AP course to expand content beyond Java to provide an engaging, inspiring, and rigorous advanced course [15], the development of the Exploring Computer Science curriculum which introduces students to foundations of computer science within an inquiry-based and culturally relevant context [14], and additional interventions drawing upon theories of culturally relevant pedagogy [16] to engage students of color in computing [17][18][19]. These promising practices, however, are generally positioned within high schools and there remains a need for rigorous out-of-school computer science engagement to serve students without access to computer science courses within their high schools.

1.3 Social/Psychological Barriers

Even with access and opportunity, students of color are often discouraged or disinterested in studying computer science due to misconceptions about computer science as a discipline and career path [20][21][22], perceptions of computer science as lacking interpersonal or social relevance [23][24][25], lack of diverse role models [19][26][27][28], isolation [29], stereotype threat associated with being a member of a marginalized group in math and science [30], and the presence of stereotypical cues within computer science environments [31][32].

Interventions focusing not just on increasing access to rigorous computer science courses, but also on reshaping the pedagogical practices and curricula within computer science classrooms and alleviating social/psychological barriers have demonstrated positive outcomes among underrepresented students of color [14][19]. To address the barriers facing underrepresented populations and expand pathways to college and careers in computer science, the Level Playing Field Institute implemented an introductory computer science course within its Summer Math and Science Honors (SMASH) Academy, serving high school students of color in California. This study seeks to expand understanding of effective interventions for improving computer science outcomes among high school students of color by examining the effectiveness of the pilot introductory computer science course within the SMASH Academy. Specifically, this study aims to examine the following research questions: (1) What impact does participating in a 5-week computer science exposure course have on the computer science knowledge and computer science college and career aspirations of underrepresented high school students of color? and (2) Does course impact vary by demographic variables (e.g., gender, SES)?

2 Methodology

2.1 Program Context

The Summer Math and Science Honors Academy (SMASH) is a 3-year, 5-week summer STEM enrichment program serving predominantly low-income and first-generation underrepresented high school students of color. The SMASH program was designed in 2004 to alleviate the barriers facing underrepresented students of color in STEM by providing a rigorous, college preparatory summer academy to prepare high school students demonstrating interest and aptitude in studying STEM subjects to enter and persist in STEM studies at four-year universities. SMASH currently holds its summer programs at four university campus sites in Northern and Southern California (UC Berkeley, Stanford, UCLA, and USC).

Each SMASH site includes three cohorts of students, those entering 10th, 11th, and 12th grades. To provide a comprehensive program which prepares students to be academically prepared for STEM studies in college, SMASH programming includes core math (Algebra II, Pre-Calculus, Calculus) and science courses (Biology, Chemistry, Physics), and elective courses including Science Writing, Mobile Apps, and Entrepreneurship. In addition, SMASH offers a college success course which guides students through the process of researching colleges, completing college applications, and researching and applying for financial aid. Finally, to counteract social/psychological barriers, SMASH exposes students to diverse STEM role models through a weekly "Speaker Series," employs a diverse teaching staff, and builds community and networks of support among SMASH students.

2.2 Participants

The study sample included n=165 high school students participating in SMASH at four sites in Northern and Southern California (UC Berkeley, Stanford, UCLA, and USC; See Table 1). Students were selected to participate in the summer program based on criteria including grades, performance in math and science courses, interest in STEM, and teacher recommendations. All students were members of ethnic groups underrepresented in the STEM fields, with the majority being Latino (57%) and African American (25%), with other underrepresented groups and multiracial students comprising the remaining 18%. The sample was equally divided based on gender, with 52% of scholars being female. The 3-year program serves students starting in the summer prior to their sophomore year through the summer prior to their senior year. The sample included rising 10th graders (66%), rising 11th graders (18%), and rising 12th graders (16%). The vast majority of students were low-income (82%) and the first in their family to complete college (81%). All students attended high schools in the San Francisco Bay Area or in the Los Angeles Area; with 94% attending public or charter high schools. Academic data indicates that the average math grade for SMASH students is a "B+", the average

California Standards Test (CST) math score is 420, and the average CST science score is 428; both the math and science CST scores fall into the Proficient/Advanced range.

Table 1. SMASH Academy Demographic Data

		Gender		Race/Ethnicity			Socioeconomic Indicators	
SMASH Site	**(n) Students**	**Male**	**Female**	**Af-Am**	**Latino**	**Other**	**% FRPL**	**% First-in-Family**
UC Berkeley	28	50%	50%	18%	64%	18%	79%	86%
Stanford	79	51%	49%	28%	46%	26%	77%	81%
UCLA	29	45%	55%	41%	52%	7%	83%	72%
USC	28	43%	57%	7%	86%	7%	96%	86%
Total	165	48%	52%	25%	57%	18%	82%	81%

Methodological note: The total number of SMASH students attending a SMASH program in 2012 was n=214. Of these, n=165 (78%) both took a Computer Science course in the summer of 2012 and completed pre-SMASH and post-SMASH surveys. The data presented within this paper does not include students with unavailable data (n=7, 3%) or those who did not take Computer Science in 2012.

Just 36% of SMASH students attended a high school which offered any form of computer science course in 2011-2012. Further, just 17% of SMASH students attended a high school in which AP Computer Science A is offered (UCOP, 2013). Not surprisingly, 69% of SMASH students in 2012 indicated that they had never previously taken a computer science course of any kind.

2.3 Procedures

All students completed a pre-SMASH and post-SMASH survey administered online through SurveyMonkey at the beginning and at the end of the program (June and August 2012). The full survey contained items measuring 21 variables, including the target variables for this study which included a computer science concept/skills inventory, computer science coursetaking aspirations, and computer science college and career aspirations. Surveys took approximately 45 minutes to complete at both pre- and post-SMASH administrations. The mean of each scale at pre-SMASH at post-SMASH was calculated and a paired-samples T-test was run to determine if mean values were significantly different in pre- and post-SMASH conditions. Analyses of Variance were used to determine if significant differences in pre- and post-SMASH computer science outcomes existed based on demographic characteristics (e.g., race, gender).

2.4 Study Instruments

Study instruments consisted of a comprehensive online survey which included the two variables examined within this project: (1) Computer Science skills/knowledge, and (2) Computer Science aspirations. The Computer Science skills/knowledge scale (α=.85), consisted of an inventory of 5 items assessing student's self-reported skills and knowledge about basic computer science topics. The 5 topics were developed in collaboration with the course instructor and were consistent with the curriculum topics being covered within the 5-week course. Items included, "I know how to design and program a basic user interface," "I know how to structure a database," and "I understand the basic processes of debugging" and were scored on a 5-point Likert scale (Strongly disagree-Strongly agree).

The Computer Science Aspirations scale (α=.92), consists of two items assessing students interest in pursuing a computer science degree in college and interest in pursuing a career in computer science. Items included, "I plan to pursue a degree in computer science," and "I plan to pursue a career in computer science," and were scored on a 5-point Likert scale (Strongly disagree-Strongly agree). The survey also included one item assessing interest in pursuing a computer science course in high school. Demographic data for each scholar was collected upon admission to the program, and included: Race/Ethnicity, Gender, Grade, High School, FRPL eligibility, First-in-Family status, Mother and Father's Highest Education Level, among others.

2.5 Course Description

Computer Science was offered to all first, second, and third year SMASH students at Stanford, USC, and UCLA, and to all first year SMASH students at UC Berkeley. The curriculum was adapted from the Exploring Computer Science curriculum, of the Computer Science Equity Alliance at UCLA (www.exploringcs.com). The curriculum integrates culturally relevant pedagogy and inquiry-based learning with content standards to ensure students are engaged and are mastering content while understanding the application of concepts and skills to their everyday lives.

3 Summary of Findings

3.1 Computer Science Skills and Knowledge

On average, students began the computer science course with low levels of self-reported knowledge and skills in computer science (M=1.39, SD=.65). By the end of the course, students reported much higher levels of knowledge of basic computer science concepts (M=2.93, SD=1.08). Students demonstrated a significant increase in computer science skills and knowledge from pre- to post-SMASH conditions (t(155) = -14.37, p=.00; Table 2). When examining gender differences, males' self-reported skills (M=1.22, SD=1.01) demonstrated a larger increase than females' skills (M=.98, SD=.88), but the difference was non-significant (F(154)=2.42, p=.12). In addition, no significant differences were found in pre-post Computer Science skills by FRPL-eligibility status, high school grade, or race/ethnicity.

3.2 Computer Science Aspirations

Prior to taking the course, students had relatively high levels of Computer Science college and career aspirations (M=2.93, SD=.83), and these aspirations increased slightly by the end of the course (M=3.04, SD=.95). Students demonstrated a significant increase in computer science college and career aspirations from pre- to post-SMASH conditions (t(154)=-1.61, p=.10; Table 2). Female students demonstrated a slightly larger increase in computer science aspirations (M=.17, SD=.87) than their male counterparts (M=.05, SD=.82), but the difference was non-significant (F(153)=.77, p=.38). There were no significant differences found in pre- and post-Computer Science aspirations by race/ethnicity, or FRPL-eligibility status.

Table 2. Paired-Samples T-Test: Pre- and Post-SMASH Computer Science Outcomes

Scale	Pre-SMASH (Mean)	Post-SMASH (Mean)	Diff.	SD	t	Sig. (2-tailed)
Computer Science Skills/ Knowledge	1.39	2.48	-1.09	.95	-14.37	.00**
Computer Science Aspirations	2.93	3.04	-.11	.84	-1.62	.10*

p<.00**, p<.10*

4 Conclusion and Implications

Preliminary evidence from the first year of the SMASH computer science course indicates that the 5-week course significantly increased self-reported computer science skills and knowledge, and computer science aspirations among underrepresented high school students. There were no significant differences by gender, suggesting that the courses, curriculum, and instruction were effective for both males and females. To expand understanding of the effectiveness of this computer science intervention, future research can utilize comprehensive formative assessments in addition to self-report surveys and can further examine with students and instructors the most and least engaging elements of the computer science introductory content in order to understand how experiences can lead to interest, aspirations, and persistence in computer science. In addition, further research is needed to examine the longitudinal impact of exposure to computer science interventions, such as the SMASH 5-week course, on the high school coursetaking and likelihood of majoring and persisting in computer science in post-secondary education. Given current constraints within K-12 public education, this model for early engagement with and exposure to computer science in an out-of-school setting among students often marginalized from opportunities in computer science can inform strategies and interventions to increase participation, engagement, preparation, and retention in computer science among students of color.

5 Acknowledgments

We would like to thank the individual, corporate, and foundation sponsors who provide support for the Level Playing Field Institute's research and programming and the SMASH Academy scholars for participating in this study.

6 References

[1] Bureau of Labor Statistics. (2009). Science, technology, engineering, and mathematics (STEM) occupations. *The Editor's Desk*.

[2] Lacey, T. & Wright, B. (2009). Employment Outlook: 2008-18: Occupational Employment Projections to 2018. *Bureau of Labor Statistics, Monthly Labor Review*, 82-123.

[3] Bureau of Labor Statistics (2011). National Occupational Employment and Wage Estimates: May 2011. *Occupational Employment Statistics, Computer and Mathematical Occupations*.

[4] National Center for Women in Technology (2011). Computing Education and Future Jobs: A Look at National, State, and Congressional District Data. Retrieved from: http://www.ncwit.org/sites/default/files/resources/csedjobsreport.pdf

[5] California Longitudinal Pupil Achievement Data System (CALPADS, 2011). Student Enrollment by Gender, Grade, and Ethnic Designation (2011-2012). Retrieved from: http://dq.cde.ca.gov/dataquest/

[6] College Board (2012). The 8th Annual AP Report to the Nation.Retrievedfrom: http://media.collegeboard.com/digitalServices/public/pdf/ap/rtn/AP-Report-to-the-Nation.pdf

[7] National Science Foundation (2012). Earned Bachelor's Degrees, by Citizenship, Field, and Race/Ethnicity: 2000-09. *Science and Engineering Indicators, 2012*.

[8] National Science Foundation (2013). Women, Minorities, and Persons with Disabilities in Science and Engineering, 2013.

[9] Computer Science Teachers Association. (2011). CSTA K-12 Computer Science Standards: Revised 2011. CSTA StandardsTaskForce.

[10] Goode, J. (2010). Mind the Gap: The Digital Dimension of College Access.

[11] Ryoo J., Margolis, J., Lee, C., Sandova, C., Goode, J. (2013).Democratizing computer science knowledge: transforming the face of computer science through public high school education, *Learning, Media and Technology*, DOI:10.1080/17439884.2013.756514.

[12] Margolis, J., Goode, J., Jellison Holme, J., & Nao, K. (2008). *Stuck in the shallow end: Education, race, and computing*. Cambridge: MIT Press.

[13] Mattern, K. D., Shaw, E.J. & Ewing, M. (2011). Is AP Exam Participation and Performance Related to Choice of College Major? New York: The College Board. Retrieved from: http://research.collegeboard.org/sites/default/files/info2go/2012/8/infotogo-2011-6-ap-participation-performance-major-choice.pdf

[14] Goode, J. and Margolis, J. (2011). Exploring computer science: A case study of school reform. ACM Trans.Comput. Educ. 11, 2, Article 12.

[15] Astrachan, O., Haynie, K., Stephenson, C., Diaz, L., & Briggs, A. (2010). Re-imagining the First Year of Computing. SIGCSE'10, March 10–13, 2010, Milwaukee, Wisconsin.

[16] Ladson-Billings, G., & Tate, W. F., IV. (1995). Toward a critical race theory of education. *Teachers College Record, 97(1),* 47-67.

[17] Eisenhart, M. and Edwards, L. (2004). Red-Eared Sliders and Neighborhood Dogs: Creating Third Spaces to Support Ethnic Girls' Interests in Technological and Scientific Expertise. *Children, Youth and Environments, 14(2),* 156-177.

[18] Scott, K.A., Aist, G., & Hood, D.W. (2009). CompuGirls: Designing a Culturally Relevant Technology Program. *Educational Technology, 49(6),* 34-39.

[19] Zimmerman, T. G., Johnson, D., Wambsgans, C., and Fuentes, A.(2011). Why Latino high school students select computer science as a major: Analysis of a success story. *ACM Trans. Comput. Educ. 11, 2, Article 10.*

[20] Badagliacco, JM. (1990). Gender and Race Differences in Computing Attitudes and Experience. *Social Science Computer Review*, *8(1),* 42-63.

[21] Carter, L. (2006). Why students with an apparent aptitude for computer science don't choose to major in computer science. *Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education*. 27–31.

[22] Margolis, J., Goode, J., & Bernier, D. (2011). The Need for Computer Science. *Educational Leadership*, February, 2011.

[23] Peckham, J., Stephenson, P., Harlow, L., Stuart, D., Silver, B., & Mederer, H. (2007). Broadening Participation in Computing: Issues and Challenges ITiCSE'07, June 23-27, 2007.

[24] Rich, L., Perry, H., and Guzdial, M. (2004). A CS1 course designed to address interests of women. In *Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education* (pp. 190-194). New York, NY: ACM Press.

[25] Williams, L., Layman, L., Slaten, K., Berenson, S., & Seaman, C. (2007). On the Impact of a Collaborative Pedagogy on African American Millennial Students in Software Engineering. 29th International Conference on Software Engineering (ICSE'07).

[26] Anderson, N., Lankshear, C., Timms, C., & Courtney, L. (2008). 'Because it's boring, irrelevant and I don't like computers':Why high school girls avoid professionally-oriented ICT subjects. *Computers & Education* 50, 1304–1318.

[27] Cain, C. (2012). Underrepresented Groups in Gender and STEM: The Case of Black Males in CISE. *SIGMIS-CPR'12, May 31–June 2, 2012.*

[28] Goode, J. (2008). Increasing Diversity in K-12 Computer Science: Strategies from the Field. SIGCSE'08, March 12–15, 2008.

[29] Moses, L.E. (1993) Our Computer Science Classrooms: Are They 'Friendly' to Female Students? *SIGCSE Bulletin, 25(3)*, 3-12.

[30] Steele, C. M., & Aronson, J. (1995). Stereotype threat and the intellectual test performance of African Americans. *Journal of Personality and Social Psychology, 69,* 797–811.

[31] Cheryan, S., Plaut, V.C., Davies, P.G., & Steele, C.M. (2009). Ambient Belonging: How Stereotypical Cues Impact Gender Participation in Computer Science. *Journal of Personality and Social Psychology, 97(7),* 1045-1060.

[32] Cheryan, S., Meltzoff, A., & Kim, A. (2011). Classrooms matter: The design of virtual classrooms influences gender disparities in computer science classes. *Computers & Education,* 57 (2011), 1825–1835.

Creating an Environment Supportive of Diversity in Computing

P. Mullins, D. Whitfield, D. Dailey, B. Gocal
Computer Science Department
Slippery Rock University
Slippery Rock, PA 16057, USA
paul.mullins@sru.edu

Abstract— *This paper describes the results of a PDE Keystone/Frederick Douglass Grant that the authors received to incorporate diversity into computing coursework in the areas of Computer Science, Information Systems, and Information Technology. Existing assignments created by professors from all three disciplines were examined for their inclusion of diversity, in its broadest sense. Where little or no diversity was found in course assignments, alternate assignments were proposed that included some degree of diversity.*

Keywords— *computer science; diversity; inclusive environment*

I. Introduction

The authors received a grant from the Frederick Douglass Institute to investigate the incorporation of diversity in computing curricula without adding curricular components. The authors envision the creation of a sample set of assignments that could be used not only within our department, but by other computing departments to create problems for students that were inclusive, that is, supported diversity by making it a normal part of the course work. This limited approach is not intended to suggest that a more inclusionary environment is not necessary or to be sought after; rather, it was intended to provide a means of showing faculty how some degree of diversity could be included in their courses with a minimum of effort or impact on how the class is taught.

Diversity was taken by the investigators in the broadest sense. That is, we included not only gender, people of color, and various ethnic backgrounds, but also included LGBT, religions, and the physically challenged. In this paper, underrepresented people is intended to include any of these.

In addition, the project provided an opportunity for our program to assess the current status of our program by looking at sample assignments submitted as part of an accreditation effort and to attempt to develop a methodology for the creation or modification of assignments to make them more inclusive.

The idea of diversity in the computing classroom is not new – many papers on the topic have been published. One example of a programming assignment used to help students learn the value of diversity and the analysis of the assignment shows the usefulness of using assignments as a conduit for diversity [1]. Many works have been published discussing techniques that attract and retain diverse populations of computing students where diversity includes minorities [2]. The national center for women and information technology [3] works to increase diversity in IT and computing. Works have been published for using widgets to support disabled learners [4].

A. Department Overview

There are three programs (majors) in the Department of Computer Science at Slippery Rock University: computer science (CS), information systems (IS) and information technology (IT). All three programs are accredited by the Computing Accreditation Commission of ABET (www.abet.org).

The department offers 37 courses that are used as requirements or electives in these three majors. One of these (CPSC 490: Independent Study) is used accommodate special situations that arise with students or scheduling. Two (CPSC 498b: Machine Learning and Robotics and CPSC 498c: Compiler Transformation) have not been offered in a number of years. In addition, there are several service-only courses that do not count toward any of the majors. 34 regularly offered courses were analyzed that count as a required or elective course in one or more of the three majors. Naturally, some of these courses count in more than one major. Table 1 provides the number of department courses used in each major. Of these 34 courses, five are also used as part of the university's liberal studies program and two for a basic university computer competency requirement.

TABLE I. BREAKDOWN OF THE 34 MAJOR COURSES

	Information Technology	Information Systems	Computer Science
Department courses	20	18	21
Required	12	11	10
Elective	8	7	11

Grant support provided by West Chester University of Pennsylvania's Frederick Douglass Institutes of the Pennsylvania State System of Higher Education.

II. METHODOLOGY

Directly addressing diversity, even on a college campus where open discussion and alternative views are honored, can be problematic. For example, one potential assignment called “The game of Choice”, refers implicitly to the question of whether sexual preference is a matter of choice. The problem statement for the course assignment is as follows:

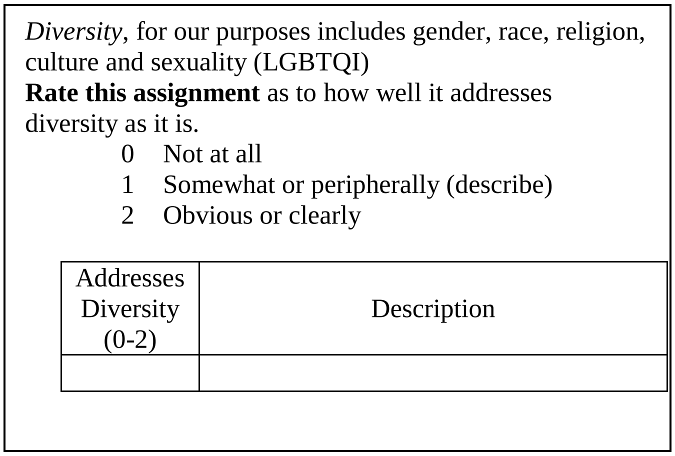
“Many still believe that sexual orientation is a matter of choice. It is well known among the ignorant, that proximity to those whose sexual preference varies from one’s own may cause one to question one’s sexuality or even to change. In Conway’s game of life, each cell has one of two states (dead or alive). In ours, each cell will have one of three states: gay, straight or questioning.”

Clearly, people that believe either way could be offended by the problem statement. The problem as originally conceived used people of three religions (Christian, Muslim and Hindu) with somewhat different rules for the state changes. Either formulation of the problem is certain to offend some group. The problem could be “sanitized” for individuals by referring to the states as A, B and C.

Similarly, someone may find that an assignment asking students to create a cash register program for specific items popular in Mexican cuisine as offensive. Similar items may, of course, be chosen from any culture. Once again, the problem is easily “sanitized” by referring to product1, product2, etc. In both cases, the authors view this sanitizing as failure to create an inclusive environment.

As part of the accreditation process, sample assignments, quizzes and exams were collected for 34 of the department courses. Student employees were tasked with completing an electronic form to evaluate the inclusiveness of each of the 135 assignments. The authors then reviewed the students’ work. The form consisted of identifying information, definitions, directions, and rubric-like scales. For example, Fig. 1 shows the portion of the form used to determine how well the actual assignment addressed diversity.

In addition, each assignment was rated as to how easily it could be modified to address diversity without changing the pedagogical goals of the assignment, and how easily diversity could be included with a possible modification of the educational goals, without changing them entirely. Next, the authors challenged themselves to generate at least two assignments that included diversity for each course that is used as a requirement or elective in all three majors.



Diversity, for our purposes includes gender, race, religion, culture and sexuality (LGBTQI)
Rate this assignment as to how well it addresses diversity as it is.

0 Not at all
1 Somewhat or peripherally (describe)
2 Obvious or clearly

Addresses Diversity (0-2)	Description

Fig. 1. Portion of data collection form

III. ANALYSIS OF ASSIGNMENTS

A total of 135 assignments were analyzed by the student employees and then reviewed by the authors. It is worth noting that some courses are offered more often than others. Some instructors either did not use or did not submit sample assignments (perhaps using a series of quizzes to evaluate students) and some focus on a small number of projects, while others submitted as many as ten small (lab) assignments. Naturally, some of the assignments are simply worksheets or textbook problems sets, while others are custom-developed projects. Table 2 provides a summary of the number of assignments and the results of the analysis, with some assignments appearing in multiple columns.

That only 24 of 135 (18%) existing assignments addressed diversity as the assignments were submitted, with only four (3%) doing so in obvious or direct manner, may be indicative of a number of things. Some, may even find 18% a surprisingly high number in the rather technical fields of CS, IS and IT. Of the assignments that could be easily changed, 20 are not included in the existing (As Assigned) assignments. We find that having 44 of 135 (33%) of the assignments easily changed to include diversity an encouraging finding. From another perspective, this implies that 33% of assignments could have addressed diversity, while only 18% (3% in an obvious or direct way) actually did.

IV. ASSIGNMENTS WITH DIVERSITY

Investigators challenged themselves to generate at least two assignments that included diversity for each course that is used as a requirement or elective in all three majors. With the exception of one course (CPSC 450: Internship), where only one assignment was devised, we were successful. Most of the assignments (see Appendix A) were uniquely generated,

TABLE 2. ANALYSIS OF EXISTING ASSIGNMENTS

	As Assigned			**Modify w/o affecting goals**	**Modification that might affect goals**
Diversity somewhat or peripherally addressed	20		**Might be changed to address diversity**	18	18
Diversity obviously addressed	4		**Easily changed to address diversity**	27	4

although some were based on existing assignments. That none of the assignments simply referred to an existing assignment may say something about how faculty generate assignments. Presumably, there is a tendency to want new, unique problems – a laudable goal when students often seek "insight" from Internet search engines as a first step to solving a problem.

One of the goals of this study was to create a set of sample assignments that could be used directly, or as the basis for modification, by other faculty at other institutions. As expected, some of the more theoretical classes where a greater challenge and those that included programming somewhat easier. Web programming, in particular, is especially conducive to including diversity, as the assignment often includes or even focuses on images or data which can be selected for diversity. There are four categories of assignments in the courses: problem statements that include examples, assignments that use specific objects, data processing problems, and theoretical problems. Each of these four categories are explained and a sample assignment the emphasizes diversity is provided.

A. *Problems statements that include examples*

Care should be taken to at least maintain neutrality in the use of examples, unless one is seeking to emphasize diversity. The use of he/she or simply using titles, rather than names can help in maintaining neutrality. Selecting Susan or Juan or Juanita as the name of the project manager and John as the programmer implicitly includes an underrepresented group.

A large number of problem statements include one or more examples to help elucidate the problem for the student. Even when the problem does not otherwise address diversity, the examples selected often can. For example, a problem asking the student to sort names (strings) might well be elucidated by the names of famous underrepresented people. Figure 2 provides an example from an IT course.

Web development is inherently more visual than many traditional computing problems and lends itself easily to the inclusion of images. Whether these images are part of the problem, an example, or simply decoration, they allow for an obvious inclusion of diversity.

CpSc 327: Systems Administration and Security

Project 2: Country of Origin

"The highest-piracy countries are Armenia, Bangladesh, Georgia, and Zimbabwe, all over 90 percent." [http://www.telecentre.org/profiles/blogs/the-global-software-piracy] Many attribute the relatively high rates of piracy to social conditions. When it comes to Internet attacks, the US often points the finger of blame at China or other global competitors. Research the issue of where Web-based attacks, phishing URL, and spam originate (or are hosted). Determine the top three countries for all malicious activity. Start with Symantec's Internet Threat Security Reports, but back this up with additional resources.

Fig. 2 Example from Information Technology

CpSc 464: Principles of Concurrent Programming and Operating Systems

Project 1: Threads

You are to write a C program that uses POSIX semaphores and threads to implement a multiple producer, multiple consumer conveyor belt that produces both sopes and molletes. Use the following information to complete the assignment: Two employees, Carlos and Fernanda, consume the Mexican snacks by removing them from the conveyor belt in the order they are produced. The conveyor belt can only have 10 items on the belt at a time with a maximum of 3 sopes permitted on the belt at a time. Additional restrictions: the Mexican snacks are consumed in FIFO order, producers should exit when a total of 100 Mexican snacks are generated, do not use global variables, pass parameters. For proper simulation, you will need to implement parameters that control timing.

-t N the delay in milliseconds between the production of each sope.

-s N the delay in milliseconds between the production of each mollete.

-F N where N is the number of milliseconds that Fernanda needs to place an item in its wrapper

-C N where N is the number of milliseconds that Carlos needs to place an item in its wrapper

Output each time a Mexican snack is produced and each time a Mexican snack is consumed. At the end of the run, print a summary of how many sopes and molletes were produced and how many Mexican snacks Carlos and Fernanda consumed.

Fig, 3 Example from Computer Science

B. *Problems that use specific items*

In many programs the problem is made more real to the student by identifying specific objects. These might be objects that are being purchased or sold, organized spatially or searched for. In any case, example objects may be selected that in some way represent diversity, items that are generally thought of as being representative of a culture (as shown in Fig. 3). Of course, care should be taken not to select items that represent a stereotype.

C. *Data processing problems*

All computing programs address data in some way, but many are designed to read, process and output data. The sample data set often offers an easy opportunity to include diverse names of people or items as seen in Figure 4.

CpSc 323: Database Systmes
Project 1: SQL Queries

A table of foods and their ethnic heritage has been created. It consists of the columns:

- FoodName
- NumIngredients
- Ethnicity
- PrepTime
- CookTime

Write the Queries to display the following:

1. FoodName and Ethnicity
2. CookTime
3. All FoodNames where CookTime is 30
4. All FoodNames where PrepTime + CookTime is < 60
5. All columns Sorted by Ethnicity

Fig. 4 Example from Information Systems

V. CONCLUSION

For 30 of the 34 courses at least two assignments that easily addressed diversity were created. Programming and discussion courses were easy to develop assignments for. However, some courses do not seem to lend themselves to more inclusionary assignments. In particular, Computer architecture, Computer Organization, Theory of Computation, and Compiler Design assignments did not fit the generic approaches described above. When the assignment is to build an adder, diversity may seem unattainable. However, an assignment to write an assembly language with instructions for non-Western cultures could be developed. Hence, great imagination can be used to coerce an assignment to be more inclusive.

For those courses that assignments cannot be created, it is generally worthwhile to discuss with the students a particular person from some underrepresented group as a means of motivating the assignment. Although this is separate from the actual assignment and does use at least a small amount of class time, it is an excellent way to make students aware of the contributions to the discipline from underrepresented people. Dr. Richard A. Tapia [5] or Maria Klawe[6]could be used for theoretically courses; Admiral Grace Murray Hopper [7] could be used for hardware or compiler courses.

REFERENCES

[1] M. Wick. “Using programming to help students understand the value of diversity”. Proceedsing of SIGCSE 2009

[2] J. Cohoon and L. Tychonievich. “Analysis of a CS1 Approach for Attracting Diverse and Inexperienced Students to Computing Majors”. Proceedings of SIGCSE 2011. Pgs 165-170

[3] National Center for Women and Information Technology. http://ncwit.org

[4] V. Gkatzidou, E. Pearson, S. Green, and FO. Perrin. “Widgets to support disabled learners: a challenge to participatory inclusive design” OzCHI '11: Proceedings of the 23rd Australian Computer-Human Interaction Conference. November 2011.

[5] Tapia, Richard. http://www.caam.rice.edu/~rat/

[6] M. Klawe, http://www.hmc.edu/about1/administrativeoffices/officeofthepresident1/bio.html

[7] Hopper, Grace. http://www.history.navy.mil/bios/hopper_grace.htm

APPENDIX A: SAMPLE ASSIGNMENTS

Each department course used in one of the majors is listed here along with a sample assignment for that course. Courses are identified by title and include an indication of which major (CS, IS or IT) uses the course and whether the course is required (R) or elective (E). A full listing of the curriculum for each major is available at http://cs.sru.edu. The complete list of example assignments is available at http://cs.sru.edu/~mullins/diversity/AppendixA.pdf.

CpSc 130: Introduction to Programming and Information Systems (IT-R, IS-R)
Project 2: Best Movie Page

Create a web page that lists various categories of films, including at least action, foreign, romance, comedy, and gay/lesbian. Using the images provided as icons (select others as needed), allow the user to choose a category. The link takes the user to a separate category page for each genre you have chosen. In each category, add at least two well-known films. If you don’t personally know of a film that should be listed (your opinion), do a key word search at imdb.com. Each Movie listing must include a title and a short description on a well-formatted page.

CpSc 140: Introduction to Programming Principles (IT-R, IS-R, CS-R)
Project 1: Flash-Card Learning Game

Write a program that maintains pairs of words (or phrases) in an array or pair or arrays, one in English, the other in Spanish. The program allows the user to select which language is presented. During play, the program randomly selects a term from the appropriate array, displays it, and reads the user’s response. The response is compared to the expected result, scored, and, if not correct, the correct answer is displayed. The user must be able to indicate when game play ends, at which time a final score is presented.

CpSc 150: Advanced Programming Principles (IT-E, IS-R, CS-R)
Project 1: Object Oriented Design

Create a program that simulates the operation of a vending machine. The user interacts with an interface that offers choices of food, accepts coins, and responds appropriately.

The program could offer a choice of language for the interface such as English or Spanish. Another option would be to offer a choice of ethnic foods such as Mexican or Italian or Classic American Fast Food.

CpSc 207: System Software and Architecture for End Users (IT-R)
Project 1: working with Basic UNIX commands

Create a file containing 10 English words each followed by a space character and then a one-word translation (using Alta Vista's Babel fish, Google's translation tools, or an actual bilingual dictionary) of that word into some other human language (extra credit for using non-Indo European languages). Write a shell script which accepts, as input, a word from either language and outputs its translation.

Discuss why this technique might not work for the general problem of translating text in one language to text in another language.

CpSc 210: Productivity Software (IT-E, IS-R)
Project 2

Prepare a PowerPoint presentation about a multinational software/hardware company based in Asia or Africa.

CpSc 217: Structured and Dynamic Web Programming (IT-R, CS-E)
Project 2: String handling and probability

First choose two languages to work with. One should be Indo-European, the other should be non-Indo European. The orthography of at least one should not be the Roman alphabet. Build a web page in which the user can type or paste a paragraph or more of text from either language. When done, the user can press a button which "analyzes" the text. The analysis consists of two tabulations: the frequency of occurrence of every glyph (character) including punctuation marks (In alphabets that distinguish between lowercase and uppercase letters, the two variants of the same letter should be considered the same so that "a" and "A" are identified) and the frequency of occurrence of every character pair (or digraph -- in which "ab" is not the same as "ba") is tabulated and presented on the page.

Finally, when a second button is pressed, new random text is generated and placed somewhere else on the page. The newly generated text should be generated with the probabilities of characters' occurrences matching the frequency of occurrence within the user's text. That is, if the occurrence of "e" accounts for 12% of the characters used, then the probability of generating an "e" should be 0.12

CpSc 236: Selected Computer Languages (IT-E, IS-E, CS-E)

All of the suggestions from CpSc 150 are appropriate for courses in other languages

CpSc 300: Challenges of Computer Technology (IT-R, IS-R)
Project 3

Why are there so few women and minorities in information technology? Use current literature to answer this question in an essay exam.

CpSc 301: Practical Computer Security (IT-E)
Project 2: What is a threat?

Different nations have different laws about freedom of speech, decency and obscenity, and privacy. Suppose you were sent by your employer to administer computing and network systems in a foreign country. Give a case study of how the laws of one particular country might affect your job.

CpSc 305: Introduction to Expert Systems (IS-E)
Project 3

Match up exchange students with host families based on cultural interests, language, ability/willingness to provide transportation, etc.

CpSc 311: Discrete Computational Structures (IT-R, IS-E, CS-E)
Project 1: History of Mathematics

List three major contributions to the history of mathematics that were contributed by persons neither from Europe nor the Americas.

CpSc 317: Scripting Languages (IT-R, IS-E)
Project 2: Mapping with Geographic Data

The map at http://granite.sru.edu/~ddailey/usmap.svg takes public data from the US Census Bureau and shades the 50 states based on any of 82 variables chosen by the user. Do the same thing for the states, provinces, or counties of another country where English is not the official language. Be sure to discuss issues involved in finding the data, its original sources and its accuracy.

CpSc 323: Database Systems (IT-R, IS-R, CS-E)

See Fig. 4.

CpSc 327: Systems Administration and Security (IT-R, IS-E)

See Fig. 2.

CpSc 343: File Processing (IT-E, IS-R)
Project 2

Design an HR database (set of files) that can be used by a company for employee management and healthcare management where the company allows domestic partners (and dependents). The system shouldn't "out" people just because they are using the benefits. Actually, this would be at least two "views" of the data.

CpSc 358: Simulation (CS-E)
Project 2: Best Fit and Bias.

Create best fit for data on table service for various ethnic and gender customers. The object would be to determine if service was fair among the various groups.

CpSc 365: Management Information Systems (IT-E, IS-R)
Project 2

Determine a way to recycle equipment to "needy people". How are they identified? How is the program sustainable? How does it help the corporation?

CpSc 370: Computer Organization (IS-E, CS-R) Project 2

Write an assembly language with instructions appropriate for non-Western cultures.

CpSc 374: Algorithms and Data Structures (CS-R) Project 1: Shortest-Path

For all of the major buildings on campus, develop a map (graph) that describes the path that wheelchair-bound person is able to take in traversing the campus. Weight each path segment based on a combination of difficulty of traversal and distance. Write a program that asks the user which building he or she is leaving and which he or she is traveling to. Determine the shortest path for the traversal.

CpSc 376: Programming Languages (CS-R) Project 2

Select a language from the list below and prepare a presentation for the class that discusses the salient design decisions.

Designer	Language
Alain Colmerauer	Prolog
Ralph Griswold	Icon
Guido van Rossum	Python
Ole-Johan Dahl, Kristen Nygaard	Simula-67
'83:Jean Ichbiah, '95: Tucker Taft	Ada
Yukihiro Matsumoto	Ruby

CpSc 378: Theory of Computation (CS-R) Project 3: The Game of “Choice”

Many still believe that sexual orientation is a matter of choice. It is well known among the ignorant, that proximity to those whose sexual preference varies from one’s own may cause one to question one’s sexuality or even to change. In Conway’s game of life, each cell has one of two states (dead or alive). In ours, each cell will have one of three states: gay, straight or questioning.

The program will read the state (rectangular map) from a file or allow the user to select cells interactively and assign a state. Each generation, including the initial is displayed. The rules for determining the next generation are as follows:

- If the cell is “straight” and 4 or more neighbors are straight, the cell stays the same.
- If the cell is “straight” and 2 or 3 neighbors are straight, the cell becomes “questioning”.
- If the cell is “straight” and 0 or 1 neighbors are straight, the cell becomes “gay”.
- If the cell is “gay” and 4 or more neighbors are gay, the cell stays the same.
- If the cell is “gay” and 2 or 3 neighbors are gay, the cell becomes “questioning”.
- If the cell is “gay” and 0 or 1 neighbors are gay, the cell becomes “straight”.
- If the cell is “questioning” and 4 or more neighbors are questioning or the number of neighboring gay cells equals the number of neighboring straight cells, the cell stays the same. Otherwise, the cell changes to the state of the maximum of gay and straight neighbors.

Submit with a DFA to show state changes.

Note: As implied above, this is not intended to model the real world. The “questioning” state is simply a contrivance for the program. We cannot replace “questioning” with bisexual, as we are not dealing with the gender of the neighbors. (And, a bisexual might well argue that he or she is not questioning his or her orientation.)

CpSc 413: Systems Analysis (IT-R, IS-R) Project 2

Temp Office Personnel specializes in providing jobs for women with children. They match special skills to needs at a company and take into account the time needs of the mothers. A position is often filled by two, and potentially, more temps. All the skills have to match. Times need to correlate. And, they need a way to effectively transition work. Create a data model for such a system.

CpSc 423: Computer Networks (IT-R, IS-R, CS-R) Project 3: Digital Divide

For each of the populated continents (Asia, Africa, North America, South America, Europe and Australia), identify at least three servers that will respond to ping and tracert in three different countries (assume Australia includes Tasmania, New Guinea, the Aru Islands and Raja Ampat Islands). For each of your servers, run ping tests to determine the minimum, average and maximum response times. Show the results using a graph. Use tracert to determine paths taken to each destination. Identify shared links in the paths and significant bottlenecks. Draw a graph showing the results.

CpSc 427: Interface Design (IT-R) Project 2: Color perception and color terms

Write a paper on the neurological, perceptual and cultural issues involved in the use of colors on a web site. Include such topics as why women have more terms for colors than men and cross cultural differences in the emotive connotations of various colors.

CpSc 443: Software Project Management (IT-E, IS-R)

This assignment should be the next step of either of the CpSc 413 projects. Presentation on projects described in some selected set of papers (that would describe projects in Asia, Africa, etc.)

CpSc 450: Internship (IT-E, IS-E, CS-E) Project 1: SRU Proprietary Report

After completing the internship, write a report that describes the company environment or culture as it pertains to multiculturalism. Describe how the culture could be improved while maintaining effectiveness and profitability

CpSc 456: Introduction to Computer Graphics (CS-E) Project 1: Avatar Creator

Many social networking sites allow the user to post an image which might be an avatar to represent them pictorially, some make more extensive use of avatars. For our purposes, we will assume a 2D, static avatar is the goal. Write a program that assists that user in creation of an avatar. Your program must take into account the preferred gender and race (skin tone) of the user. For extra credit, include height and weight preferences (for example, tall & slim).

CpSc 464: Principles of Concurrent Programming and Operating Systems (CS-R)

See Fig. 3.

CpSc 466: Compiler Design and Implementation (CS-E) Project 1

Use the biography of Vugranam Sreedhar as an introduction to a Compiler Design assignment.

In program analysis, compiler optimization, programming languages, security, business process, multicore Dr. Vugranam C. (VC) Sreedhar is currently a Research Scientist and a Project leader at IBM TJ Watson Research Center working in the area of Information Security. His Ph.D. thesis is in the area of program analysis and is entitled "Program Analysis Using DJ Graphs”. Dr. Sreedhar has worked in several projects architecture, concurrency analysis, software quality, and embedded systems.

Jugranam C. (VC) Sreedhar is a lecturer with the ACM Distinguished Speaker Program on various topics including "Static Single Assignment Form and its Applications ".

ABSTRACT:

Static Single Assignment (SSA) Form is now a well accepted intermediate representation for compiler optimization. In this talk I will discuss the core ideas behind SSA form, related representations, and how they is used in improving program analysis techniques and in improving compiler optimizations. I will also discuss some of the key algorithms for constructing SSA form. I will then discuss some new applications of SSA in the area of typestate analysis and detecting security vulnerabilities.

CpSc 474: Computer Architecture (CS-R) Project 1: Architecture History

Find and report on accomplishments of computer scientists of specific gender and races One example of a woman is Admiral Grace Hopper, a woman who did research on early computers such as the Mark I and languages such as COBOL. Another example is an African, Philip Emeagwali, a Nigerian who made major contributions to the design of the Connection Machine. (www.math.buffalo.edu/mad/computer-science/emeagwali_philip.html)

CpSc 476: Artificial Intelligence (CS-E) Project 2

Create a program that helps the visually impaired learn about a specific topic such as the importance of color in Monet’s paintings.

CpSc 478: Analysis of Algorithms (CS-E) Project 1

Use Djikstra's shortest path algorithm to perform a search of G=(V, E) where the vertices in the graph are cities in Africa and the edges of the graph are distances between African cities.

CpSc 488: Software Engineering (CS-R) Project 2: Project Design and Development

For the major course project, identify a female colleague or professional willing to act as customer-manager for each group. Students then develop the project iteratively with all “customer questions” directed to the customer-manager.

Microsoft Kinect Interface for Children’s Education

Bryant Hicks, Jonathan Kissinger, and Roger Lee

Central Michigan University
Department of Computer Science
Mount Pleasant, MI 48858, USA

Abstract - The Microsoft Kinect is a motion sensing interface device developed by Microsoft that has nearly limitless potential real-world applications. It supports gesture, voice, and facial recognition by use of a range camera. We are interested in researching an application of the Kinect that will enhance child development games and create a more intuitive and fluid learning environment for users. We have collected data on similar applications of the Kinect (e.g. existing gesture recognition systems, didactics, and potential future applications) in order to learn from existing similar systems. We then propose a solution that is tailored to our problem of enhancing the usability of child development games. Our solution incorporates existing gesture recognition systems and modifies them to make them more accessible for children.

I – Introduction

The education of our future generations is an important matter, and current technology should be incorporated into modern education. As a society where new technical innovations and applications are constantly being introduced, it is imperative that children gain exposure to technology and that it is incorporated into their education from an early age. We propose using the Microsoft Kinect for this very purpose. Educational games are a valid way to get children interested in learning in an interactive manner. Therefore we believe that the creation of educational interactive games for the Kinect is a great medium for learning that can be used in a multitude of environments from a family home, to a daycare, to a classroom. In this paper we propose several ideas for Kinect games that will help young children to learn the foundational knowledge required for future education.

II – Background

A – Children’s Interactive Games in the Past

Interactive games of many shapes and forms have been an important tool in the education process for years. Even before the digital age games were used for educational purposes. There are a multitude of examples of these types of interactive games; from flash cards, to the well-known I-Spy series of books, to simple card matching games. These time-tested interactive learning devices show that it is easiest to engage a child in learning when the child sees the activity as entertainment. Entertainment is one form of intrinsic motivation which has been shown to increase learning[5]. In a study done with computer games, Habgood and Ainsworth showed that when intrinsically motivated, children significantly outperformed their peers when learning division. Such devices provide a more visual means of education which can benefit all children. Interactive games such as these are still widely used in children’s education today, and will continue to remain a staple of developmental learning.

B – Modern Children’s Interactive Games

As computers became more prevalent in our everyday world they also became more prevalent in an educational setting as well. Many schools adopted the use of a computer class in order to enhance basic computer literacy and integrate educational games into the curriculum. Some popular examples of such games include Oregon Trail, Gizmos and Gadgets!, and Logical Journey of the Zoombinis. These early interactive computer games taught children everything from critical thinking, basic physics and math, history, and logic. Games such as these were an important tool that helped children not only grasp foundational knowledge, but also to interact with computers and computer programs. More recently entire platforms have been created for educational gaming. The most notable example of this is the Leapfrog system. Leapfrog offers a multitude of personal gaming platforms whose games focus on learning in an interactive manner.

C – Future Children's Interactive Games

The evolution of interactive education has shown that in order to engage children in learning we must keep up with the most advanced mediums of interactivity. The Microsoft Kinect is a cheap system that allows for simple and natural interfacing with educational gaming software. Interacting with the Kinect involves basic gestures that are easily performed by young children. "The current focus on the use of new technologies and media for teaching and learning purposes has led to an intensifying interest in the properties and peculiarities of educational videogames. These are now considered a powerful tool that can revolutionize the current teaching-learning methodologies. Many scholars agree that they are the future of education." [1]

III – Goals of our Proposed Games

We hope to enhance children's education by developing games that will captivate the interest of young children while including an educational aspect. By using a Natural User Interface (the Microsoft Kinect) we believe we can create applications that young children (3-6) can easily navigate and play with little adult guidance.

IV – Methods

A – Choosing a Platform for Development

We started by collecting a number of research articles that in some way relate to our topic of enhancing learning for children using the Microsoft Kinect. Our articles ranged from topics about current implementations of learning systems used for disabled people, to general interfacing with the Kinect, to child development. We drew the pertinent knowledge from all our collected sources and decided upon the best model for our project. We decided upon using the Kinect because of the wide variety of applications that can be developed for it. "With its wide availability and low cost, many researchers and practitioners in computer science, electronic engineering, and robotics are leveraging the sensing technology to develop creative new ways to interact with machines and to perform other tasks, from helping children with autism to assisting doctors in operating rooms." [2]
Another reason we chose the Kinect as our intended platform is because of the ability to download a free software development kit and use it to easily create very customizable software applications. The article "Brave NUI World" discusses this very issue: "Since the Microsoft SDK precludes commercial use, many early academic and enterprise projects using PrimeSense and/or stripped down Kinect hardware use either homebrewed algorithms or open source drivers and middleware released by consortia such as OpenCV or OpenNI, the natural interface forge formed in November 2010, by PrimeSense and robotics pioneer Willow Garage." [3]

B – Proposed Interactions with the Kinect

We decided to use simple gestures, in order to implement the basic functions of the applications. These gestures include: pointing at a desired image or button to select it, and moving the hand from right to left and vice versa. Many of the interactions we decided on are commonly used in normal interactions with the Kinect, and are actually also similar to using a touch screen device. Since the Kinect supports multiple user recognition, our applications would be able to be extended to include multiple players competing against each other. The article "Kinect Identity: Technology and Experience" describes the multi-user profile support, "To maximize the chances of creating a successful identity tracking system, Kinect's developers experimented with a set of independent identification technologies, selecting a set that, when combined, created a complete picture." [4] By enabling multiple users playing the games, we add a potential competitive aspect which also allows for greater social development amongst the children.

C – Proposed Applications

We decided that it is easier to get children to learn when they see it as a "fun" activity. Therefore we propose that we should use interactive learning games in order to engage the children in learning. We targeted our games for children ages 3 to 6 who are still learning simple things such as the alphabet, counting, and basic spelling. Our first game idea is an alphabet ordering game targeted at young children learning the alphabet. A simple proposed layout is shown in Figure 1.

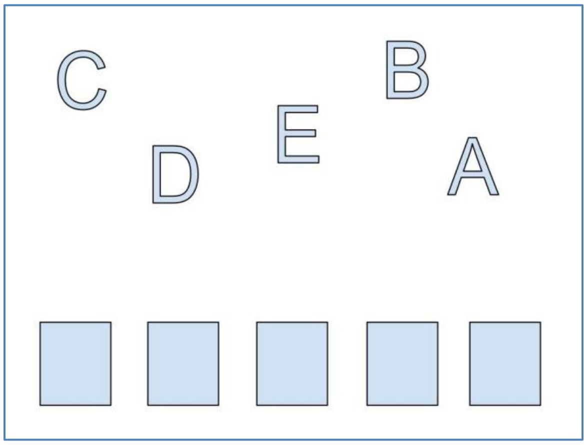


Figure 1 – A simple proposed layout for the letter ordering game.

The game would feature a screen where a set of letters are displayed out of order. The user would then point to a letter and drag it to its proper location in the sequence of letters. This game would help teach children the alphabet in an easy way by splitting it up into subsections. It is easier to learn the order of letters when they are in smaller groups and the game could be expanded to include the whole alphabet once the child has progressed to an adequate level. Our second game idea is another matching game where the user matches a word to a picture. A simple proposed layout is shown in Figure 2.

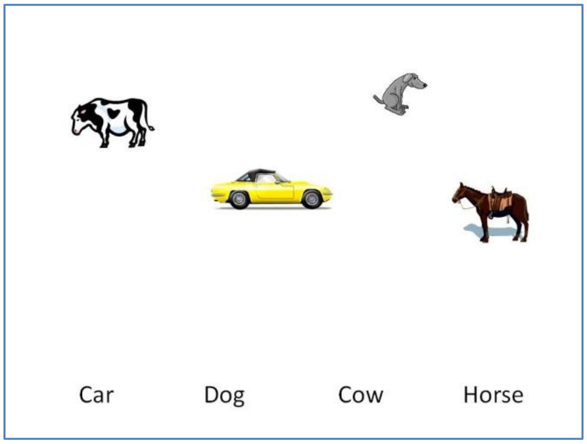


Figure 2 – A simple proposed layout for the picture matching game.

The screen would display a set of words randomly strewn across the top of the screen, and a row of pictures on the bottom of the screen. The user would then point to a word and drag it to the matching image at the bottom of the screen. This game would help children to connect language to a visual representation, and also help with spelling. This game could include many different sets of pictures/words that could be tailored to children just learning to spell (which would include only short well known words) or older children who have already learned the basics (which would include longer words). Our third game idea is a number ordering game. A simple proposed layout is shown in Figure 3.

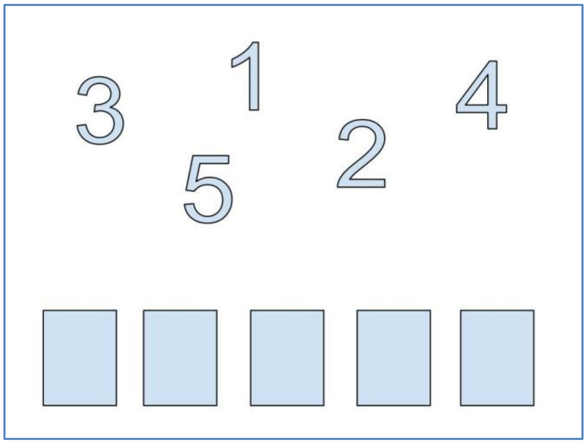


Figure 3 – A simple proposed layout for the number ordering game.

The main screen would display a series of numbers in a scrambled order. The user would then point to a number and drag it to its proper location in the number sequence. This game would help children with basic counting and understanding orders of magnitude. This game could include sets of numbers that are single digit, multiple digit, and even non-sequential numbers. Single digit sets of numbers would help the youngest of learners with basic 1 through 10 counting skills. Multiple digit and non-sequential number sets would be targeted towards older children who are learning orders of magnitude and advanced counting skills. Our ideas cover basic language and math for children users, and are designed for simplicity.

V – Conclusion

Throughout the years education and technology have grown hand in hand. As new technologies are introduced, educational applications are created alongside them. The Microsoft Kinect presents an opportunity to utilize cutting edge technology for educational purposes at a reasonable price. Since Microsoft released an SDK for developing software for the Kinect the potential educational uses of the system are almost limitless. We propose several ideas for games that would focus on interactive learning for young children.

VI - Future Research

In the future we would like to develop our proposed applications and conduct testing regarding their efficacy. We would like to first create skeleton versions of our ideas in order to test them ourselves. We would then fully flesh out our applications and move on to testing with our target age group (young children 3-6). If these tests proved fruitful we would then publish our applications for use in educational settings.

References

[1] Di Tore, Stefano, Francesca D'Elia, Paola Aiello, Nadia Carlomango, and Maurizio Sibilio. "Didactics, Movement and Technology: New Frontiers of the Human-machine Interaction." *6th INSHS International Christmas Sport Scientific Conference* 7 (2011): 178-82. Web.

[2] Zeng, Wenjun. "Microsoft Kinect Sensor and Its Effect." *Multimedia at Work* (2012): 4-10. Web.

[3] Goth, Gregory. *Brave NUI World*. Publication. ACM, Dec. 2011. Web.

[4] Leyvand, Tommer, Casey Meekhof, Yi-Chen Wei, Jian Sun, and Baining Guo. "Kinect Identity: Technology and Experience." *Entertainment Computing* (2011): 94-96. Print.

[5] M. P. Jacob Habgood, Shaaron E. Ainsworth. "Motivating Children to Learn Effectively: Exploring the Value of Intrinsic Integration in Educational Games." *Journal of the Learning Sciences* (2011): 169-206. Web.

Engaging Preadolescent Females in Computer Science Through Robotic Art

Brian McLaughlan
Brian.McLaughlan@uafs.edu
Department of Computer and Information Sciences
University of Arkansas – Fort Smith
5210 Grand Ave
Fort Smith, AR 72921

Abstract – **Robotic competitions have become a popular and effective method for teaching science, technology, engineering, and math (STEM) topics. However, these contests remain decidedly male dominated despite efforts to engage females. Robotic art exhibitions were developed as a method for attracting females and other underrepresented groups to robotics. This paper quantifies the effectiveness of these exhibitions with preadolescent females, an age where most children are deciding whether or not they enjoy science. Also examined is the appropriateness of various technology products in conducting these exhibitions.**

Keywords: Robotics, education, learning styles, gender

I. Introduction

Robotic competitions such as FIRST Robotics and BEST robotics have demonstrated the viability of teaching science, technology, engineering, and math (STEM) concepts in ways that are attractive to students in both K-12 and post-secondary schools. For years, MIT's robot competitions have filled stadiums with teams and fans. "Battlebots" and "Robot Wars" competitions became popular enough to be turned into television shows that aired for several seasons.

While robotics is an excellent way to integrate many facets of STEM into a single learning project, many students are not attracted to them. For instance, the percentage of female participation in MIT's robot competitions is significantly lower than the base demographic numbers would suggest. The results are similar in younger students, with FIRST Robotics LEGO league having only 30% girls competing (Melchior, et al, 2004). Other studies have suggested that computer science education should be integrated with art, music, and literature, in addition to traditional engineering contexts (AAUW, 2000).

In the mid-nineties, Wellesley College noticed this trend and designed an open-ended intersession course on robotic design. The course had the goal of introducing the "big ideas of engineering" (Turbak and Berg, 2002) by exposing students to broad engineering concepts such as:

- Understanding the design and implementation cycle – implementation, testing, debugging, and refining designs.
- Systems design involving complex interaction of heterogeneous components.
- Designing for the real world.

The goal of this paper is twofold. First, it seeks to quantify the effectiveness of robotic art projects in attracting girls to and retaining them in STEM topics, particularly computer science. Second, it examines the appropriateness of various technologies currently on the market for this type of learning. This paper focuses on girls aged eight to eleven.

II. Approach

The study approached the teaching of robotics concepts in a structured manner. The students met each week in two hour blocks for four weeks. Each group was limited to ten students, with three groups of students being examined. At the beginning of the first meeting of each group, students were given a survey to determine their initial attitudes towards STEM topics and learning in general. Initial questions addressed the following categories:

- Previous experience in computer science (programming, building circuits, et cetera)
- School likes and dislikes (math, science, art, social science, reading, school in general)
- Project attitudes (excitement/apprehension, attitudes toward crafting, working with electricity and circuitry, programming computers)
- Potential career attitudes (STEM careers versus traditional “women’s roles” careers)

Second, students were introduced to the concept that computers had to be instructed quite literally in order to get them to behave as desired. To accomplish this, the following ice-breaking activity was employed (Genesereth and Nilsson, 1987):

Students were divided into groups of four to five. Each group was given an identical, simple task such as moving pompoms from one bowl to another. Each student in a group was given a particular function. One student would function as the *Brain*. This student would make the decisions for the robot. However, the Brain cannot see the scene. In addition, the Brain is the only one who knows what the goal is. Another student functions as the *Eyes*, telling the Brain what it sees. Two students would function as *Hands* (either left or right hand). These students were blindfolded and given a single oven mitt to cover their “manipulator” hand. A Hand can only follow simple instructions and can only make a beep noise if it encountered a problem. If a fifth or sixth student was present, they also functioned as the Brain. The Brain was then given the task, and the group attempted to accomplish it. Students were allowed to rotate positions between tasks so that they could experience the robot from multiple points of view.

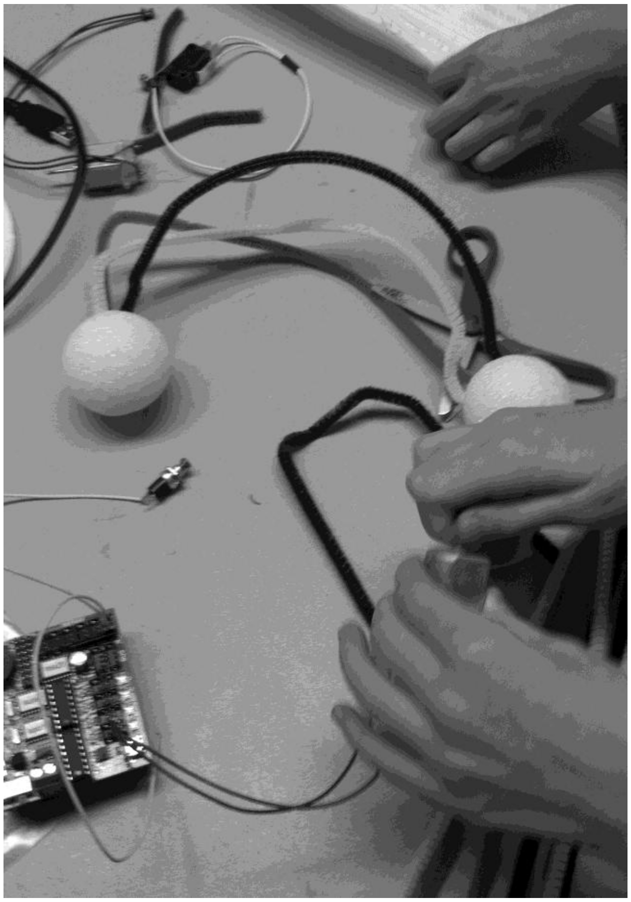
Next, students were presented with the microprocessor that would form the robot brain of their craft projects. Students were familiarized with the various ports and components, and any questions were answered. In addition, the students were introduced to some basic robotics terminology such as “actuator” and “sensor” and were shown what actuators and sensors were available for their projects. Available sensors and actuators were:

- Small electric motors
- LED lights
- Pressure/contact switches
- Lever switches
- Light sensors
- Sound sensors
- Speaker (built into the processor board)

Finally, the students were let loose on their projects. Students were allowed to form groups of two or three, with most groups staying at two members. Helpers were available to assist students with wiring and programming as well as nudging with hints for solving problems along the way or for getting students back on track. However, the majority of the work was paced and accomplished by the students. At each weekly session, the investigators asked questions to determine the students’ current attitude toward the project and STEM.

In conjunction with this study, the hardware used was evaluated. Three different, popular microprocessor systems were utilized: the Gleason Research SuperCricket, the Lego Mindstorm NSX and the Arduino

Uno. If the particular microprocessor was packaged with sensors and actuators, those devices were evaluated as well, giving the processor a final score based on the composite evaluation of all its parts.



As a follow up after the project ended, students were asked to complete another survey to assess any changes in attitudes. The survey questions addressed the following categories:

- School likes and dislikes (as described above in pre-project survey)
- Project attitudes (as described above in pre-project survey)
- Future plans ("Would you like to do this again?", "Would you like to program more computers?", "Would you like to do more science experiments?")
- Future career attitudes (as described above in pre-project survey)
- Attitudes toward specific equipment (likes, dislikes, ease of use)

In developing the curriculum for this project, the advice from previous research was incorporated (Rusk, et al, 2008). In particular, this project employed four strategies:

1. Focus on themes, not just challenges
2. Combine art and engineering
3. Encourage storytelling
4. Organize exhibitions rather than competitions

Strategy 1: Focus on Themes
Most robotics projects are oriented towards accomplishing goals or overcoming challenges. For example, the goal might be to develop a robot that can play basketball or that can sort objects. Instead of a single design challenge, this work presents a broad theme that allows the students to focus their efforts on what they are interested in.

The theme for this study was "Over the Rainbow." The aim was to provide a theme that would spark the participants' imaginations. Student projects ranged from spinning rainbow hats to singing spiders, providing a nice diversity of exhibits.

Strategy 2: Combine art and engineering
Students were provided an array of building materials including Styrofoam cups and bowls, pompoms, colorful pipe cleaners, wiggly eyes, and construction paper. They were allowed to work on any aspect of the project – building, wiring, and programming -- as they saw fit.

Strategy 3: Encourage storytelling
Most traditional robotics projects focus on the mechanical properties of the robot. While this is appropriate, it is not the most appealing method for all students. Some students prefer a more dramatic approach, using technology to illustrate a story. To facilitate this style of play, this study prompted students to brainstorm their ideas not only in terms of what the robot does, but also the context in which it is doing it. After being encouraged in this manner, many students began looking at their projects not simply as a robot in the traditional sense, but as a stage where they were the directors.

Strategy 4: Organize exhibitions
Competitive environments are not attractive to all students. This study organized an art exhibition in which families and friends of the students could interact with the students and their finished projects. The move to an exhibition allowed students more freedom in choosing what to develop rather than building towards the goal of winning.

III. RESULTS

The study produced a number of very interesting robotic projects. A small number could be classified as traditional "robot that does something" projects, but many creative students went well beyond this. Among the projects were:

- A multi-colored caterpillar that comes out of a hole when a light shines on it. It retreats back into its hole when the light is removed.
- A hat with a spinning rainbow on top. The rainbow spins when the hat is placed on a head.

- A spider that wiggles and plays a tune when a button is pushed.
- A ballerina that dances on a rainbow when a button is pushed.
- A robot that wiggles when it hears clapping (actually any loud noise will trigger it).

IV. Observations and Conclusions

By examination of the survey data as well as compiling the notes taken by the investigator and helpers, a number of patterns emerged.

The icebreaker activity proved to be very enlightening to the students. Students who missed the first session of the study were at a significant disadvantage. These students had more difficulty recognizing the literalness of the instructions that must be given to the microprocessors to make the robots work than students who participated in the initial activity. This activity will be used in future implementations of this study for all age groups.

Regarding the four strategies, several observations can be made. First, the "Over the Rainbow" theme seems to have been taken more literally than intended, with most projects incorporating rainbows directly into their robots. Students were reminded that the robots could involve anything that you might associate with rainbows and fantasy, such as unicorns or the Wizard of Oz setting. However, this age group tended to not think too far outside the box. Fortunately, the projects were still very creative, but more work will need to be made on future themes to ensure they do not restrict creativity but instead enhance it.

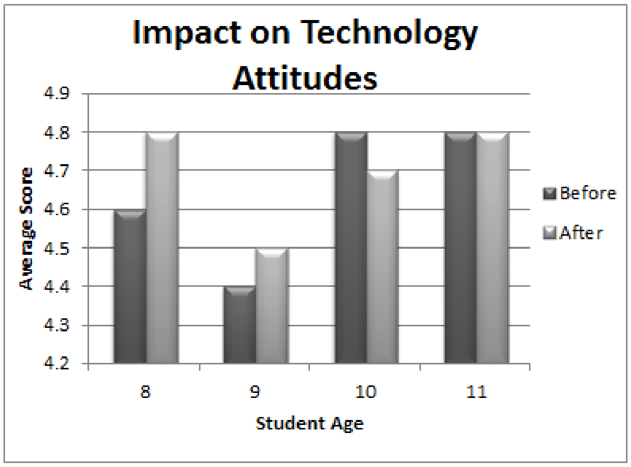


Figure 1: Impact on technology attitudes

While the students were generally quite familiar with craft projects and the materials used in these projects, they were faced with many engineering problems as they attempted to incorporate circuitry and motors into their works of art. Motors often generated too much torque, tearing themselves from their weak tape or glue adhesion, and requiring students to come up with better mounting solutions. Other students encountered load-bearing problems or had difficulty figuring out how to attach sensors and actuators to their artwork without degrading the aesthetics of the project. The solutions to these issues required the application of problem solving skills, experimentation, and thinking outside the box.

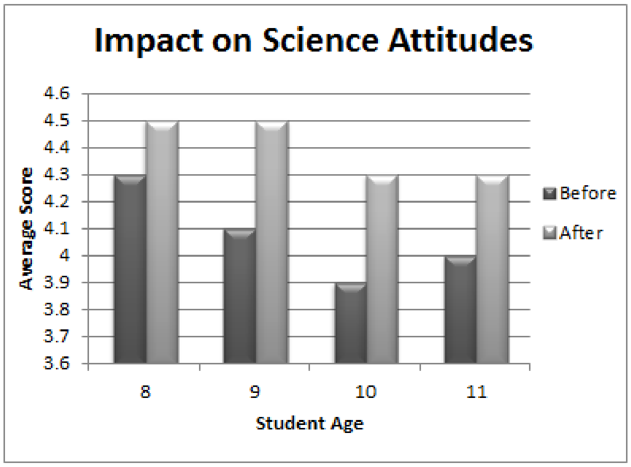


Figure 2: Impact on science attitudes

Several students initially began by building what they envisioned as being a traditional science fiction robot, that is, something from a Star Wars movie. As the projects progressed, however, most began making up backgrounds for their robots and began gravitating toward a "story" style project.

The exhibition style finale to the study proved to be a success. A few students asked if there would be awards for the best project, demonstrating that the desire for competition exists in all demographics. However, the students were intentionally not told about the exhibition until the second session to see if this altered their attitudes. It appeared to have a significant effect on the students as it became a major drive for completion of the projects.

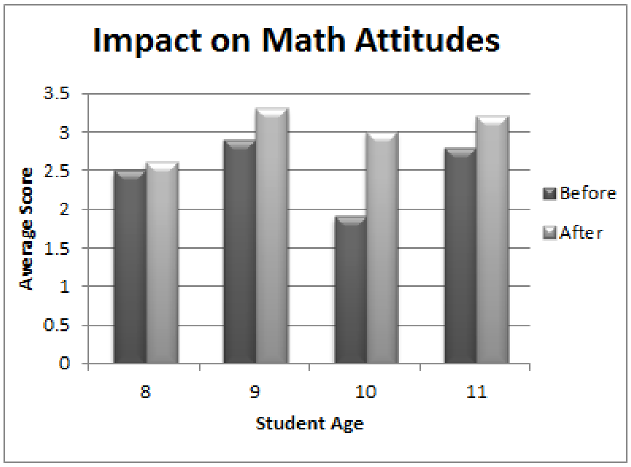


Figure 3: Impact on math attitudes

Some quick observations to be made regarding the survey data:

- All the students had positive initial attitudes toward technology (Figure 1). This was not the case for science (Figure 2), and certainly not the case for math (Figure 3).
- While all the students were excited about the project, the excitement seemed to peak with the 10-year-olds. The younger students exhibited more initial apprehension and timidness, while a few of the older students initially seemed slightly aloof.
- Fortunately, the students enjoyed the study. In fact, all the students reported that they would like to do a project like this in the future.
- In addition to initial apprehension, eight-year-olds tended to show some frustration at times.
- Most students enjoyed all aspects of the project, including programming the processor. This enjoyment varied slightly depending on which processor the student was using.
- Since most of the students had rather high initial attitudes toward robotics and technology, these attitudes did not increase by much due to the study (Figure 1).
- Since adults overseeing the crafting process were constantly emphasizing the math and science components that were required to build the robots, students had an observable increase in favorable attitudes toward math (Figure 3) and science (Figure 2) once they made the connection between these topics and building robots.
- As a control, the students' attitudes toward reading did not significantly change throughout the study (Figure 4).
- Younger students found the survey to be too long.

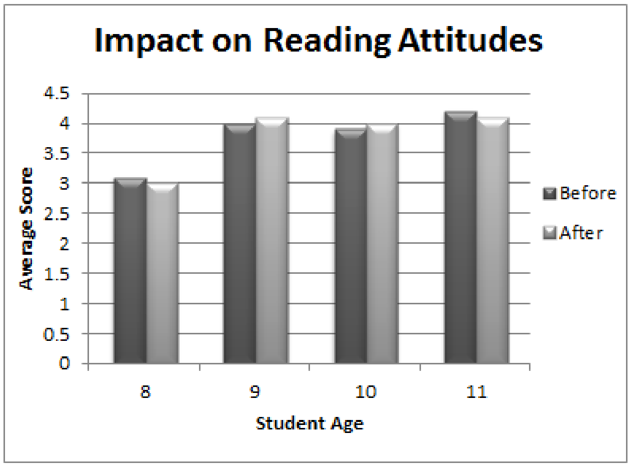


Figure 4: Impact on reading attitudes

Some observations and conclusions regarding the various hardware:

The Arduino was the cheapest processor, but it required the most setup, such as soldering components. It also earned the lowest score from students due to the more traditional programming interface (i.e., it is programmed by typing syntax).

The Supercricket had significant technical problems due to its infrared connection for downloading programs to the processor memory. The program compiler was often unable to detect the microprocessor, leading to discouragement from the students. Compounding the problem is a lack of support by the manufacturer, with most of the support for the device coming from independent forums. If those technical problems could be averted, the Supercricket would represent an excellent value; it had a moderate cost but was completely assembled. As it stands, the problems caused too many delays and discouragement to be worth buying.

The Mindstorm was the most polished product as well as the most expensive by a significant factor. Students rated this processor highest. Some of that attitude could be from the familiarity with LEGO as a toy rather than a device with exposed circuitry. However, that familiarity came at an additional cost (besides dollar value). That toy-like exterior reduced the students' recognition that they were actually programming a robot brain instead of playing with a toy. Additionally, students using the LEGO system tended to want to stick with the LEGO materials that came with the kit rather than incorporate non-LEGO materials. These problems are actually rather minor; the major problem with the Mindstorm is its enormous price.

All three processors have significant good and bad points. However, if a drag-and-drop programming interface could be located or written, the Arduino would be the best overall choice for this age group.

V. Future work

This paper is part of a larger study to determine the feasibility of implementing this teaching method as a companion to BEST and FIRST robotics competitions already being held in the area. The lessons learned here are currently being implemented in a pilot program at the middle school level in which representatives from the university visits area schools and present a similar course. With access to class average grades, it will be possible to not only quantify student attitudes before and after the course, but also any changes to their capabilities in STEM subjects.

VI. References

American Association of University Women (AAUW). (2000). *Executive Sumary: Tech-Savvy: Educating girls in the new computer age.* Washington, DC.

Genesereth, M.R., and Nilsson, N.J. (1987) *Logical Foundations of Artificial Intelligence.* Morgan Kaufmann.

Melchior, A., Cutter, T., Cohen, F. (2004) “Evaluation of FIRST LEGO League.” Waltham, MA: Center for Youth and Communities, Heller Graduate School, Brandeis University.

Rusk, N., Resnick, M., Berg, R., Rezalla-Granlund, M. (2008) “New Pathways into Robotics: Strategies for Broadening Participation” *Journal of Science Education and Technology*, vol. 17, no. 1, pp 59-69.

Turbak, F., and Berg, R. (2002). “Robotic Design Studio: Exploring the Big Ideas of Engineering in a Liberal Arts Environment.” *Journal of Science Education and Technology*, vol. 11, no. 3 pp. 237-253.

SESSION

TEACHING METHODS + TEACHING SUPPORT SYSTEMS AND TOOLS + CASE STUDIES

Chair(s)

TBA

Teaching Hands-on Design in Systems Engineering Principles

Ken Ferens
Dept. of Electrical and Computer Engineering
University of Manitoba, Winnipeg, MB, Canada
Ken.Ferens@ad.umanitoba.ca

Abstract— Following a successful introduction of the first 100% hands-on course "ECE 3730 Principles of Embedded Systems Design" offered in the Department of Electrical Engineering in the winter term of the 2012-2013 year [1], a similar 100% hands-on course was implemented in the fall term of the 2012/2013 year in the Department of Computer Engineering at the University of Manitoba. Similar with the previous course, this course "ECE 3740 Systems Engineering Principles" was designed specifically to directly assess student performance particularly in the CEAB attributes of Design, Investigation, Problem Analysis, and Tools. Differently than the previous course, this course used the Digilent MX7cK microcontroller board [2] to design, develop, and implement a self-configuring clustered wireless sensor network (WSN). The assessment included in-depth direct evaluation of student performance in 5 sub-projects, consisting of three evaluative components and hands-on midterm test and final exam, which were performed in the laboratory in real-time.

Keywords—design, UML modeling; multitasking; and C, HTML, and AJAX code development, design thinking, project based learning, CEAB attributes.

I. Introduction

Since 2010 the Canadian Engineering Accreditation Board (CEAB) [1] has instituted a transitional and development period for new accreditation criteria, part of which requires undergraduate programs in Canadian Engineering educational institutions to develop methodologies that demonstrate that their graduates possess the specified 12 CEAB attributes:

1. A knowledge base for engineering
2. Problem analysis
3. Investigation (Validation)
4. Design
5. Use of engineering tools
6. Individual and teamwork
7. Communication skills
8. Professionalism
9. Impact of engineering on society and the environment
10. Ethics and equity
11. Economics and project management
12. Life-long learning

The transitional period ends in 2015, at which time the CEAB will require full compliance with the new accreditation criteria.

In response to this new CEAB attribute based assessment mandate, and taking advantage of the opportunity, the author re-designed the ECE 3740 Systems Engineering Principles course in the fall of 2012. The course was re-focused to provide students with hands-on design and development experience and to provide a method that directly trains and assesses student performance, particularly in the CEAB attributes of Design, Investigation, Problem Analysis, and Tools.

II. Model of Course Design

Previously, the course was taught using the lecture style approach, which consisted of 32 50-minute lectures and five 3-hour laboratory sessions. Among the course objectives was to teach students the following computer systems engineering principles: divide and conquer; incremental design; minimizing complexity; maximizing cohesion; abstraction; designing for reuse and reusing existing designs; and unit and system test plans and procedures. The course met with limited success because students were not given sufficient opportunity to experience these concepts due to the style and method by which the material was presented to students. The laboratories were hands-on, but they were limited to 3-hours and no complex project could be done in that amount of time.

To overcome this limitation, in the fall semester of 2012, a project-based learning (PBL) and 100% hands-on learning approach was implemented in the course "ECE 3740 Systems Engineering Principles" in the Department of

Electrical and Computer Engineering, University of Manitoba.

Systems Design

A large and complex project was chosen as the single project which students would work on throughout the entire course. The instructor broke down the project into 5 sub-projects, each of which consisted of three parts. Splitting the course project into smaller parts demonstrated to students how to practise and implement the principles of divide and conquer, incremental development, unit testing, minimizing complexity, and maximizing cohesion. The instructor guided students in how to think about solving a complex design problem, and how to implement a design process. These principles were not taught to students through the lecture style of teaching; rather, students experienced these concepts on their own during the design process. A small company design process was simulated, where the instructor played the role of the systems engineer and team leader, and the students played the role of engineers and team members. However, lectures were given to guide, strengthen, and reinforce the concepts, at the same time students were experiencing these principles. Since students designed, developed, and implemented the sub-projects, they were intimately involved in the procedure; thus, students more deeply understood how and why the project was broken down into smaller pieces, and the rational for doing so. For instance, the sub-projects demonstrated how a complex project may be split into cohesive parts, where each part contained only parts which belonged together, and the rest were kept out. Furthermore, the sup-projects were selected to minimize the associations with other parts, thus minimizing coupling between other parts. As such, the splitting of the project in this way demonstrated to students how each part may be reused in other completely different scenarios and projects. Anecdotal observations showed a higher level of learning, participation, and retention in students. It was noted that during the reinforcing lectures, students were more involved in class discussions; the lectures were transformed into peer-to-peer style discussions and debates about design alternatives and other design principles.

Tools Usage

By implementing a 100% hands-on approach to teaching, the CEAB attribute of Tools was more easily assessed. Each student was given the required hardware and software to implement the course project to take with them for the duration of the course. The hardware was of limited size so that it could fit in a *lunchbox* type of container, and students carried their lunchboxes like they would textbooks to class, labs, and wherever. In this way, students were able to work on their projects whenever they had spare time, either commuting to the University, during lunch, during class, in the library or study halls, or at home. Throughout the entire term, students were required to demonstrate, in person to the team leader (i.e., instructor) - in a hands-on fashion - their design, analysis, investigation, and tools usage abilities, using the supplied computer hardware and software. The instructor noted that there can be no better way of measuring students' ability to use computer systems tools, than by requiring students to directly demonstrate this ability not only once, but during the entire term. In this way, their developmental improvement was witnessed first-hand as well.

Problem Analysis

For each of the sub-projects students practised an iterative development procedure, which consisted of domain analysis; requirements gathering, review and analysis; modeling, translation and coding; unit and system testing; deployment, and management. Short, high-level descriptions of the sub-projects to be solved were created and intended to be incomplete, ill defined, and sometimes ambiguous to emulate real-life initial project descriptions. From these descriptions, students were required to uncover what was actually being requested by following the given systems engineering process. First, students needed to study the domain of the problem in order to become more familiar with the topic. Once a reasonable understanding of the domain was achieved, students then gathered additional problem definition statements from the stakeholders, which were sourced by the instructor and any other pertinent information source. Students reviewed and analyzed the statements to form them into proper statements of requirements. By experience (i.e., learning via the hard way) students realized the statements should be formatted as unambiguous, verifiable, identifiable, safe, consistent, and realistic. Throughout the domain analysis and requirements phases of the iterative procedure, students used appropriate knowledge and skills to identify, formulate, analyze, and solve the complex engineering problems of the sub-projects and the complete project. Students practiced this iterative systems engineering process throughout the course and project development - in the labs assignments, tests, and final exam.

Investigation

Students practiced investigation through the five projects, one midterm, and the final exam. The test and final exam were completely hands-on, and students needed to determine some sort of debugging method to get their code working properly. Many students discovered the use of breakpoints in the software. The instructor demonstrated several times in person to each student how to logically approach a problem. Many times students would consult with the instructor regarding a problem with their project by saying, "well, it doesn't work." The instructor guided the students by trying to get them to start thinking about locating a strategic point in the software to place a breakpoint.

"Carefully analyze the symptoms of the problem, determine what the program is supposed to be doing at each point in your program, and then determined a strategic point to place a breakpoint," the instructor would say to the students. "When the program stops at your breakpoint, analyze the contents of the registers and memory locations to determine if they contain the expected values," the instructor went on to say. "By stepping through each instruction following the breakpoint, you will then determine at which instruction the problem begins to manifest," the instructor finalized. By guiding students in person in this way, and going through the debugging process with them in person and in real-time, students really understood the procedure, and were able to solve future problems by themselves in this manner.

III. IMPLEMENTATION OF COURSE DESIGN

A wireless sensor network (WSN) for gathering sensor readings and reporting the data to a central base station on the Internet was chosen as the course project. This project was a more elaborate version of the project suggested by Microchip's TCP/IP Stack [2], a real-time vending machine monitoring system (people can monitor the inventory of a vending machine using a web browser anywhere on the Internet). However, this project may also be used in general sensing, monitoring, and control applications, such as real-time smart building monitoring and control, lake water quality monitoring, water content for flood prediction in the Red River Basin, shopping for dummies, and forest fire monitoring and control.

Diglent's MX7cK board was chosen to implement each node in the WSN. Features of the MX7cK include an 80 MHz 32-bit microcontroller, 10/100 Ethernet controller, USB controller, CAN controllers, SPI interface, I2C interface. Pmod expansion connectors allowed additional components, such as microphone, gyroscope, accelerometer, temperature, WiFi transceiver, and real-time clock (Fig. 1). The board may be powered and debugged through a single USB cable and USB connector on the board. This proved to be very convenient for students as they could carry the entire hardware in the given Mx7cK package.

The Mx7cK board does not come with a TCP/IP stack. Microchip's TCP/IP stack [2] was adapted to work on the MX7cK board [3]. The adaptation included many software and configuration modifications to make the TCP/IP stack compatible with the Mx7cK board [4]. Furthermore, since the TCP/IP stack only supports an SPI EEPROM, extensive code porting was required to make the use the I^2C EEPROM on the Mx7cK board with the TCP/IP stack.

A sequence of five projects incrementally developed a client-server based system, which was autonomously configured in two phases. In "Phase 1 Clustering Process" sensor nodes autonomously and automatically learned the identification of a server node (cluster head), joined the cluster, and became cluster members of the cluster head's cluster of nodes. After the clustering process phase, the cluster head broadcasted to the cluster members a TDMA schedule, which informed the cluster members their data communications slot. In "Phase 2 Data Communications" cluster members, periodically, according to their TDMA slot, reported their sensor readings to the cluster head. The cluster heads aggregated their data (i.e., computed averages), and sent the data to the gateway node (Master Node), which was connected to the Internet through a wired connection. The Master Node on demand relayed the aggregated sensor data to sink, which was implemented by a web page on a remote host on the Internet. The web page used AJAX to continuously update the web page according to a user's demand. In addition to data communications between sensor nodes and the web page, the system also allowed the user to specify and configure sensor node parameters remotely through the web page and AJAX. More details of the course project may be obtained from [4] and [5].

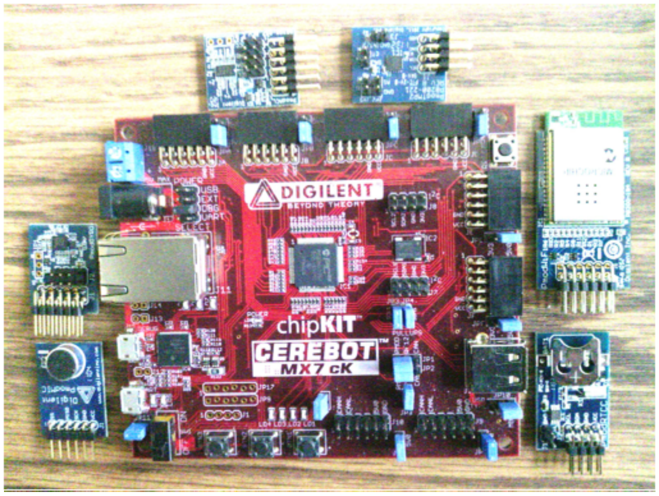


Fig. 1 Digilent Mx7cK microcontroller board, with various sensors: microphone, gyroscope, accelerometer, temperature, WiFi transceiver, and real-time clock.

Final Exam

The final exam directly assessed student performance in the CEAB attributes of Design, Problem Analysis, Investigation, and Tools. The exam was 100% "hands-on," and it was performed entirely in the laboratory. The placement of the workstations in the lab was done in a way to allow the invigilator to easily monitor and observer the performance of each student without being a distraction. Each student was assigned a development workstation, which had supporting software on it. Students brought their MX7cK microcontroller board, with Ethernet cable, to the exam, and used it to solve the given problem. A preparatory question on the exam asked students to demonstrate that "ping" was working on their system. This required students to make modifications to the TCP/IP stack software on the MX7cK board, configure the TCP/IP stack on their workstation appropriately, and to make the necessary connections between the MX7cK and the workstation. Students raised their hands when they were ready to demonstrate the ping functionality, and the invigilator examined their solution in real-time.

IV. Summary

This paper reports on the implementation of a project-based-learning and 100% hands-on approach to teaching ECE Systems Engineering Principles at the University of Manitoba. The objective of the course was to implement a wireless sensor network for gathering sensor readings at a remote location, and reporting the data to a central base station on the Internet. The course was designed specifically to provide a direct method of assessing student performance, particularly in the CEAB attributes of Design, Investigation, Problem Analysis, and Tools. Throughout the entire term, students demonstrated, in person - in a hands-on fashion - their design, analysis, investigation, and tools usage abilities, using the supplied computer hardware and software, which was given to each student at the start of the term to carry with them and take home for the duration of the course. Five sub-projects were designed, each of which contained three parts/deliverables. Each sub-project was designed to impart experience to students and to practise embedded systems development through "divide and conquer," incremental development, software and hardware reuse, maximizing cohesion, and minimizing complexity. The test and final exam were performed by students entirely in the laboratory; they brought their embedded systems hardware, solved the given hands-on problems of the test/exam, and demonstrated their solutions, in real-time. This novel methodology allowed the examiner to directly assess student performance in the CEAB attributes of Design, Analysis, Investigation, and Tools, because their designs and solutions were actually demonstrated in actual hardware and software, not just on paper, like the conventional approach to student tests/exams.

V. Acknowledgements

The author acknowledges the Department of Electrical and Computer Engineering for funding the hardware for ECE 3740, as well as supporting the project-based-learning and hands-on approach to teaching computer engineering courses in the Department of Electrical and Computer Engineering, University of Manitoba.

References

[1] Engineers Canada. (2013, May) Engineers Canada. [Online]. http://www.engineerscanada.ca/files/w_Accreditation_Criteria_Procedures_2012.pdf

[2] Microchip Technology Inc. (2012, May) Microchip Technology Inc. [Online]. http://www.microchip.com/

[3] Digilent Inc. (2013, January) Digilent Inc. [Online]. http://www.digilentinc.com

[4] Ken Ferens, "Teaching Design in Computer Engineering," in *Proc. of the Canadian Engineering Education Association (CEEA/ACEG) Conference*, Montreal, QB, 2013.

[5] Ken Ferens. (2012, September) ECE 3740 Systems Engineering Principles 1. [Online]. http://ece.eng.umanitoba.ca/undergraduate/ECE3740/

The development an interactive iBook application to harness the next generation of 'computer architecture' educational experiences

Fiona Carroll[1], **Nathan Thomas**[1], **and Mark Ware** [1]
[1] Department of Computing & Mathematics, University of South Wales, Pontypridd, Wales, UK, CF37 1DL.

Abstract - *Research shows that Higher Educational students have difficulties grasping a base knowledge of the subject Computer Architecture. In this paper we attempt to tackle some of the main challenges surrounding this issue. The authors present an interactive digital approach to the teaching of Computer Architecture; they feel that this approach is more aligned with their students' lives and more intuitive to how they experience and learn. The focus of the research lies in triggering a student's inner motivation for learning, designing for a new, engaging and interactive learning experience. The paper will describe the design and development of a Computer Architecture iBook application to enhance students learning of the subject. It will discuss the evaluation of the iBook and also the researchers' plans to take this work further.*

Keywords: Computer Architecture, learning, interactive, experience, iBook.

1 Introduction

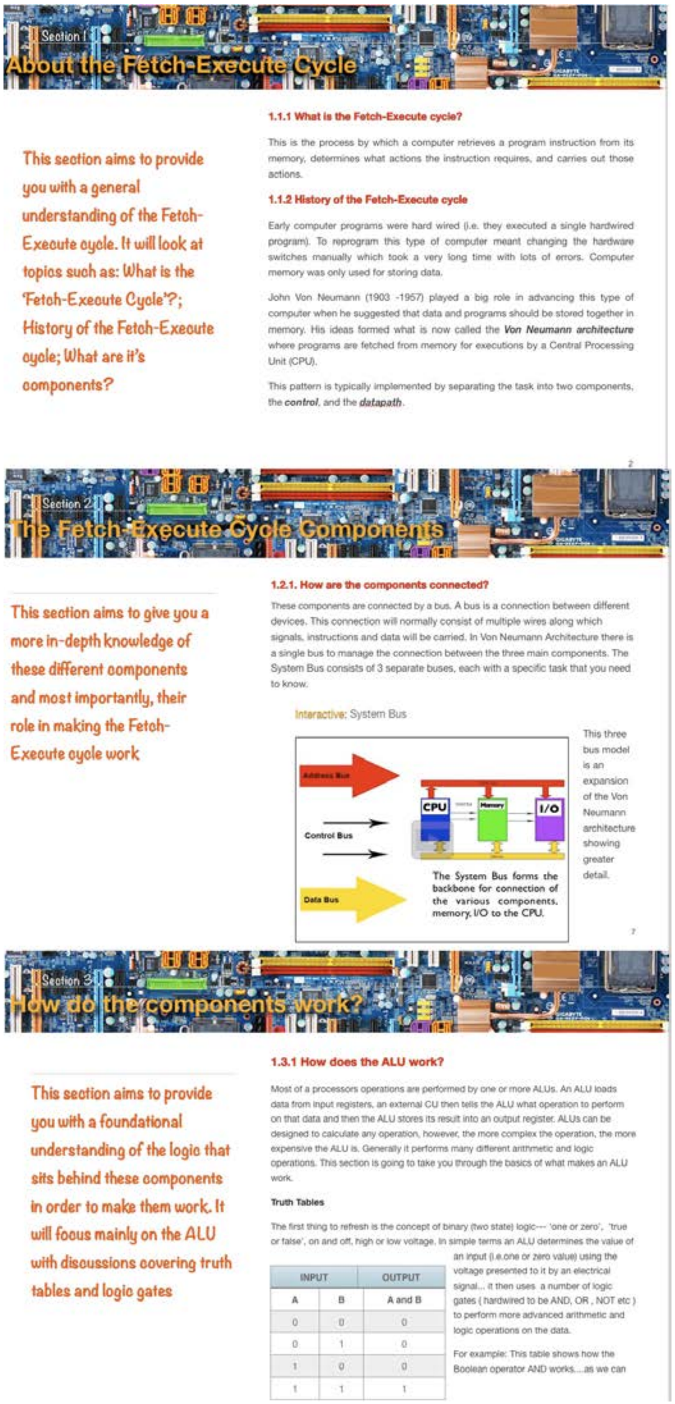
The academic literature records many difficulties encountered by Higher Education (HE) students studying Computer Architecture [1][2][3]. As [4] highlights, there is a worldwide need for better pedagogical methods and tools when it comes to the teaching of the subject. There is a need to teach the subject 'top down' as opposed to 'upside down' to ensure that all students are provided with enough 'cognitive hooks' to enable them to relate the new material to that of previous courses/experiences [5]. By creating a Computer Architecture iBook application the aim of this research is to tackle some of the main challenges surrounding the student's understanding of this traditional and at times complex area of computer science. What is unique and innovative about this work is how it proposes to weave stimulating multimedia content (such as text, photo, video, audio and animation) with new and exciting interaction activities of the iPad device. The aim is to help students absorb, fully visualise and understand the subject. The key lies in triggering the student's inner motivation for learning, designing for a new, engaging and active learning experience. This paper details the design and development of the first chapter of the Computer Architecture iBook application for HE first year Computer Architecture students. It will discuss the evaluation of the work and also the researchers' plans to take this work further.

2 Building the iBook learning experience

This research was funded by the University of South Wales' CELT (Centre for Excellence in Learning & Teaching) Innovation in Learning and Teaching Grants (2012 – 2013) to explore the interactive iBook technology for HE. A Learning, Research and Development Framework was used as guidance in the design of the iBook infrastructure, with special emphasis on the research, design, implementation, and evaluation of the 'fetch-execute cycle' chapter [6][7].

Placing a strong emphasis on engagement, the authors approach the subject of Computer Architecture from the 'top down' [5] and in doing so aim to tackle some of the main challenges with learning the subject. In detail, the fetch-execute cycle chapter has been strategically divided into three sections: the first section gives a general overview of the cycle, the second section aims to probe a little deeper into the components involved in the cycle and their relationships, and the last section focuses in detail on the ALU (an important component of the cycle) (see Figure 1). This structure has been based on Barthes' Narrative Structure Model (1977) [1]. The authors are particularly interested in using this model to support engagement – providing the students with 'cognitive hooks' [5] that will enable them to make connections – whilst also adding value to their learning experience.

[1] In his early work, [8] described narratives as a hierarchy of instances and he identified three levels – functions, actions and narration: narration on the top level, actions in the middle and functions on the bottom level. He explains that a function is the smallest unit of narrative that only holds meaning in so far as it combines with the other units, on the same level or on a higher level.



Section 1

About the Fetch-Execute Cycle

This section aims to provide you with a general understanding of the Fetch-Execute cycle. It will look at topics such as: What is the 'Fetch-Execute Cycle'?; History of the Fetch-Execute cycle; What are it's components?

1.1.1 What is the Fetch-Execute cycle?

This is the process by which a computer retrieves a program instruction from its memory, determines what actions the instruction requires, and carries out those actions.

1.1.2 History of the Fetch-Execute cycle

Early computer programs were hard wired (i.e. they executed a single hardwired program). To reprogram this type of computer meant changing the hardware switches manually which took a very long time with lots of errors. Computer memory was only used for storing data.

John Von Neumann (1903 -1957) played a big role in advancing this type of computer when he suggested that data and programs should be stored together in memory. His ideas formed what is now called the ***Von Neumann architecture*** where programs are fetched from memory for executions by a Central Processing Unit (CPU).

This pattern is typically implemented by separating the task into two components, the ***control***, and the ***datapath***.

2

Section 2

The Fetch-Execute Cycle Components

This section aims to give you a more in-depth knowledge of these different components and most importantly, their role in making the Fetch-Execute cycle work

1.2.1. How are the components connected?

These components are connected by a bus. A bus is a connection between different devices. This connection will normally consist of multiple wires along which signals, instructions and data will be carried. In Von Neumann Architecture there is a single bus to manage the connection between the three main components. The System Bus consists of 3 separate buses, each with a specific task that you need to know.

Interactive: System Bus

The System Bus forms the backbone for connection of the various components, memory, I/O to the CPU.

This three bus model is an expansion of the Von Neumann architecture showing greater detail.

7

Section 3

How do the components work?

This section aims to provide you with a foundational understanding of the logic that sits behind these components in order to make them work. It will focus mainly on the ALU with discussions covering truth tables and logic gates

1.3.1 How does the ALU work?

Most of a processors operations are performed by one or more ALUs. An ALU loads data from input registers, an external CU then tells the ALU what operation to perform on that data and then the ALU stores its result into an output register. ALUs can be designed to calculate any operation, however, the more complex the operation, the more expensive the ALU is. Generally it performs many different arithmetic and logic operations. This section is going to take you through the basics of what makes an ALU work.

Truth Tables

The first thing to refresh is the concept of binary (two state) logic--- 'one or zero', 'true or false', on and off, high or low voltage. In simple terms an ALU determines the value of an input (i.e.one or zero value) using the voltage presented to it by an electrical signal... it then uses a number of logic gates (hardwired to be AND, OR , NOT etc) to perform more advanced arithmetic and logic operations on the data.

INPUT		OUTPUT
A	B	A and B
0	0	0
0	1	0
1	0	0
1	1	1

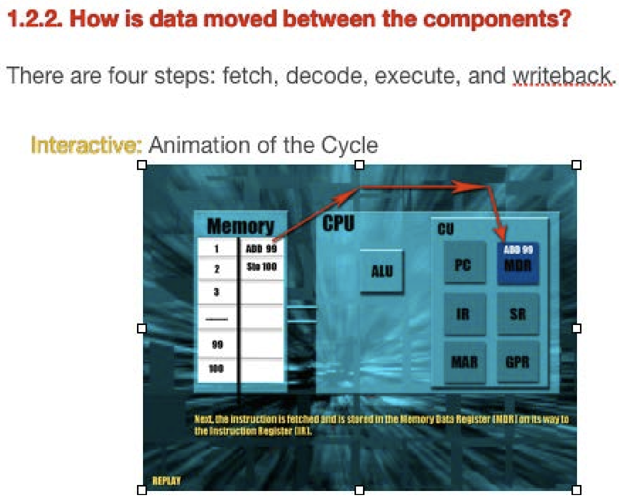
For example: This table shows how the Boolean operator AND works....as we can

Figure 1: The three sections of the iBook application

In conjunction with engagement, interaction also plays a crucial role in the design of the learning experience of this iBook. As [9] point out 'Interaction is a fundamental component of learning'. The authors of this paper focus on the learner-content interaction to encourage students to think deeply about the information they encounter. They explore the interactivity offered by a range of multimedia (i.e. video, interactive animation, interactive graphics, interactive keynote presentations and interactive quizzes) to ensure that all students have the right (and enough) opportunities to interact with and get to grips with the content. For example, video was used to teach students about the fetch-execute cycle; three lecturers from the University's School of Computing were filmed reading from two strategically different genre of script. The aim of this was to actively engage students with the subject by providing them with different opportunities to effectively break down some of the difficulty surrounding the computer architecture components and the relationships between them (see Figure 2). The first script was designed to be short, concise and informative, primarily describing/ defining a feature/ component of the fetch-execute cycle. The second type of script was designed to be more playful, in that, certain components of the fetch-execute cycle (i.e. Control unit and Arithmetic Logic unit) were personified by the lecturers. This was based on the Little Man Computer (LMC) a model that was developed by Stuart Madnick of MIT [10]. The rationale behind these scripts was to provide students with two routes to understanding the workings of the fetch-execute cycle (i.e. exact definitions and descriptions of the fetch-execute cycle as well as an accessible model that represents the architecture of the cycle). In total, ten clips (short and manageable clips of between 2-5 minutes in length) were created and five were used and integrated through the main body of the iBook.

Figure 2: The three lecturers from the School of Computing

As [11] believes, deep and meaningful learning takes place when student-content interaction is supported at a high level; by high levels he means students to be personally active in the interaction, actively engaged with the content. To facilitate this high level of interaction, the authors aim to facilitate the interaction with the content through many different channels. Therefore, in addition to the video, interactive animation was also used in the book to detail, define but most importantly bring to life each component involved in the fetch execute cycle and to animate each step involved in the process. As you can see from Figure 3, each step is highlighted visually and textually within the animation to ensure that the learner is fully aware of and engaged in what is happening in each step of the cycle. Interactive graphics are also used in the iBook to highlight certain areas of the fetch-execute components. For example, the graphic of the ALU can be zoomed in and out, to further engage the learner in understanding the mechanics between this and other components.



1.2.2. How is data moved between the components?

There are four steps: fetch, decode, execute, and writeback.

Interactive: Animation of the Cycle

Next, the instruction is fetched and is stored in the Memory Data Register (MDR) on its way to the Instruction Register (IR).

Figure 3: Interactive animation

Also, an Interactive keynote presentation is embedded in the iBook application to highlight the different buses within the fetch-execute cycle and to detail their different purposes. The learner can interact with the different buses to activate a corresponding explanation/description. Again, the objective here is to get the learner to interact with the content and in doing so, encourage them to further absorb the material. In conjunction with these, an interactive review has been placed at the end of each of the three sections. Each review has approximately ten questions. These questions have different formats (e.g. multiple choice, drag and drop etc) and take an open book approach in that students can check and redo their answers. The rationale for this approach is that it encourages learners to move away from surface learning and gets them interacting further with the material to find their answers.

3 Evaluating the iBook learning experience

The aim of this study is to evaluate the Computer Architecture iBook application, particularly in terms of its potential to enhance the learning/understanding of the subject of Computer Architecture. A total of 14 participants (13 male and 1 female) took part in the study. Their ages ranged from eighteen to fifty years and they are all recorded to be studying and/or have studied the subject of Computer Architecture. For the study, each participant was briefly shown the Computer Architecture iBook application and then asked to explore the iBook application individually for five-ten minutes. After this, each participant was asked to complete a short questionnaire to provide insight into their perceptions of the iBook in relation to their own learning experience of the subject.

3.1 Results

From the qualitative data collected, it is clear that the majority of participants were impressed with the iBook application and had a positive view on its potential to effectively enhance their understanding of the Computer Architecture topic. One student (Participant 13) went as far as saying that the iBook application will change the way we learn whilst helping us learn in new ways. As we can see from some of the responses, the overall participants' response/attitude to the actual design and structure of the iBook application was very encouraging:

- The iBook is well structured and completely flexible. It makes good use of the iPad functionality and does not make itself too complicated. (Participant 1)

- Very well structured with images, animations and video to break up the text. (Participant 2)

- I thought the app was really well designed with good use of colour, it was also well laid out with enough information per page and engaging level of interactivity (Participant 3)

- Easy to use, no learning needed, can just pick up and go. (Participant 4)

- Simple, effective layout which was pleasing and easy on the eye interaction was good: there wasn't too much but not too little either…The colour scheme was suitable and relaxing, easy on the eye and the text was easy to read. (Participant 6)

- Overall I found the application very easy to use and informative. A job well done! (Participant 7)

- The app worked well, good layout into sections and had an overview of what the section was about (Participant 8)

- I think it is an all round quality iBook application and it has good animation as well as interactivity. (Participant 11)

- I think the material is put together very well. The contents are very clear and the interactive video is very engaging. (Participant 14)

In terms of its use in education and most importantly, its potential to enhance the learning of the subject of Computer Architecture, the feedback was also very positive. As one student (Participant 1), summed it up…'this would make a good learning tool for students as it is much more interactive than PowerPoint slides and encourages the user to interact with the information on the screen'.

Some of the other responses reinforce this point:

- … if the whole class is using it, people can all learn at their own speeds and there is support for the inbuilt iOS dictionary, allowing students to look up definition if a lecturer is busy etc. (Participant 2)

- As a teaching resource I feel the app would be of use during a lecture to gain more information on what is being discussed. I feel it would also perform well as a revision aid. (Participant 3)

- The Quiz function works really well and helps user to highlight problem areas. Would work well as a tool alongside lectures/ labs sessions. (Participant 4)

- I think the iBook computer architecture app is useful as an education tool as it explains all you need to know about the subject with words, images and quizzes. (Participant 5)

- I think that this application would be very useful in lectures and labs. It will help the end user understand the subject or the current topic. (Participant 7)

- It is a very creative app that can be easily used in class room sessions (Participant 12)

- I think it is good to use in lectures because it is easy to use and has good features on it. (Participant 13)

- Yes, I think it will do very well as a lecture notes and learning material. (Participant 14)

However, in terms of how it actually enhances their learning, participants had different ideas. Some felt that it had something to do with the different interactions and the use of different media (i.e. graphic, animation, video etc). Others felt it is the interactive quiz at the end of each section which enhances the learner's confidence in their learning.

- The app was very easy to use and brief enough to profile all the information necessary (Participant 3)

- Good interactive features to keep user engaged in topic…Uses different memory techniques (sound, picture, interaction) to aid learning (Participant 4)

- I think this app can be used in a lecture as it gives sufficient knowledge of the subject and allows you to test your knowledge through pictures/ videos and quizzes. (Participant 5)

- After using the app I have found it very informative. The app consisted of various components such as interactive diagrams and videos. The videos were very informative and interactive. (Participant 7)

- The app does enhance my knowledge on the subject. The review questions at the end is a good idea. (Participant 9)

Overall, this study has allowed us to identify some of the main impressions of first year students after using the iBook application on Computer Architecture. More importantly, it has given us an insight into the iBook application and its ability to teach complex topics like Computer Architecture. Through interaction (i.e. video, animation and interactive quiz), several participants felt that the iBook application has the potential to help them with the understanding of complex topics. In fact, one student mentioned that changing the learning process to one which is more interactive is more beneficial to their learning (participant 4), another student noted that more interactivity will encourage people to read and learn more (participant 5) and that video makes the topics more interesting (participant 8). In summary, this study has highlighted the participant's feelings on the important role of 'engagement' and 'interaction' in their learning experience.

4 Conclusions

This paper explores new ways to support and enhance the learning of the core and often difficult concepts of Computer Architecture. As the study shows, it is the creative yet very tangible interactive approach that the iBook takes to Computer Architecture that the authors envision as having the potential to enhance the learning of the subject. They feel that by breaking away from descriptive lesson plans and injecting excitement as well as intellectual rigour back into the learning of the subject, the iBook has the potential to tackle some of the difficulties students are experiencing with the subject. As the study highlights, interaction and engagement were two terms used when participants were reflecting on their experience of the iBook application. As one student says 'I think iBooks will change the way we learn primarily due to its interactive nature. It is much easier to learn when you can interact with your knowledge source'. The continued development of this iBook depends on further funding; however, the authors of this paper believe the development of a complete interactive multi-modal e-textbook course on Computer Architecture to be the natural course for future research. By giving students and lecturers open access to an engaging, interactive and flexible educational resource they are confident that there is the opportunity to reshape not only the learning but also the teaching of Computer Architecture on a global scale.

5 References

[1] L. Cassel, D. Kumar, K. Bolding, J. Davies, M. Holliday, J. Impagliazzo. "Distributed expertise for teaching computer organization and architecture" (ITiCSE 2000 Working Group Report). ACM SIG-CSE Bulletin, Vol. 33, No. 2, 2000, pp. 111 – 126

[2] O. Mirmotahari, C. Holmboe, J. Kaasbøll. "Difficulties learning computer architecture" in proceedings of ITiCSE '03 Proceedings of the 8th annual conference on Innovation and technology in computer science education. Pages 247 – 247. ACM New York, NY, USA, 2003. Available online: http://delivery.acm.org/10.1145/970000/961606/p247-mirmotahari.pdf?ip=193.63.148.21&acc=ACTIVE%20SERVICE&CFID=173193260&CFTOKEN=18493437&__acm__=1358942453_fba6bba0c9f5648edb0d20b6bdeaf8ed (23/1/2013)

[3] C. Yehezkel, R, Leibovitch & I. Levin. "Reinforcing and Enhancing Understanding of Students in Learning Computer Architecture". In Book Informing Science and Information Technology. Volume 8, 2011.

[4] E. Z. Bem. "A case for teaching computer architecture". In Proceedings of the Fifth Australasian Conference on Computing Education - Volume 20 (Adelaide, Australia). T. Greening and R. Lister, Eds. Conferences in Research and Practice in Information Technology Series, vol. 140. Australian Computer Society, Darlinghurst, 2003, Australia, pp. 1-7. Available online : http://dl.acm.org/citation.cfm?id=858404 (Accessed on 30/11/2011)

[5] Greg W. Scragg. "Most computer organization courses are built upside down". SIGCSE Bulletin, 23, 1, 1991, 341-34. Available online: http://delivery.acm.org/10.1145/110000/107079/p341-scragg.pdf?ip=193.63.148.21&acc=ACTIVE%20SERVICE&CFID=173193260&CFTOKEN=18493437&__acm__=1358942786_e300e9025520a42f7e5c67f84b43627b (23/1/2013)

[6] N. Thomas, F. Carroll & R. Kop. "iBook learning experience: the challenge of teaching computer architecture to first year university students". In Proceedings: WORLDCOMP'12- The 2012 World Congress in Computer Science, Computer Engineering, and Applied Computing, July 16-19, 2012, USA.

[7] F. Carroll & R. Kop "A Learning, Research and Development Framework to Design for a Holistic Learning Experience", E-Learning and Digital Media Journal. 2011. ISSN 2042-7530.

[8] Roland Barthes. "Image, Music, Text". London: Fontana Paperbacks, 1977.

[9] M. Murray, J. Pérez, D. Geist, A. Hedrick. "Student Interaction with Online Course Content: Build It and They Might Come". Journal of Information Technology Education Research 11, 1. 2012 Available online: http://www.jite.org/documents/Vol11/JITEv11p125-140Murray1095.pdf (11/2/2013)

[10] I. Englander. "The Little Man Computer. The Architecture of Computer Hardware and Software Systems: An Information Technology Approach". John Wiley & Sons: UK, 2009.

[11] T. Anderson. "Getting the mix right again: An updated and theoretical rationale for interaction". In-ternational Review of Research in Open and Distance Learning, 4(2), 2003. Available from http://www.irrodl.org/index.php/irrodl/article/view/149/708

Using BDI-extended NetLogo Agents in Undergraduate CS Research and Teaching

Jonathan Wiens
Computer Science Dept.
Faculty of Cooperative Studies
Berlin School of Economics and Law, Germany
j.wiens@stud.hwr-berlin.de

Dagmar Monett*
Computer Science Dept.
Faculty of Cooperative Studies
Berlin School of Economics and Law, Germany
Dagmar.Monett-Diaz@hwr-berlin.de

***Abstract*—This paper introduces both the subject and the research results of an undergraduate student project. The project focuses on an agent architecture that implements the beliefs-desires-intentions (BDI) model. It proposes a new way to extend a BDI library in NetLogo based on a case study from Computational Economics. Furthermore, this work presents how such an undergraduate research supports Artificial Intelligence teaching activities at the Berlin School of Economics and Law.**

***Keywords*—Multi-agent systems, BDI agents, NetLogo, undergraduate research, teaching.**

I. Introduction

In dynamic environments like economic models, telecommunication networks or biochemical organisms, simulation processes often rely on multi-agent systems (MAS). MAS play a major part both in Artificial Intelligence (AI) curriculum and research. They are a basis for many commercial models simulating complex non-deterministic behavior. Just as developing simulations of complex environments with MAS is an inherent difficult task, teaching by undergraduates and conducting programming MAS is not an easy one, due to a necessary high time exposure for MAS learning, design and implementation.

The rest of the paper continues as follows: Section II presents both a theoretical background of the beliefs-desires-intentions (BDI) model and NetLogo, a multi-agent programmable modeling environment. In Section III, a case study from the area of Agent-based Computational Economics (ACE) is introduced. Fractional reserve banking supports the need to extend an existing BDI library for NetLogo, which is the topic of an undergraduate research project at the Berlin School of Economics and Law (BSEL). It is presented in Section IV. Finally, Section VI gives insights concerning our experiences in undergraduate Computer Science (CS) education, when teaching these subjects in an AI course at the BSEL.

II. Theoretical background

This section introduces a practical reasoning model for agents with future-oriented intentions, the BDI model in NetLogo. NetLogo [1] is a multi-agent programmable modeling environment that aims at providing standards for high power users at low learning curves. Thus, it serves as an excellent teaching tool to discover basic concepts of agent programming with a hands-on approach.

To date, a third-party library providing functions for the implementation of BDI agents written in native NetLogo code has been published [2]. This library, however, only includes belief revision procedures and intention handling and execution, thus missing other elemental features that the BDI approach encompasses. Subsequently, the main topic of the undergraduate student project was to enhance the existing library in order to acquaint students with a more sophisticated form of the BDI model in NetLogo. Furthermore, the improved library can be used not only by students but also by anyone interested in building BDI agents in NetLogo.

The major goals of the undergraduate student research project are:

- to propose a simplified model for fractional reserve banking,
- to integrate BDI concepts,
- to design a MAS in NetLogo for the new model,
- to analyze which BDI components cannot be modeled using the available BDI library,
- to propose extensions to that BDI library for covering the discovered needs,
- to design, implement, test and evaluate those changes using NetLogo, and
- to support the AI teaching of these contents both theoretical and practically.

All these goals were successfully achieved and are topic of the sections that follow.

A. BDI

In 1987, the philosopher Michael Bratman introduced a model for human practical reasoning as a way to explain future-directed intentions [3] which led to the BDI model in AI. First, a logic was formally instituted in 1991 [4] and later, a practical model was implemented as an approach to build highly dynamic applications, like the OASIS air-traffic management system, the Optimal Aircraft Sequencing using Intelligent Scheduling [5].

* Contact author.

The BDI approach focuses on the reasoning of resource-bounded agents, while resource boundaries include limited computational power of both the agent and the dynamic environment. This implies that (i) while the environment changes, the agent cannot take infinite time to plan its actions but needs to be flexible in re-planning and (ii) the agent may not reason open-ended because of world changes that lead to new dangers or opportunities.

At the center of such BDI agents lies the notion of mental attitudes, namely beliefs, desires and intentions. Beliefs are the agents representation of assumptions about the world, the environment. Beliefs are not facts about the world, since beliefs are subject to error of reception. Thus, beliefs can change and can be combined to infer new beliefs. Desires represent the goals of the agent and thereby ultimately define an agent's purpose. An agent can have different goals, competing for each other. One difficulty in BDI is how to distinguish which goals are most *beneficial* to the agent. The substance of desires are intentions, which describe how to reach the deliberated goals, like directions in a recipe. While it can be assumed that for most desires there are several intentions, a selector is needed to determine which intention is appropriate at a given time. Intentions are, in other words, the actions necessary to fulfill the agent's goals. A sequence of intentions is called a plan.

B. NetLogo

If BDI is an architecture describing how to build a specific type of agents, then NetLogo is the software developer tool kit providing a framework to design the agents and their environment. NetLogo is a multi-agent programming language with an integrated development environment (IDE) for simulating complex phenomena for both research and educational purposes. It is used across a wide range of disciplines and university levels and is particularly well suited for multi-agent systems that evolve over time.

Compliant with it's motto "low threshold, no ceiling", NetLogo allows the modeling of hundreds and thousands of autonomous agents, all operating concurrently. The graphical user interface allows the developing of quick simulations of two and three dimensional spatial behavior of agents, making the process of designing MAS visual and therefore easier to understand and troubleshoot, but also entertaining and appealing [1].

To date, there are several languages for programming BDI agents that are highly sophisticated and offer different components and approaches to implement multi-BDI agent systems [6]. These environments, though powerful and advanced, have steep learning curves and are unsuitable for educational purposes that involve a hands-on, short-term curriculum. Because NetLogo is an excellent tool for teaching, for researching and for designing MAS, the notion to create BDI agents with NetLogo comes natural.

C. BDI Library for NetLogo

The library for realizing BDI agents in NetLogo was written by Sakellariou [2] and its current version is NetLogo 4 compatible. It consists of 23 *commands* and *reporters*[1] and is split into three logical tiers: *belief management*, *intention management* and *utilities*. Utilities support the belief and intention management with, for example, timeout interruptions.

1) Belief Management: A belief consists of two elements, stored in a list: the *type* and the *content*. The type of the belief describes what kind of belief it is or what class it belongs to. For example, a type of a belief could be any string like `"location"` or `"capital"`.

The content declares the actual value of the belief type and can be of any NetLogo structure.[2] Notice that there can be competing beliefs with the same type but different content. This makes sense when a belief can hold multiple attributes. An agent could believe, for example, that the coffee is hot (type `"coffee"`, content `"hot"`) and strong (type `"coffee"`, content `"strong"`) at the same time.

The following example shows three sets of beliefs, as part of one list:

```
[["location" [3 4]] ["coffee" "hot"]
 ["coffee" "strong"]]
```

2) Intention Management: Like a belief, an intention consists of two elements: the *name*, which maps to a NetLogo command, and a *done-condition*, which maps to a NetLogo reporter. Intentions are stored in a stack and are popped out when to be executed. If the done-condition is satisfied, the intention is removed and the next intention is popped out consecutively.

An example intention stack in NetLogo is the following:

```
[["task [do-nothing]" "true"]
 ["task [move-to [0 3]]"
  "task [at-location [0 3]]"]
 ["task collect-wood" "task wood-loaded"]]
```

The name and done-condition of an intention are stored in *tasks* – a task is a special NetLogo structure that is a placeholder for runnable code, in other programming languages also known as lambda, closures or first-class functions.

III. A Case Study

Having covered so far the theoretical background of BDI, NetLogo and the available NetLogo BDI library, the next step is to use the library by designing a sufficient complex model in order to exhaust the library's functionalities and, consequently, to demonstrate in what ways it can be extended. The Agent-based Computational Economics (ACE) interdisciplinary research field promises scenarios that can be as complex as human action itself and therefore suits the purpose of building a complex model in NetLogo.

[1] In NetLogo, commands describe the actions an agent should carry out resulting in some effect, while reporters include instructions to compute a value, which is then reported by the agent.

[2] Like booleans, strings, integers, doubles, lists, etc.

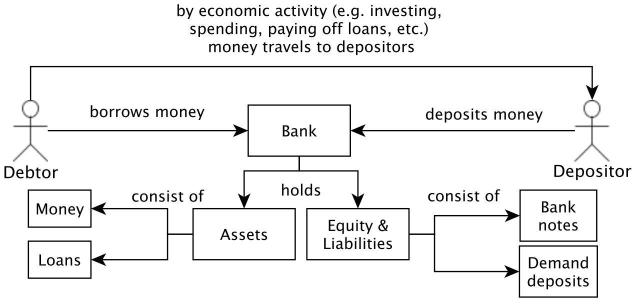


Fig. 1. Fractional reserve banking simplified.

A. Agent-based Computational Economics

Agent-based computational economics is “the computational study of dynamic economic systems modeled as virtual worlds of interacting agents” [7], while agents can represent individuals (e.g. people) or abstract entities (e.g. a market interest rate). Like with other MAS, the modeler provides the agents with an initial configuration and lets the computational world evolve over time, as its constituent agents interact with each other. Initial configuration, depending on the subject and desired complexity of the model, includes any variable that is specified to reach the desired goal. Hence, the scientific goal is to “[...] test theoretical findings against real-world data in ways that permit empirically supported theories to cumulate over time” [8].

ACE has been applied to several research areas, like asset pricing [9], transaction costs [10], and macroeconomics [11], to name a few. The economic subject of the NetLogo model we present is this paper is fractional reserve banking (FRB), which offers an arbitrary amount of complexity depending on the modeler’s choice.

B. Fractional Reserve Banking

In economics, the chief activity of a bank is, as a financial intermediary, to receive deposits from savers at a given deposit interest rate and to channel these deposits by lending them at a higher interest rate to credit-worthy or productive borrowers, thus making a profit. FRB is the process when banks keep only a fraction of the received deposits, called the reserves, and lend the remaining to borrowers. Subsequently, by spending, investing or doing other economic activity, part of the money lent is deposited within the same or another bank, allowing further lending (see Figure 1).

As a result, the bank holds more liabilities than assets at a growing rate. This process increases the money supply over time, leading to inflation and currency debasement. The degree of the effects is mainly influenced by the reserve ratio, which is called the reserves-to-warehouse-receipts quotient [12]:

$$\text{Reserve ratio} = \frac{\text{Reserves}}{\text{Warehouse receipts}}$$

Warehouse receipts are the outstanding bank notes the bank is required to redeem on demand to the depositor, usually stored in a checking account. The depositor is able to withdraw on his account anytime and anywhere checks are accepted as payment. However, if the liabilities exceed the assets, then the depositor will not be able to withdraw his entire deposit, since the bank only holds a fraction of it, hence the name fractional reserve banking.

TABLE I. EXAMPLE BALANCE SHEET OF FRACTIONAL RESERVE BANKING.

Bank			
Assets		Equity & Liabilities	
Gold	\$50,000	Warehouse	
Loans	\$80,000	receipts for gold	\$130,000
Total Assets	\$130,000	Total Liabilities	\$130,000

a) An example for FRB: In the absence of a central bank, and for the sake of simplicity, a bank has \$50,000 of gold[3] deposited and issues \$80,000 in warehouse receipts, it lends them out and expects to be repaid the \$80,000 plus interest (see Table I).

The bank lends \$80,000 of warehouse receipts it doesn’t own. The general money supply is thus increased by the amount of the credit (i.e. \$80,000), the fraction being $\frac{5}{13}$ (from $\frac{\$50,000}{\$130,000}$). Thus, FRB generates an increase in the money supply by issuing warehouse receipts for money that did not exist previously. Money in circulation has increased by the amount of warehouse receipts issued beyond the supply of gold in the bank. The lower the fraction of the reserve, the greater the amount of new money issued and, therefore, the greater the decrease of the purchasing power of the dollar. This is why FRB has a number of negative effects on the economy, in terms of interest rates, inflation and the purchasing power of the dollar in form of currency debasement. The fact that banks are never able to redeem the entire deposits of all depositors has three implications:

- the bank is at all times bankrupt, since it cannot redeem all the deposits it owes to the depositors, hence
- the depositors statistically never demand to redeem all their deposits at once, unless
- they lose confidence, or trust, in the bank and, therefore, want to redeem all their deposits, resulting in a *bank run*.

A bank run is the phenomena when a large number of depositors withdraw their deposits due to loss of confidence in the bank or when they speculate that the bank might have become insolvent. As the bank run progresses, it develops momentum in a positive-feedback loop as more customers withdraw their deposits, thus prompting further withdrawals from other customers. This ultimately results in the bank’s insolvency, as it fails to provide the money demanded.

IV. NetLogo Model for FRB

The simplified model for FRB derived in a complex enough model for BDI in NetLogo. The purpose of the FRB model is to depict the implications fractional reserve banking has over time. Its current stage is kept to a simple world design without inter-bank activities, a central bank and government regulation.

[3] Whether gold or government paper does not matter here.

As a consequence, this model reflects by no means a real-world example but is merely an extract of modern banking to illustrate, on a low-level, what effects fractional reserve banking has on both interest rates and currency debasement.

Since MAS evolve over time, the FRB model is put into a time frame that is measured in *ticks*, discrete steps that are used in most NetLogo models. Figure 2 shows the model with the concurrent activities that are considered within a tick cycle. At the heart of the model are three parties: the debtors, the depositors and the bank. As described in Section III-B, the bank plays the role of the intermediary between the depositor and the debtor, handling the deposits, loans and their corresponding interest rates. Each party deliberates once per tick. The deliberation process determines, according to the agent's personal value scale (PVS) what it will do. At the end of each cycle there is one possible action per agent.

A. Depositor

The depositor's motivation is to accumulate as much capital as possible without running the risk of the bank's insolvency, which would cause him immediate loss of capital. Therefore, there are four different general actions a depositor can accomplish[4] (see Figure 3).

Since the actions of the depositor are determined by his PVS, the PVS should be designed carefully. It is the result of the agent's algorithm and embodies what would be most reasonable to undertake. It would be unreasonable, for example, for a depositor to deposit 70% of his on-hand cash into a bank he has 10% of trust in. The goal is to project the depositor's most reasonable behavior using an algorithm that takes account all influencing variables, like his time preference p, his current capital C, the deposit interest rate β and the trust in the bank t, in a feasible way. In order to design an algorithm that captures all of these parameters in a balanced relationship, an iterative process of implementing, testing and adapting is necessary. The following pseudo-code shows the result of this process:

```
if $0.5 \leq t \leq 1$ then
    $D = C \cdot (p + \beta)$
else if $0.3 \leq t < 0.5$ then
    $W = S \cdot (1 - p)$
else
    $W = S$
```

where D is the amount to deposit, W is the amount to withdraw, and S is the current deposits or savings in the bank.

The algorithm relies heavily on the agent's trust into the bank he's engaged with. If it falls to a certain amount, he will refuse to deposit but rather withdraw, even to the point of withdrawing the entire deposit, if the confidence in the bank is lost. Both the time preference and interest rate determine what amount of capital should be deposited or deposits withdrawn. The time preference is an attribute that cannot be deduced by computational methods, since it has a non-deterministic character. Therefore, the time preference is generated through a randomized procedure for each agent individually.

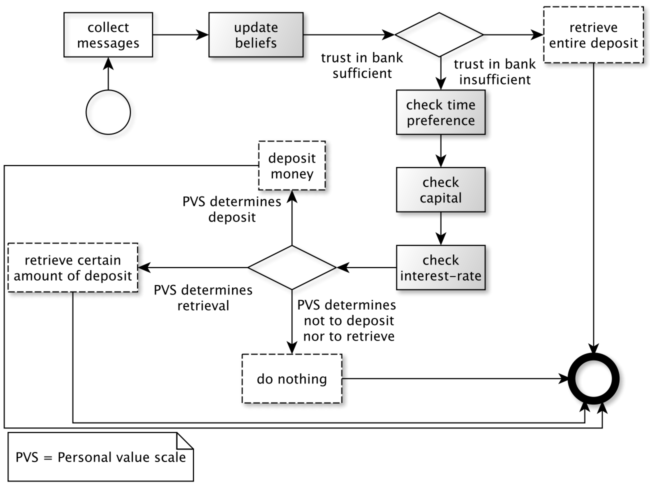


Fig. 3. Deliberation process for the depositor, including evaluation of beliefs (grey boxes) resulting in four general intentions (boxes in dashed lines).

B. Debtor

The debtor's deliberation process is similar to the depositor's. The debtor's four possible actions are to deleverage by a certain amount, to lend a specific or the maximum amount of money, or to do nothing. They were designed and implemented analogously to the depositor (for more information, refer to the student research project documentation [13]).

C. Bank

The bank's main procedures are to accept deposits, to issue withdrawals, to lend, to deleverage and, corresponding to these, to compute profitable deposit and loan interest rates, as well as to compute the reserves, liabilities and assets. The goal of the bank is to make profit through interest rates by maintaining the loan interest rates greater than the deposit interest rate. The figure of each rate is determined by the economic law of supply and demand. For example, if there is more demand than supply in the deposit business, then the bank will decrease its interest rate for deposits. If the demand diminishes, then the interest rate goes up for giving potential customers incentives to deposit.

V. Extention of the BDI Library

The case study in ACE resulted in the proposition for plans and redefining intentions. The original implementation of the BDI library relies solely on intentions for an agent's actions. Because the complexity of agents and the implementation and maintenance of simple intentions correlates, the modeler needs to create a new intention for each action and has no possibility to cluster intentions. Therefore, the primal reason to incorporate plans into the library is to group a sequence of actions, or intentions, and other plans. Like a knowledge base, a plan defines what needs to be done in order for the goal to be fulfilled.

The second major upgrade was the redefinition of intentions. The original implementation of intentions did not allow calls by reference. This has a strong influence on the agent's

[4] The four actions being to retrieve the entire deposit, retrieving a specific amount that is lower than the total of the deposit, depositing a certain amount, or doing nothing.

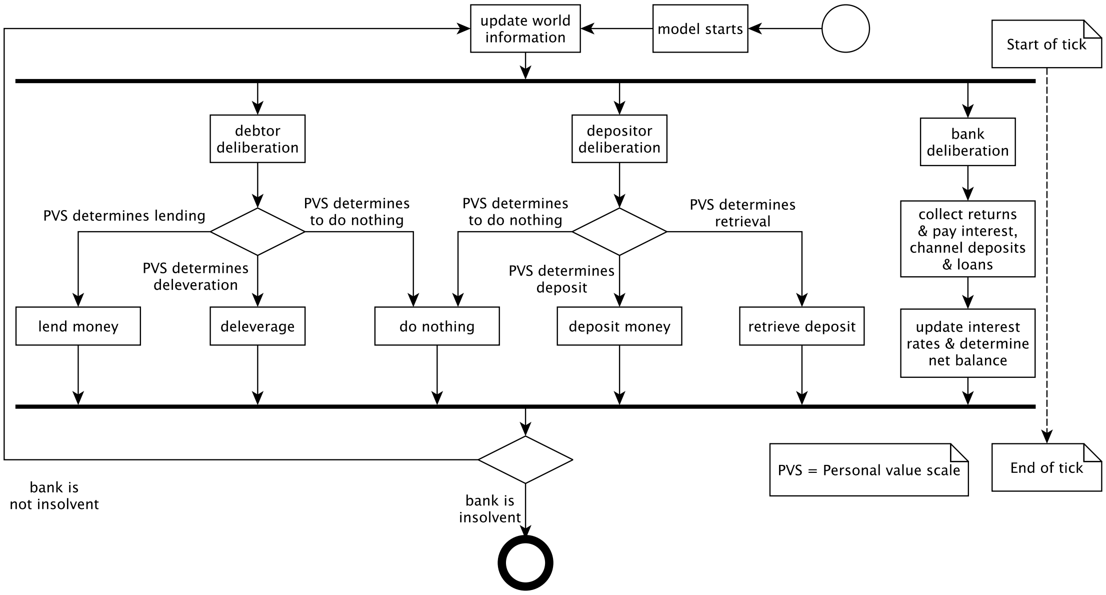


Fig. 2. Fractional reserve banking model overview.

capabilities, since it cannot act on its beliefs in an efficient manner.

An example of a plan as described for the extension can be written as follows:

```
create-plan (list boil-water
add-coffee-powder
stir-beverage drink-coffee)
task [coffee-finished?]
"make coffee"
```

The items `boil-water`, `add-coffee-powder`, `stir-beverage`, and `drink-coffee` refer to intentions that are executed consecutively in order to achieve the plan. The task `coffee-finished?` is the reporter to verify whether a plan has been completed. The field `"make coffee"` is an optional info field.

With this approach, the programmer can combine multiple intentions that achieve one purpose (in this example drinking coffee). Without the extension for plans, the programmer would have put the intentions on a stack manually, making the management of intentions with one purpose difficult to manage. The structure for plans allows dynamic adding, as well as deleting single or multiple intentions at a time.

This is a significant feature for developing more complex projects, for example MAS student projects. In the following sections, we present our experiences in undergraduate CS education when applying the concepts and the BDI extension introduced so far.

VI. Experiences in CS education

The Artificial Intelligence course at the Berlin School of Economics and Law is part of the curricula in the fifth semester of the Bachelor of Computer Science, carrier accredited and recently re-accredited (for 7 more years) by the AQAS e.V.[5] It has two modules: the first course module being *Autonomous agents and multi-agent systems* (33 hours à 45 minutes) and the second course module being *Knowledge-based systems* (44 hours à 45 minutes). The AI course is an optional course that offers 7 ECTS-credits,[6] and that has a final exam as the primary evaluation form, although students can opt for some credits upon team working in a course project. In such a case, concrete assignments that improve both their written and oral skills are required.

The content is taught in lectures following both a teacher and a student-centered approach. They are organized in six weeklong sections of four hours each. The main subjects that are taught in the first module cover much of those in Wooldridge's book [14] on agents and multi-agent systems.[7] Special attention is given to the BDI model, as well as to both the development and the simulation of practical reasoning agents.

A. The Group

Since the AI course is an optional course, it enrolls only a part of all fifth semester CS students at the BSEL. Most AI classes do not exceed 20 students. Teaching small-sized groups has had positive implications when lecturing, selecting and administering exercises, course projects, as well as on didactic methods to be used in the classroom.

B. NetLogo in the Classroom

NetLogo has already been used as an educational tool in numerous university courses worldwide [15]. We have

[5] German agency for the accreditation of study programs.

[6] European Credit Transfer and Accumulation System. In Germany, one credit point is equivalent to 30 hours of study.

[7] Teaching resources including lecture slides and syllabus descriptions that accompany Wooldridge's book can be found at http://www.cs.ox.ac.uk/people/michael.wooldridge/pubs/imas/resources.html

also used NetLogo in our AI course since 2006. BSEL's CS students learn how to simulate multi-agent scenarios by programming in NetLogo for different undergraduate course projects [16]. Typical multi-agent scenarios are those of rescue agents simulations and predator-prey simulations, for instance. Extensions to NetLogo have been topic of several student research projects at the BSEL as well.

C. The Sandwich Model

For supporting a wide spectrum of class activities and learning opportunities in the first module of the AI course, topics were taught by three different persons in Summer 2012: a faculty professor, a technical assistant, and an undergraduate student in the role of a teaching assistant. In what follows, we refer only to the part taught by the undergraduate student. He is also the developer[8] of the BDI extension already introduced in this article. A differentiated evaluation schema was considered, since the student was also enrolled in the AI course. For doing this, not answering some final exam questions was offered to gain credits against teaching as part of the teaching staff.

Teaching involving the undergraduate student was devoted to the BDI theory, to BDI agents, and to the use of NetLogo to simulate them. All topics were scheduled and discussed with the faculty professor prior to the lectures. They were taught at two lectures following a sandwich model, i.e., by combining passive and active learning units. In his first talk, the student introduced NetLogo as the supporting tool for the course projects to be developed by the rest of the students. He explained NetLogo's main features and where to find what kind of resources. Such a general introduction took about 10 minutes. He then started a practical section of half an hour of duration, where students modeled from scratch, step by a step, and by following his instructions, a predator-prey scenario: almost all of Netlogo's elements and options were introduced and demonstrated based on the Wolf Sheep Predation model from Wilensky [17]. In his second talk, the student followed the sandwich schedule presented in Figure 4 for introducing BDI-related concepts. The duration time in minutes for each activity is given at the left hand side in the figure.

A sandwich structure includes breathing-in and breathing-out phases, which successfully combine theoretical input and practical assignments, as described in [18]. The video that was shown in the beginning opened with a group discussion to motivate both BDI termini and BDI agents. A passive learning section followed, in which beliefs, desires, intentions, and agent commitments towards goals were introduced, as well as a representation of such terms by using Rao and Georgeff's logic [4]. After that, the students had the opportunity to practice the new acquired knowledge by solving exercises specially prepared by their teaching classmate. Solutions were discussed and presented in plenum and were moderated by the teaching student. Then, BDI architectures were introduced briefly, followed by a concrete, practical example in NetLogo using the BDI model. The Wolf Sheep Predation model shown in the first talk a week before, was now being implemented according to the BDI paradigm. The sandwich schedule ended with some advise for implementing BDI in the course projects (both with and without using the new extension to the existent BDI library) before the conclusions of the talk were addressed.

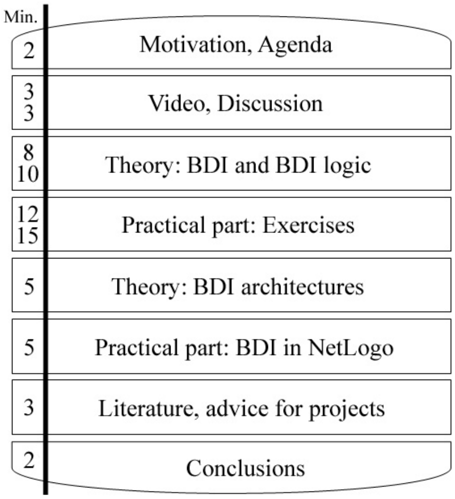

Min.
2
Motivation, Agenda
3
3
Video, Discussion
8
10
Theory: BDI and BDI logic
12
15
Practical part: Exercises
5
Theory: BDI architectures
5
Practical part: BDI in NetLogo
3
Literature, advice for projects
2
Conclusions

Fig. 4. Schedule of a student's talk as a sandwich structure.

VII. Evaluation

The extensions to the BDI library that are presented in Section V were analyzed, designed, implemented, tested and evaluated successfully. Further details are described in the student research project documentation as well and are not addressed here since it would go beyond the scope of this paper (for more information we refer to the original research paper [13]).

Since the project has completed and the results presented, the new library has been part of AI teaching in the classroom. Students can now learn not only the theory concerning BDI but also have the possibility to design and implement MAS in NetLogo, especially when using BDI practically. After following the instructions given by their classmate step by step, they were able to incorporate the BDI approach into their course projects, while gaining more diverse programming skills in NetLogo. The support of the teaching student was highly welcomed in this respect.

Moreover, the students were given at the beginning of the course the possibility to choose either to complete a final exam to evaluate the module *Autonomous agents and multi-agent systems*, or to work on a course project using NetLogo and BDI. After attending the teaching student's tutorials, almost all the groups chose to work in projects, which were all qualitatively high and fully completed by the end of the term.

Last but not least, the extended library has been effectively used in other simulations in NetLogo. Two new undergraduate students currently work in its further evaluation as part of their student research projects. They simulate real world scenarios using FRB and BDI for modeling bank crises and their behavior over time. Their goal is to analyze the BDI library including the extension for efficiency in order to advance it. The project thereby evolves in the classroom, making the students the researches, developers and teacher at the same time. Constant communication between the students who take over the project

[8] At the time of writing this paper, Jonathan Wiens is an undergraduate student attending the fifth semester in the CS career at the BSEL.

and the supervising professor foster the undergraduate research and teaching.

VIII. Conclusions

This paper introduced the subject of an undergraduate student project that focused on extending a BDI library for NetLogo, by conducting a case study in the field of Agent-Based Computational Economics. The extension included a third mental notion to the agents, i.e., the addition of plans in order to allow the clustering of intentions and other plans for the modeling of agent's actions that consist of other actions, as well as the redefinition of intentions to allow calls by reference. Furthermore, this work presented how such undergraduate research supports AI classroom teaching at the Berlin School of Economics and Law. The idea to combine student research and student teaching has received a positive echo from all participating parties – the school, faculty and students.

Beyond the basics of mental notions, the BDI paradigm does not have fixed specifications dictating what the library should offer. Future work includes analyzing useful features and extending the planning of agents by adding dynamic prioritizing. Dynamic prioritizing would enable agents both to have and two swap competing plans at run time.

References

[1] U. Wilensky, "NetLogo," Evanston, IL, U.S.A., 1999, available online at http://ccl.northwestern.edu/netlogo/, retrieved January 3, 2013.

[2] I. Sakellariou, P. Kefalas, and I. Stamatopoulou, "Enhancing NetLogo to Simulate BDI Communicating Agents," in *Proceedings of the* 5^{th} *Hellenic Conference on Artificial Intelligence, SETN'08*, ser. Lecture Notes in Artificial Intelligence (LNAI), J. Darzentas, G. Vouros, S. Vosinakis, and A. Arnellos, Eds., vol. 5138. Syros, Greece: Springer Berlin Heidelberg, October 2008, pp. 263–275.

[3] M. E. Bratman, *Intention, Plans, and Practical Reason*. Cambridge, MA, U.S.A.: Harvard University Press, 1987.

[4] A. Rao and M. Georgeff, "Modeling Rational Agents within a BDI-Architecture," in *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning, KR'91*, J. Allen, R. Fikes, and E. Sandewall, Eds. Morgan Kaufmann, April 1991, pp. 473–484.

[5] ——, "BDI Agents: From Theory to Practice," in *Proceedings of the First International Conference on Multi-Agent Systems, ICMAS'95*. San Francisco, CA, U.S.A.: AAAI Press / The MIT Press, June 12–14 1995, pp. 312–319.

[6] V. Mascardi, D. Demergasso, and D. Ancona, "Languages for Programming BDI-style Agents: an Overview," in *WOA*. Pitagora Editrice Bologna, 2005, pp. 9–15.

[7] L. Tesfatsion, "Agent-based Computational Economics: Modeling Economies as Complex Adaptive Systems," *Information Sciences*, vol. 149, pp. 263–269, 2003.

[8] L. Tesfatsion and K. Judd, Eds., *Handbook of Computational Economics: Agent-Based Computational Economics*, 1st ed. Elsevier, 2006, vol. 2.

[9] J. Arthur, W. Holl, B. Lebaron, R. Palmer, and P. Tayler, *Asset pricing under endogenous expectations in an artificial stock market*. Addison-Wesley, 1997, pp. 15–44.

[10] T. Klos and B. Nooteboom, "Agent-based computational transaction cost economics," *Journal of Economic Dynamics and Control*, vol. 25, no. 3-4, pp. 503–526, March 2001.

[11] M. Oeffner, "Agent-Based Keynesian Macroeconomics – An Evolutionary Model Embedded in an Agent-Based Computer Simulation," September 2008.

[12] M. Rothbard, *The Mistery of Banking*, Second ed. Ludwig von Mises Institute, 2008.

[13] J. Wiens, "Extending a NetLogo library for BDI-architectures in multi-agent systems," August 2012, computer Science Dept., Faculty of Cooperative Studies, Berlin School of Economics and Law, Student research paper, unpublished.

[14] M. Wooldridge, *An Introduction to MultiAgent Systems*, Second ed. UK: John Wiley & Sons Ltd, May 2009.

[15] "NetLogo Courses," Evanston, IL, U.S.A., n.d., available online at http://ccl.northwestern.edu/courses.shtml, retrieved January 3, 2013.

[16] D. Monett, R. Janisch, and S. Starroske, "NL-Analyzer: Enhancing Simulation Tools to Assist Multiagent Systems' Teaching," in *Proceedings of the Workshop Multi-Agent Systems for Education and Interactive Entertainment, MASEIE'2010*, W. van der Hoek, G. A. Kaminka, Y. Lesperance, M. Luck, and S. Sen, Eds., Toronto, Canada, May 10–14 2010, pp. 1–6.

[17] U. Wilensky, "Netlogo Wolf Sheep Predation model," Evanston, IL, U.S.A., 1997, available online at http://ccl.northwestern.edu/ netlogo/-models/WolfSheepPredation, retrieved January 3, 2013.

[18] D. Monett and M. Sänger, "Research and Teaching with Remo: Student research projects and teaching for and by undergraduate students," in *Proceedings of The 2012 International Conference on Frontiers in Education: Computer Science and Computer Engineering, FECS'2012*, H. Arabnia, V. Clincy, and L. Deligiannidis, Eds., vol. 2. Las Vegas, NV, U.S.A.: CSREA Press, July 2012, pp. 353–359.

A Problem Based Approach to Teaching Programming

W. Pullan[1], S. Drew[2], and S. Tucker[1]
[1]School of ICT, Griffith University, Gold Coast, Queensland, Australia
[2]Griffith Institute of Higher Education, Griffith University, Gold Coast, Queensland, Australia

Abstract - *Java Programming Laboratory (JPL) is a cloud based learning environment used for teaching object-oriented programming at Griffith University, Australia. JPL incorporates a number of features found in other successful programming learning environments and builds upon them with a range of innovative features. JPL provides a database that tracks individual students' successes and progression through scaffolded programming exercises and assessment items and gives students immediate feedback on their use of programming language syntax and correctness of problem solutions. A data querying and visualisation facility allows analysis of the database to provide real-time performance indicators from the overall course / problem level down to the individual student / specific problem level. Programming instructors and curriculum designers will find that this facility allows a responsive approach to student engagement, assistance and progression; as well as course problem tuning in a just-in-time manner.*

Keywords: *Object-oriented Programming; Problem-Based Learning; Real-Time Progress Tracking; Scaffolded Development;*

1 Introduction[1]

Learning object-oriented programming is a difficult process for most first-year students but, once mastered, it is transformative in that it provides a means to represent and solve a range of computing related problems in many application areas. Compounding the difficulty in learning programming is the fact that it is normally taught, in first year, to a cohort of students with a considerable range of academic abilities and experiences. During the pre-2000 dot.com boom, by far the majority of ICT students at the authors' university were high achievers who had studied relevant preparatory courses at high school and were motivated by the prospect of a relatively highly paid career in an industry where they were in high demand. More recent moves towards massification of higher education (Altbach, Reisberg and Rumbley, 2009) and a local reduction in popularity of many ICT related professions has resulted in the lowering of entry requirements for ICT programs to maintain budget quotas. While these moves have increased the difficulty of teaching programming, it should be noted that, even in the past, teaching introductory programming was challenging and often resulted in student avoidance of further programming based courses. What is reported in this paper are the results of a project that addresses these challenges by implementing a successful learning and teaching system based on engagement theory (Kearsley and Schneiderman, 1998) and constructivist design philosophy (Jonassen, 2003; Karagiorgi and Symeou, 2005).

In the professional setting, where most academics are untrained in educational theory, a simple and effective approach is provided by having a focus on "active learning" strategies. Biggs (1999) strata of conceptions of teaching progress from concentration on student as recipient of the teacher's wisdom, or *what the student is*, to a concentration on executing the act of teaching, or *what the teacher does*, and ultimately a focus on *what the student does* in the quest for knowledge and eventual understanding. It is this final conceptual level of teaching that leads to a focus on active learning (Jonassen and Rohrer-Murphy, 1999; Scanlon et al., 2002; Tetard and Patokorpi, 2005; Baugh, 2009) and student learning through experience (Kolb, 1984; Boud, Cohen and Walker, 1993). Active learning, in the context of teaching introductory programming, where a student is to learn a programming language and the tools of programming they must write programs so task setting is not difficult. However, the best effect will be obtained in an environment that facilitates and motivates them to write programs and, with a wide range in student ability, this is where the challenge arises - how to keep the poorer students progressing while not causing the more able students to lose interest.

A programming learning environment, Java Programming Laboratory (JPL) was developed to address the design criterion of providing scaled levels of challenge to maintain student engagement with learning for a cohort of varied ability. An active learning approach allows teaching staff to concentrate on *what the student does* on their journey to learning programming in an immersive software development environment. There are a number of key design features of the JPL learning environment to address relevant instructional design theory. *Firstly*, it presents a component approach (Pintrich and de Groot, 1990; Biggs, 1999b) that is as 'concrete' conceptually as possible with students having to focus only on the topic currently being taught. *Secondly*, it presents a problem-based learning approach (Savery and Duffy, 1995) and provides a large range and number of problems, selected from an even larger range of problems available for each topic. Problems are designed to scaffold development (Simons and Klein, 2007) to cater for the wide range of student abilities, providing multiple entry points into

[1] This paper uses the naming convention program and courses to refer to a degree structure where a program is a set of courses. This is in contrast to the course / (unit/subject) naming convention.

each topic, and additional problems for better students. Thirdly, it promotes responsive, formative feedback (Nicol and Macfarlane-Dick, 2006), with continuous access to personal and class (group) performance measures, and access to just-in-time assistance. Fourthly, it provides a consistent, simple environment available for any computing platform; and flexible access to the learning environment that is available in university computer laboratories or off campus via the Internet. Fifthly, as a client-based application it ensures students can maintain learning momentum and time-on-task even while the Internet may not be accessible. Finally, to maintain originality of student learning it does not allow solutions to problems to be passed from year to year which mitigates the learning process.

JPL incorporates a large number of features, some of which are based on existing open-source programming teaching systems. These include the Stanford University (Stanford, 2011) teaching initiative titled *CodingBat* that has been used to teach Java programming at that institution, *VideoNotes* (Cornell, 2011) and interactive web-pages (Kjell, 2006). Using the lessons learned from these teaching systems, this paper now describes the JPL approach that has been taken to dealing with the issues highlighted above in teaching introductory programming.

2 Java Programming Laboratory

Java Programming Laboratory is an educational system designed to assist students learning Java as their first programming language. Basically, JPL provides a software-based environment, available on both the universities and students computers, that allows students to develop their programming skills by starting with simple, targeted, Java code fragments and slowly transitioning to complete Java programs. The fundamental concept underlying JPL is – "*you can learn programming skills by writing many, small, targeted code fragments*". Within JPL, learning a programming language is devolved into smaller steps of learning and practicing computer-based problem solving techniques. This simultaneously aids learning of programming language constructs, their syntax and semantics.

The overall system structure of JPL is shown in Figure 1 and the components of this system are described in the following bullet points / sub-sections.

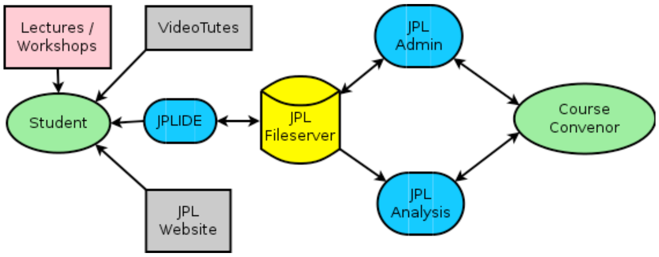


Figure 1 : JPL System Overview

- Lecture / Workshops : The lectures / workshops are very much a "*teach by example, learn by doing*" exercise in that the lectures are a 50/50 combination of topic discussions and joint (lecturer / students) JPL problem solving and implementation. The on-campus workshops are solely based on the student working in JPL with tutorial assistance to solve a specific issue directly available if required.
- JPL VideoTutes : An integral part of JPL is the use of short (~10 minutes) video tutorials explaining key programming concepts and problem solving techniques. In effect the student is able to 'look over the shoulder' at the computer screen as an experienced programmer demonstrates both programming language features and computer based problem solving from problem analysis to program implementation. Shorter versions of video-tutes are also used to provide hints to solve problems. These video micro-tutorials that support student learning of individual constructs provide an indexed system of instruction and reviewable examples to support incremental development of problem solving and programming knowledge through guided practice.
- JPL Website : The JPL website contains a number of interactive learning tools, some of which were originally created by Bradley Kjell, Central Connecticut State University (Kjell, 2006). For the object-oriented programming course in Java, these include multiple choice quizzes, each with correct answers supplied on completion, fill-in-the-blank review sessions and multi-page interactive topic discussions. The JPL Website is available at www.ict.griffith.edu.au/JPL.
- JPL Fileserver : The JPL Fileserver acts as the central repository for all JPL related files. When JPLIDE is initiated, if needed and the Internet is available, a merge of the local copy of the student's log and the student's log on the JPL Fileserver is performed and a new copy of the JPL Course Control File downloaded from the JPL Fileserver. As the student performs JPL activities, both copies of the student's log are kept up to date. If the Internet is not available, the student is still able to use JPLIDE and the logs will be merged when the Internet does become available.

2.1 JPL Integrated Development Environment (JPLIDE)

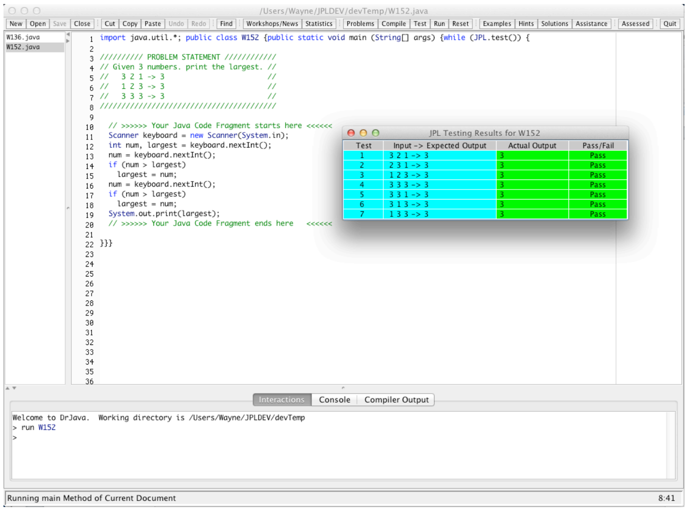
Students perform all programming activities completely within the JPLIDE, which is based on the DrJava open source IDE and has the user interface shown in Figure 2. To choose a problem, students access the Problems item on the Toolbar and this produces the JPL Problem Selector display shown in Figure 3. The problem selector contains a table of problem identifiers for the course, where the current status of the problem (template obtained, compiled, failed test, successfully completed (Green)) for the student is colour coded as background to the problem identifier. The leading character of the problem identifier gives the problem type (W -> Workshop problem, H -> Homework problem). Students start at the bottom row of the problem selector table and work their way up the table (which allows for the situation where problem rows are released as the semester progresses) with problem difficulty increasing from left to right.

For straight-forward problems, such as that shown in Figure 2, the complete problem specification is stored in the problem template and the Java code after the Problem Statement is not present in the initial template. For more difficult object oriented problems, the problem specification, including items such as UML diagrams, are stored in a PDF file that is automatically downloaded when the associated problem is selected. In addition, all source files for supplied classes for a more advanced problem will be automatically downloaded directly into JPLIDE and any input disk files required will also automatically downloaded on the first execution of the student's Java program. This automatic downloading of everything that is required for a problem, allows the student to focus solely on developing a class which demonstrates a particular object oriented concept (e.g. inheritance, polymorphism).

Once a problem has been selected, the student will start creating code and use JPL automated testing to check for correct logic. Figure 2 shows the completed program and the response to a successful execution for a simple, non-object oriented problem.

A brief summary of the JPL related commands available on the JPLIDE ToolBar is:

- News/Workshops : Display current course news and expected workshop activity for each week of the course.
- Statistics : Provides information on how well a student is performing with regard to the rest of the class and a weekly / overall breakdown of the JPL tasks they have performed. In particular, this shows the student their JPL Performance Indicator and how it compares to the performance indicators for the complete class (Figure 4). This data is available immediately after the first workshop in Week 1 of semester.
- Problems : Presents the JPL Problem Selector (Figure 3) that displays all the problem identifiers, the current status of the problems for this student and allows a problem to be selected.
- Compile : Compiles the current Java program and saves a copy of the source file to the JPL Fileserver.
- Test : Using the current Java program, performs the JPL automated testing which uses test data and expected outputs stored in the JPL Course Control File.
- Run : Executes the current Java program outside the JPL testing environment (i.e. with manual input and normal output).
- Stop : DrJava command which terminates the currently executing Java program. Primarily used to terminate an infinite program loop.
- Examples : This allows students access to many Java code fragments to perform specific tasks and also to complete Java programs that they can compile and execute.
- Hints : These include a solution flowchart available as a hint; solution pseudo-code available as a hint; and hint suggesting Java methods to use – e.g. for a String problem, which String methods (of the 43 available) is best to use in the solution.



```
import java.util.*; public class W152 {public static void main (String[] args) {while (JPL.test()) {

////////// PROBLEM STATEMENT ////////////
// Given 3 numbers. print the largest. //
//   3 2 1 -> 3                       //
//   1 2 3 -> 3                       //
//   3 3 3 -> 3                       //
/////////////////////////////////////////

  // >>>>>> Your Java Code Fragment starts here <<<<<<
  Scanner keyboard = new Scanner(System.in);
  int num, largest = keyboard.nextInt();
  num = keyboard.nextInt();
  if (num > largest)
    largest = num;
  num = keyboard.nextInt();
  if (num > largest)
    largest = num;
  System.out.print(largest);
  // >>>>>> Your Java Code Fragment ends here   <<<<<<

}}}
```

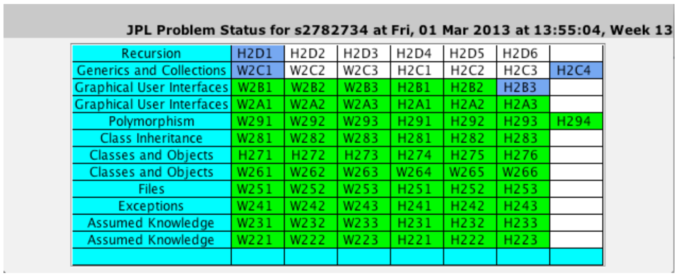
JPL Testing Results for W152

Test	Input -> Expected Output	Actual Output	Pass/Fail
1	3 2 1 -> 3	3	Pass
2	2 3 1 -> 3	3	Pass
3	1 2 3 -> 3	3	Pass
4	3 3 3 -> 3	3	Pass
5	3 3 1 -> 3	3	Pass
6	3 1 3 -> 3	3	Pass
7	1 3 3 -> 3	3	Pass

```
Welcome to DrJava.  Working directory is /Users/Wayne/JPLDEV/devTemp
> run W152
>
```

Running main Method of Current Document 8:41

Figure 2 : JPLIDE User Interface and Automated Testing Result



JPL Problem Status for s2782734 at Fri, 01 Mar 2013 at 13:55:04, Week 13

Recursion	H2D1	H2D2	H2D3	H2D4	H2D5	H2D6	
Generics and Collections	W2C1	W2C2	W2C3	H2C1	H2C2	H2C3	H2C4
Graphical User Interfaces	W2B1	W2B2	W2B3	H2B1	H2B2	H2B3	
Graphical User Interfaces	W2A1	W2A2	W2A3	H2A1	H2A2	H2A3	
Polymorphism	W291	W292	W293	H291	H292	H293	H294
Class Inheritance	W281	W282	W283	H281	H282	H283	
Classes and Objects	H271	H272	H273	H274	H275	H276	
Classes and Objects	W261	W262	W263	W264	W265	W266	
Files	W251	W252	W253	H251	H252	H253	
Exceptions	W241	W242	W243	H241	H242	H243	
Assumed Knowledge	W231	W232	W233	H231	H232	H233	
Assumed Knowledge	W221	W222	W223	H221	H222	H223	

Figure 3 : JPLIDE Problem Selector

- Solutions : Provides, via a problem selector, solutions to selected problems. These are released after the relevant topic has been covered and students are unable to access a solution until they have made a reasonable effort at solving the problem.
- Assistance : Allows a student to request assistance with a problem. This results in an automatic email being sent to the course convenor who, working in Admin mode, is able to take over the student's JPL environment, make code corrections or just insert suggestions into the java source files. Once help has been given, the course convenor returns to their own JPL environment and an email is automatically generated and sent to the student.
- Assessed : Causes JPLIDE to enter Assessed Mode so that an Assessed Workshop can be performed. In Assessed Mode, JPLIDE is 'locked down' and the only file that can be opened is the Assessed Problem that is not normally assessable. In addition, a number of other

JPLIDE commands (e.g. Paste) are disabled. Assessed mode can only be entered as the first command after JPLIDE is started and, while JPLIDE is in Assessed Mode, all other executing programs on the computer are monitored to detect disallowed programs.

- Quit : Terminates JPLIDE after updating the JPL Fileserver (if the Internet is available).

Less commonly used JPLIDE commands that are available via a menu items include: Last Test Result, Student History, Restart Problem, Change Course, Java Tutorials, JPL Videos, JPL Website, and Administration items such as those described in the next sub-section.

2.2 JPL Course Administrator (JPLAdmin)

The JPLAdmin interface allows the course convener to create the workshop contents and also update the News/Workshop display. For the course convener only, the menu items available in JPLIDE under the Admin menu item include:

- Assist Student : In response to a student request for assistance, the course convenor is able to directly assist the student using the mechanism described for menu item Request Assistance above.
- Workshop Problems : This allows the course convenor to scan a library of problem templates (over 500 are available) for possible inclusion in a workshop. Problems are grouped by topic and difficulty ranking. There is a simple Course Definition text file that is updated to include a problem in a workshop.
- Build Workshop : This command takes the Course Definition text file edited above and creates the JPL Course Control File.
- Update Server : Loads the current JPL Course Control File up to the JPL Fileserver ready for automatic download by JPLIDE.

2.3 JPL Course and Student Analysis (JPLAnalysis)

JPLAnalysis is a stand-alone program that performs analysis of all student logs held on the JPL Fileserver. Statistics can be obtained at the problem, student, course level and any combination of these through a command line interface.

2.3.1 Student Level Reports

As an example of tracking an individual student, Figure 3 and Figure 4 show some of the information available at the student level. From Figure 3 it can be seen that this student, from the 74 JPL problems available in this course, successfully completed 60 problems, attempted 4 more and did not attempt 10 problems. The first graph in Figure 4 shows how a students JPL Performance Indicator (horizontal blue line) compares with the performance indicators for all the class (red columns). This graph is constantly available to students from Week 1 of semester and gives them constant feedback on their performance in relation to other students in the course. Obviously this student is performing very well however the second graph shows the students work pattern and clearly they were able to work ahead during the first portion of the semester and then able to make less effort during the latter part of semester.

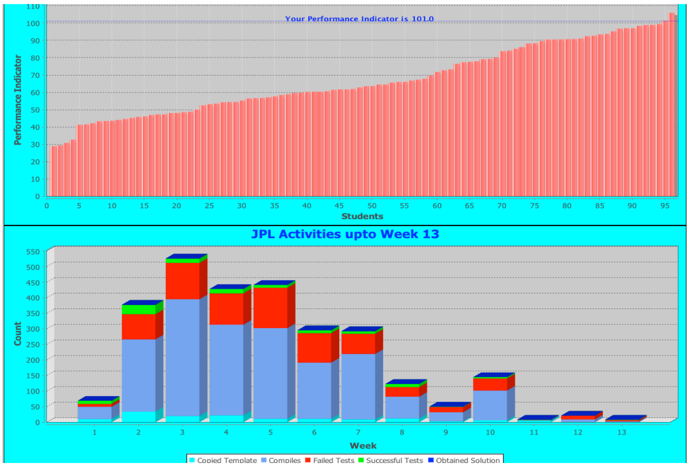


Figure 4 : JPL Performance Indicator and activity for a particular student

Tracking individual student activity within JPL has a number of uses in the university context. These range from identifying points in the course where students 'switch-off' to evaluating the amount of effort a student has put into producing an assignment so as to flag possible plagiarism cases.

2.3.2 Course Level Reports

An example of a course level report is shown in Figure 5 which shows the total student JPL activity on a week by week basis for the course.

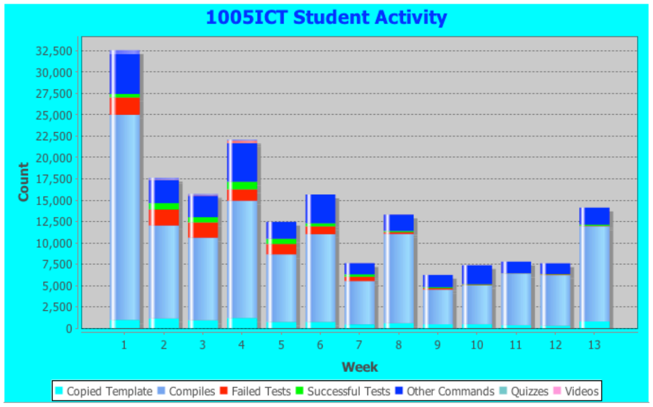


Figure 5 : Total student activity for each week of the semester

The slow decline in JPLIDE commands during semester is mostly due to the fact that the JPL problems became more difficult as the semester progressed so the rate at which students were inputting commands decreased.

2.4 JPL Course Results

The topics covered in this course are shown on the left-hand side of the JPL Problem Selector in Figure 3 and a summary of student activity is given in Table 1.

JPL Activity	Semester Total
JPLIDE Starts	8,065
Total JPLIDE Commands	189,805
Open a new JPL Template	7,794
Compiles	123,734
Failed Tests	10,239
Passed Tests	4,495
Total JPLIDE Dev. Commands	157,105
Home activity as a % of total activity	62%

Table 1 : JPLIDE Usage Counts for the Object Oriented Programming course with around 100 Students

A frequency analysis of constructs derived from student survey responses under the headings of "*what worked well*" and "*what needs improvement*" in the JPL based course revealed constructive information. Most popularly, JPL was perceived as helpful to students learning programming and that they appreciated the learning environment as being easy to use. Next most popular responses relate to the instructor facilitated laboratory learning experience based around practical exercises. Here students appreciated the quality of the tutorial assistance, the tutorial/laboratory learning experience, and the level and access to help that they found most effective. Frequent mentions were made of the flexible access to JPL from outside of laboratory classes and the appreciation of the lecturer and lecture/workshop classes. Interestingly, students found that JPL assisted them in understanding the curriculum and course structure and that the curriculum design assisted their learning. Several other lower frequency constructs were mentioned relating to various JPL and course design features.

In the area of what students felt needed most improvement there was a wide range of issues of low frequency suggesting a range of mainly individual learning needs that could be addressed. For most students a lack of understanding or experience of curriculum design for learning programming may limit the number of ideas they can offer for course improvement. Many, above, experienced an effective learning experience and would not seek to make changes. Indeed, the most frequent response in the survey was that they could think of no improvements. Next most frequent responses were from the percentage of the class having had no experience of learning to program. Some of these students suggested more help, and a softer introduction to the course, technical language, and requirements for undertaking programming. Relating to this there were requests for more tutorial assistance, more tutorials, and making the course less difficult. In direct contrast, and indicating the mixed levels of ability and experience amongst students, there were also requests for more problems with higher levels of difficulty and challenge. From the broad range of responses it appears that JPL and the course design managed to succeed in meeting the needs of such a diverse class.

3 Conclusions

JPL is based on the premise '*to learn programming you have to do programming*' and places students in an environment where they can totally focus on that. An observation from the current semester is that students are motivated by the JPL testing feedback they receive and are also motivated by the JPL Performance Indicator and the colour indicators on the JPL Problem Selector panel. These factors enhance student engagement with learning to program.

Within JPL, learning a programming language is devolved into smaller steps of learning and practicing computer based problem solving techniques. This simultaneously aids learning of programming language constructs, their syntax and semantics. Problems available in JPL are designed to scaffold student learning through a number of stages. These problems range from modifying existing programs, 'fill in the blanks' type exercises, developing Java code fragments and finally developing full Java programs. All student work in JPL is automatically tested and, as each student performs work on an exercise, the event is registered by the system. At a glance, at any point in the semester, the convenor or tutor can see how each student is performing and investigate any potential problems immediately.

Flexibility and continuity of access is important for our cohorts so the complete JPL system is available to students both in the ICT computer laboratories and also on their home computers. JPL has been used in a number of programming courses including high schools, an Australian university and a Chinese university. Within the university context, JPL seeks to address student retention and poor learning outcomes by improving scaffolding and learning support for key ICT programming courses. For the flagship ICT undergraduate programs, the courses that involve learning programming languages have been identified as containing key threshold concept areas. In all major strands of these degrees, learning to program is normally compulsory and starts from first semester, first year. Computer programming is considered a difficult learning area and can have a high student failure rate. Building upon a successful international blended-learning model, JPL provides an integrated program development environment which also includes automated testing and a comprehensive set of construct level, video-tute resources to aid computer program development and self-paced learning. This online program development and problem-based experiential learning environment enables academics and students to monitor progression through automatically evaluated learning objectives. JPL instructor access to students' achievements on tutorial and assessment tasks allows earliest possible identification of students who are at risk of failing in order to provide timely remedial assistance. JPL also provides feedback to the academic/teaching team designing problem sets and curricula to identify where extra learning assistance or redesign is required.

Outcomes of the JPL approach to teaching introductory programming have thus far been very encouraging in terms of impacting positively on student learning experience and learning outcomes.

4 References

[1] Altbach, P. G., Reisberg, L., & Rumbley, L. E. (2009). Trends in global higher education: Tracking an academic revolution. Center for International Higher Education.

[2] Baugh, J. M. (2009). Let's Have Fun with That Required Computer Information Systems Introduction Course. Information Systems Education Journal, 7(73).

[3] Biggs, J. B., (1999a). Teaching for Quality Learning at University, Open University Press / Society for Research into Higher Education.

[4] Biggs, J. B. (1999b). What the Student Does: teaching for enhanced learning. Higher Education Research & Development, 18(1), 57 - 75.

[5] Boud, D., Cohen, R., & Walker, D. (1993). Using experience for learning. Buckingham [England]; Bristol, PA: Society for Research into Higher Education and Open University Press.

[6] Cornell University, (2011). VideoNote, from http://www.videonote.com/study.aspx

[7] Jonassen, D. H., & Rohrer-Murphy, L. (1999). Activity theory as a framework for designing constructivist learning environments. Educational Technology Research and Development, 47(1), 61-79.

[8] Jonassen, D. H. (2003). Learning to solve problems with technology: a constructivist perspective (2nd ed.). Upper Saddle River, N.J.: Merrill.

[9] Karagiorgi, Y., & Symeou, L. (2005). Translating Constructivism into Instructional Design: Potential and Limitations. Journal of Educational Technology & Society, 8(1), 17-27.

[10] Kearsley, G., & Schneiderman, B. (1998). Engagement Theory: A Framework for Technology-Based Teaching and Learning. Educational Technology, 38(5), 20-23.

[11] Kjell, Bradley , (2006), Introduction to Computer Science Using Java, Central Connecticut State University http://chortle.ccsu.edu/CS151/cs151java.html

[12] Kolb, D. A. (1984). Experiential learning : experience as the source of learning and development. Englewood Cliffs, N.J.: Prentice-Hall.

[13] Nicol, D., & Macfarlane-Dick, D. (2006). Rethinking Formative Assessment in HE: a theoretical model and seven principles of good feedback practice. Studies in Higher Education, 31(2), 199-218.

[14] Pintrich, P. R., & de Groot, E. V. (1990). Motivational and self-regulated learning components of classroom academic performance. Journal of Educational Psychology, 82(1), 33-40.

[15] Savery, J. R., & Duffy, T. M. (1995). Problem-based learning: An instructional model and its constructivist framework. In B. G. Wilson (Ed.), Constructivist learning environments: Case studies in instructional design (pp. 135-148). Englewood Cliffs, NJ: Educational Technology Publications.

[16] Scanlon, E., Morris, E., diPaolo, T., & Cooper, M. (2002). Contemporary approaches to learning science: technologically-mediated practical work. Studies in Science Education, 38(1), 73 - 114.

[17] Simons, K. D., & Klein, J. D. (2007). The Impact of Scaffolding and Student Achievement Levels in Problem-based Learning Environment. Instructional Science, 35(1), 41-72.

[18] Stanford University, (2011), CodingBat, from http://www.codingbat.com

[19] Tetard, F., & Patokorpi, E. (2005). A Constructivist Approach to Information Systems Teaching: A Case Study on a Design Course for Advanced-Level University Students. Journal of Information Systems Education, 16(2), 167-176.

Computing Education and Research for High School Students through Subject Modularization and Basic Software

Jiho Kim[1], Hwamog Kim[2], and Seongjai Kim[3]
[1]301-1301 Koaroo, Gak-ri, Ochang-eup, Cheongwon-gun
Chungcheongbuk-do, S. Korea Email: jiho6617@naver.com
[2]Department of Mathematics and Statistics, Mississippi State University
Mississippi State, MS 39762 USA Email: hk404@msstate.edu
[3]Department of Mathematics and Statistics, Mississippi State University
Mississippi State, MS 39762 USA Email: skim@math.msstate.edu (Contact Author)

Abstract—*It is interesting to construct a research environment which allows high school (HS) students to make significant contributions to interdisciplinary research. A project is launched by introducing two new denoising models as variants of the Bregman model, for three HS students. To help them learn the denoising models and C++ effectively, the mathematical subjects are partitioned into easy-to-solve small modules (subject modularization), and two sets of basic software are implemented for the Bregman model: one in Matlab and the other in C++. The students are requested to implement a code for the new models by adding functions in Matlab first and then in C++. All the students could finish given assignments successively. It has been found that HS students can learn C++ more easily when they begin with an easy language and that they can make significant research contributions when the subjects are modularized appropriately and a basic software is provided.*

Keywords: Computing education, image denoising, basic software, subject modularization.

1. Introduction

Enrollment in engineering continues to be lower than the demand. Outreach activities, active recruiting, and retention efforts can help reverse this trend [1]. While this situation generally applies to all fields of science, technology, engineering, and mathematics (STEM), this article studies an effective way of teaching and learning high school (HS) computing. In particular, we are interested in the construction of a research environment which allows HS students to make fundamental contributions to interdisciplinary research in computer sciences and applied mathematics, by implementing a sequence of functions in C++.

It is difficult for HS students to learn sophisticated computational languages such as C++ in a short time. Furthermore, it is hardly possible for them to make a research collaboration with researchers or practitioners concerning real-world issues. It is particularly true when research subjects are either interdisciplinary or publishable to an academic journal or conference proceedings. It is mainly due to the fact that most research activities require large hours of computer implementation and a deep understanding over at least one area of mathematics, computer sciences, and engineering; which is beyond the scope of knowledge obtainable by normal HS students. Thus a challenging task in HS computing is to develop strategies and construct an effective research environment for the students to be able to understand the research objective thoroughly, share the work in research collaboration, and make important contributions.

In this article, we present an effective mentoring strategy and its results, in the area of mathematical image denoising, implemented in Matlab/Octave and C++ and carried out by three HS students. Image denoising is one of oldest problems in image processing; it is important in various image restoration tasks (medical imaging and remote sensing) and is often necessary as a pre-processing for other image processing tasks. However, most denoising algorithms introduce certain artifacts to the resulting images, such as staircase effects and image blur. As a research project and also as a case study for HS computing education, we develop new partial differential equation (PDE)-based denoising models to minimize the drawbacks in image denoising.

Our educational objective is to construct a suitable environment for HS students to be able to make fundamental contributions to such an interdisciplinary research. We are aware of learning theories such as *information processing* [2], [3] and *social constructivism* [4], [5], [6]. We believe that learning occurs most effectively through information processing and social interaction between a more experienced and competent individual (i.e., a mentor) and a less competent individual (i.e., a student). Such interaction is the conduit through which knowledge is constructed, to help the learner internalize specific concepts.

The proposed research, presented in detail in Sections 2 and 3, requires both strategic development and large hours of computational experiments. Thus the problem appears too difficult and complicated for HS students to be involved in the research. However, we introduce a new strategy of arranging solution procedures for the students to achieve a great success, called *subject modularization* (SM). We first partition appropriately both mathematical subjects and the corresponding implementation tasks into small *modules*, each of which is manageable for the students. Then we provide the students with a *basic software*, composed in an easy computational language such as Matlab or Octave, to promote their success with computer implementation. A basic software is a computer code, often incomplete or previously used for similar research subjects, on which one can add or modify modules for various applications of interests.

Composing of an effective basic software along with an appropriate SM is crucially important for the mentors to provide a balance of support so that the students can have a reasonable share of the work and derive a maximum pleasure from the research experience. With an appropriate support from mentors, the students can successfully accomplish the given research project by resolving and implementing a series of small easy-to-solve modules.

In the beginning stage of the current project, the students show difficulties in understanding mathematical backgrounds and physical implications for most of the modules. However, after being experienced with a basic software written in Matlab/Octave which carries out the Bregman iterative refinement for PDE-based image denoising [7], [8], the students can understand more thoroughly what they are doing. Furthermore, they have been able to modify given functions or implement new functions for two similarly-formulated new denoising models. The new models turn out to be more controllable or more effective than the Bregman refinement.

As the main computing task, the students are given a basic software written in C++, simulating the Bregman refinement in the same way as in the Matlab basic software. The students are first instructed for basics of C++ and the C++ basic software for the Bregman refinement. Then they are requested to implement/modify functions in the basic software for the new denoising models, with a help from one-hour research discussions arranged through tele-communication twice a month. For the new task, all three students could finish their implementation successfully in three months.

The article is organized as follows. The following section begins with a brief review of conventional PDE-based denoising models and the Bregman iterative refinement method, followed by new formulations of the iterative refinement. In Section 3, we select three denoising models of which two are new; pedagogical processes for the simulation of the new models are presented in detail. Section 4 shows research results. Our experiments on the research and HS computing education are summarized in Section 5.

2. Preliminaries

This section presents PDE-based denoising algorithms as preliminaries for computing education for HS students. After a brief review of conventional PDE-based denoising models, the Bregman iterative refinement method is discussed along with its variants. Detailed educational experiments for HS computing are presented in Section 3.

2.1 Conventional PDE-based denoising models

Given an observed (noisy) image $f : \Omega \to \mathbb{R}$, where Ω is the image domain which is an open subset in $\mathbb{R}^2$, we consider the noise model of the form

$$f = u + \eta, \tag{1}$$

where u is the desired image and η denotes the noise of a zero mean. Then a common denoising technique is to minimize a functional of gradient:

$$u = \arg\min_u \Big\{ \int_\Omega \rho(|\nabla u|)\, d\mathbf{x} + \frac{\lambda}{2} \int_\Omega (f-u)^2\, d\mathbf{x} \Big\}, \tag{2}$$

where ρ is an increasing function (often, convex) and $\lambda \geq 0$ denotes the constraint parameter. It is often convenient to transform the minimization problem (2) into a differential equation, called the *Euler-Lagrange equation*, by applying the variational calculus [9]:

$$-\nabla\cdot\Big(\rho'(|\nabla u|)\frac{\nabla u}{|\nabla u|}\Big) = \lambda\,(f-u). \tag{3}$$

When $\rho(s) = s$, the model (3) becomes the total variation (TV) model [10]:

$$\kappa(u) \,=\, \lambda(f-u), \qquad \text{(TV)} \tag{4}$$

where $\kappa(u)$ is the negation of the *mean curvature* defined as

$$\kappa(u) = -\nabla\cdot\Big(\frac{\nabla u}{|\nabla u|}\Big). \tag{5}$$

It is often the case that the constraint parameter λ is set as a constant, as suggested by Rudin-Osher-Fatemi [10]. In order to find the parameter, the authors merely multiplied (4) by $(f-u)$ and averaged the resulting equation over the whole image domain Ω:

$$\lambda = \frac{1}{\sigma^2}\frac{1}{|\Omega|}\int_\Omega (f-u)\,\kappa(u)\,d\mathbf{x}, \tag{6}$$

where σ^2 is the noise variance

$$\sigma^2 = \frac{1}{|\Omega|}\int_\Omega (f-u)^2\,d\mathbf{x}. \tag{7}$$

The TV model tends to converge to a piecewise constant image. Such a phenomenon is called the *staircasing* effect. In order to suppress it, Marquina and Osher [11] suggested to multiply the stationary TV model by a factor of $|\nabla u|$:

$$|\nabla u|\,\kappa(u) \,=\, \lambda\,|\nabla u|\,(f-u). \qquad \text{(ITV)} \tag{8}$$

Since $|\nabla u|$ vanishes only on flat regions, its steady state is analytically the same as that of the TV model (4). We will call (8) the *improved TV* (ITV) model, as called in [12]. Such a non-variational reformulation turns out to reduce the staircasing effect successfully; however, it is yet to be improved for a better preservation of fine structures.

The conventional PDE-based denoising models, including ones presented in this section, can be written in the following general form

$$\mathcal{L}(u) = \mathcal{C}(f - u), \tag{9}$$

where $\mathcal{L}$ is a diffusion (smoothing) operator and $\mathcal{C}$ denotes the constraint parameter.

2.2 The Bregman iterative refinement

In order to recover fine structures in the image, iterative refinement procedures employing an original idea by Bregman [7] have been introduced in image restoration and image zooming [8], [13]. The Bregman iterative refinement applied to the general denoising model (9) reads as follows: if u_1 is the solution of (9)

$$\mathcal{L}(u_1) = \mathcal{C}(f - u_1), \tag{10}$$

we denote the corresponding residual by r_1, i.e.,

$$r_1 = f - u_1.$$

Then we again solve the TV model with the signal replaced by $f + r_1$; the solution u_2 will satisfy

$$\mathcal{L}(u_2) = \mathcal{C}(f + r_1 - u_2), \tag{11}$$

and the new residual is defined as

$$r_2 = f + r_1 - u_2.$$

In general, the m-th iterate of Bregman iteration, u_m, is computed as the restoration for the signal $f + r_{m-1}$, i.e.,

$$\mathcal{L}(u_m) = \mathcal{C}(f + r_{m-1} - u_m), \quad m \geq 1, \tag{12}$$

where $r_0 = 0$, and the new residual is defined as

$$r_m = f + r_{m-1} - u_m. \tag{13}$$

As for other conventional PDE-based denoising models, each step of the Bregman iteration (12) may be parameterized by an artificial time t for a convenient numerical simulation. That is, u_m can be considered as an evolutionary function and the corresponding evolutionary equation can be obtained by adding $\frac{du_m}{dt}$ on the left side of (12).

$$\frac{du_m}{dt} + \mathcal{L}(u_m) = \mathcal{C}(f + r_{m-1} - u_m). \quad m \geq 1. \tag{14}$$

The explicit temporal discretization of (14) can be formulated as

$$u_m^n = u_m^{n-1} + \Delta t \left[-\mathcal{L}(u_m^{n-1}) + \mathcal{C}(f + r_{m-1} - u_m^{n-1}) \right], \tag{15}$$

where $u_m^0 = u_{m-1}$ and Δt denotes the temporal stepsize.

The Bregman iterative procedure first recovers fine scales of the image and then recovers the noise to converge to the observed noisy image f [8]. For this reason the procedure must be stopped *manually* when the quality of the obtained image appears satisfactory. Another research objective is to develop PDE-based denoising models which can restore images effectively and stop *automatically* satisfying the given stopping criterion.

2.3 The new denoising model

As an iterative refinement model for the basic denoising model of the form (9), we suggest the following. Given a noisy image f, set $v_0 = 0$ and find v_m by recursively solving

$$\mathcal{L}(v_m) = \mathcal{C}(f - v_m) + \mathcal{L}(v_{m-1}), \quad m \geq 1. \tag{16}$$

The new model deserves the following remarks.

1) The Bregman procedure (12) can be rewritten as

$$\mathcal{L}(u_m) = \mathcal{C}\, r_m = \mathcal{C}(f - u_m) + \mathcal{C}\, r_{m-1}.$$

When the problem is solved *exactly* in each iteration, we have $\mathcal{L}(u_m) = \mathcal{C}\, r_m$ for all $m \geq 1$. Thus the last term of the above equation, $\mathcal{C}\, r_{m-1}$, can be replaced by $\mathcal{L}(u_{m-1})$:

$$\mathcal{L}(u_m) = \mathcal{C}(f - u_m) + \mathcal{L}(u_{m-1}). \tag{17}$$

Since the first iterates of (12) and (16) are the same each other, i.e., $v_1 = u_1$, it follows from (16) and (17) that $v_m = u_m$ for all $m \geq 1$.
However in practice it is often the case that each step in (12) is solved *approximately*, employing an iterative linearized solver (inner iteration) as in (15). Thus $\mathcal{L}(u_{m-1})$ becomes different from $\mathcal{C}\, r_{m-1}$ and therefore v_m differs from u_m in practice. It has been numerically verified that the iterates of the new algorithm v_m are smoother than those of the Bregman iteration (u_m). It has also been observed that as m becomes large, v_m shows a tendency of gaining the noise from the observed image f, but much more slowly than u_m.

2) As for the Bregman iterative refinement, each step of the new denoising model (16) can be parameterized by an artificial time t for a convenient numerical simulation.

$$\frac{dv_m}{dt} + \mathcal{L}(v_m) = \mathcal{C}(f - v_m) + \mathcal{L}(v_{m-1}), \quad m \geq 1. \tag{18}$$

Its equilibrium solution is a smooth, restored image of f with $\mathcal{L}(v_{m-1})$ incorporated as an extra source. It has been numerically verified that the extra curvature source allows the new iterate v_m to reveal fine features more clearly. The explicit temporal discretization of (18) reads

$$\begin{aligned} v_m^n &= v_m^{n-1} + \Delta t \left[-\mathcal{L}(v_m^{n-1}) \right. \\ &\qquad \left. + \mathcal{C}(f - v_m^{n-1}) + \mathcal{L}(v_{m-1}) \right], \end{aligned} \tag{19}$$

where $v_m^0 = v_{m-1}$ and Δt is the temporal stepsize.

2.4 Variable constraint parameter

For various denoising models [14], [15], [16], it has been numerically verified that the restoration quality depends strongly on the choice of the constraint parameter $\mathcal{C}$. Since the noise level is often unknown, it is hardly possible for the parameter to be set as in (6)–(7).

An effective strategy for the choice of constraint parameter is as follows. Below $\mathcal{C}_m$ denotes the constraint parameter for the m-th (outer) iteration.

1) For the first iterate ($m = 1$) in (16), select a reasonable constant $\lambda > 0$ and set

$$\mathcal{C}_1(\mathbf{x}_{ij}) = \lambda \, \frac{1}{|N_{ij}|} \int_{N_{ij}} |\nabla f| \, d\mathbf{x}, \tag{20}$$

where $\mathbf{x}_{ij}$ is a pixel point in Ω and N_{ij} denotes a neighborhood of $\mathbf{x}_{ij}$. For example, when the neighborhood is set as the 3×3 window centered at $\mathbf{x}_{ij}$, the constraint parameter (20) becomes

$$\mathcal{C}_1(\mathbf{x}_{ij}) = \frac{\lambda}{9} \sum_{p=i-1}^{i+1} \sum_{q=j-1}^{j+1} |\nabla f|_{pq}, \tag{21}$$

where the subscript pq indicates the evaluation at the pixel point $\mathbf{x}_{pq}$.

2) For $m \geq 2$, we first estimate the noise variance, using (7):

$$\sigma_{m-1}^2 = \frac{1}{|\Omega|} \int_\Omega (f - u_{m-1})^2 \, d\mathbf{x}. \tag{22}$$

Then, compute the spatially-varying constraint parameter, using the same strategy involved in (6):

$$\mathcal{C}_m(\mathbf{x}_{ij}) = \frac{1}{\sigma_{m-1}^2} \frac{1}{|N_{ij}|} \int_{N_{ij}} (f - u_{m-1}) \, \mathcal{L}(u_{m-1}) \, d\mathbf{x}. \tag{23}$$

When the neighborhood is set again as the 3×3 window centered at $\mathbf{x}_{ij}$, the constraint parameter in (23) reads

$$\mathcal{C}_m(\mathbf{x}_{ij}) = \frac{1}{9\sigma_{m-1}^2} \sum_{p=i-1}^{i+1} \sum_{q=j-1}^{j+1} [(f - u_{m-1}) \, \mathcal{L}(u_{m-1})]_{pq}. \tag{24}$$

Note that the above constraint parameter is computed once for each of outer iterations and it is not updated during the inner iteration.

3. Experiments for HS Computing

The first step of the HS computing experiment is to settle a research project which is publishable; the project would require the students a reasonable amount of programming work and make them learn about computing. For the purpose, we consider the following three PDE-based denoising models, of which $\mathcal{M}_2$ and $\mathcal{M}_3$ are new models to study that would reduce drawbacks of the model $\mathcal{M}_1$.

($\mathcal{M}_1$) The Bregman iterative refinement method (15), with $\mathcal{C}_m^n = \lambda |\nabla u_m^{n-1}|$ for a constant λ

($\mathcal{M}_2$) The new iterative refinement model (19), with $\mathcal{C}_m^n = \lambda |\nabla u_m^{n-1}|$ for a constant λ

($\mathcal{M}_3$) The new iterative refinement model (19), with $\mathcal{C}_m^n = \mathcal{C}_m$ as presented in (21) and (24)

Here the ITV model (8) is selected for the basic PDE-based model, i.e.,

$$\mathcal{L}(u) = -|\nabla u| \, \kappa(u) = -|\nabla u| \, \nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right),$$

and $\mathcal{C}_m^n$ denotes the constraint parameter for the m-th refinement (outer) iteration and the n-th inner iteration.

The model $\mathcal{M}_1$ is implemented in both C++ and Matlab/Octave and refactored in order to improve code readability and reduce complexity. Table 1 shows the implemented functions and their operations; each function is saved into a separate file. The codes are distributed to two different groups of students in a different way, to be used as a basic software for the algorithm development for the new models $\mathcal{M}_2$ and $\mathcal{M}_3$.

Group 1: The first group consists of three HS students and a undergraduate student (junior). The three HS students gathered in Seoul, S. Korea, early August 2012 for a week to learn about Matlab, to carry out small programming projects, and to be instructed about the denoising model $\mathcal{M}_1$ and its corresponding Matlab code. Their first mission is to add/modify M-functions for the simulation of models $\mathcal{M}_2$ and $\mathcal{M}_3$. In order to help them carry out the required programming, one-hour group meetings are arranged weekly through the SKYPE connection and its screen share capability. For the model $\mathcal{M}_2$, each of the HS students could easily develop the code by adjusting `Bregman.m` and `TimeMarch.m`. For $\mathcal{M}_3$, they are introduced to use the Matlab built-in functions `mean` and `imfilter` for the computation of quantities in (21), (22), and (24). The undergraduate student has been guided similarly from the mid of August 2012, with one-hour weekly meetings being held face-to-face. By the end of November 2012 (in four months from the beginning), all four students could implement the required functions in Matlab to analyze three different denoising models. (One of the HS students carries a Mac; the Matlab code could run on Octave, a public-domain high-level programming language, with a couple spots of minor modifications.) Their research results will be briefly presented in Section 4.

In early December 2012, the three HS students in S. Korea gathered again to learn basics of C++ and to be instructed about the C++ code for the model $\mathcal{M}_1$. The functions in the C++ code are composed with exactly the same names and operations as those in the Matlab code. Similarities/differences in the two computational

Table 1: Implemented functions and their operations.

Function	Input	Output	Operations
Main			set algorithm parameters; call Bregman
Bregman	f	denoised image	outer iteration; call TimeMarch
TimeMarch	f, u_{m-1}, r_{m-1}	u_m	inner iteration
GradMagnitude	u	$\lvert\nabla u\rvert$	
Curvature	u	$\mathcal{L}(u)$	

languages were fully discussed for a week; C++ functions for `mean` and `imfilter` were implemented for students' convenience. The undergraduate student was advised similarly in the mid of December 2012. Their second mission is to add/modify the C++ functions for the simulation of the new models $\mathcal{M}_2$ and $\mathcal{M}_3$. As of the beginning of March 2013, all the four students made their mission accomplished with a help from one-hour research meetings arranged twice a month.

Group 2: The second group consists of two undergraduate students who are interested in image processing. They were instructed for basics of C++ and the C++ basic software used for the first group, for two weeks in the second half of August 2012. Their mission was the same: the implementation of C++ functions for the simulation of models $\mathcal{M}_2$ and $\mathcal{M}_3$. Again one-hour meetings are arranged weekly, face-to-face, from September 2012. As of the beginning of March 2013, one of the undergraduate students could finish his implementation, while the other was spending a hard time on debugging. (The successful student began working on an image interpolation project dealing with another basic software written in a combination of C, C++, and Fortran, running on Linux operating systems.)

From the pedagogical experiments, we have found that

- HS students can learn C++ more easily when they begin with one of easy-to-use computation languages such as Matlab and Octave.
- A research environment can be constructed for HS students to be able to make fundamental contributions to interdisciplinary research in computer science and applied mathematics. Two most important components are (1) to partition the problem into easy-to-solve small modules (subject modularization [17]) and (2) to provide the students with an appropriate basic software.

The authors deeply appreciate the students volunteered for the HS computing education experiments. Their research results in image processing will be published in separate articles [18], [19], [20].

4. Research Results

In this section we present a set of research results carried out by Group 1. Figure 1 shows the Lena image and its noisy image contaminated by a Gaussian noise of PSNR 24.4 (dB). Here the PSNR (peak signal-to-noise ratio) is defined as

$$\text{PSNR} \equiv 10\log_{10}\Big(\frac{\sum_{ij} 255^2}{\sum_{ij}(f_{ij}-h_{ij})^2}\Big)\,\text{dB}, \qquad (25)$$

where f is the original image and h denotes the compared image.

Table 2: PSNR analysis.

Iteration	1	2	3	4	5
$\mathcal{M}_1$	30.17	31.07	29.93	28.62	27.96
$\mathcal{M}_2$	30.17	31.06	31.05	31.05	
$\mathcal{M}_3$	30.32	31.31	31.27	31.26	

Table 2 exhibits a PSNR analysis for the three denoising models: $\mathcal{M}_1$, $\mathcal{M}_2$, and $\mathcal{M}_3$. For the models, the outer iteration is stopped when the following 1%–tolerance is satisfied:

$$\|u_m - u_{m-1}\|_\infty < 0.01.$$

The inner iteration is terminated when either the 1%–tolerance is satisfied or the maximum 40 iterations are performed. The Bregman iterative refinement ($\mathcal{M}_1$) is stopped in five iterations manually, while both of the new models ($\mathcal{M}_2$ and $\mathcal{M}_3$) have converged in four outer iterations. As one can see from the table, the Bregman refinement shows the highest PSNR value at the second outer iteration; however, the PSNR decreases rapidly for later iterations, gaining the noise from the noisy image f. On the other hand, the new models have converged in four iterations, although they reveal a weak tendency of catching up the noise from the given image. Model $\mathcal{M}_3$ results in the best image in its second iterate.

In Figure 2, we collect the restored images by the three models: (a) the second iterate of $\mathcal{M}_1$, (b) the fourth iterate of $\mathcal{M}_2$, and (c) the fourth iterate of $\mathcal{M}_3$. The restored image of second model ($\mathcal{M}_2$) appears quite similar to the best result of the Bregman iteration. Model $\mathcal{M}_3$ is able to produce a better image showing clearer edges; for instance, look at the edges of the dark circular region in the right-side of the image, which is the boundary of the mirror in the image.

5. Conclusions

This article has studied how to teach high school (HS) students C++ more effectively and how to construct a research

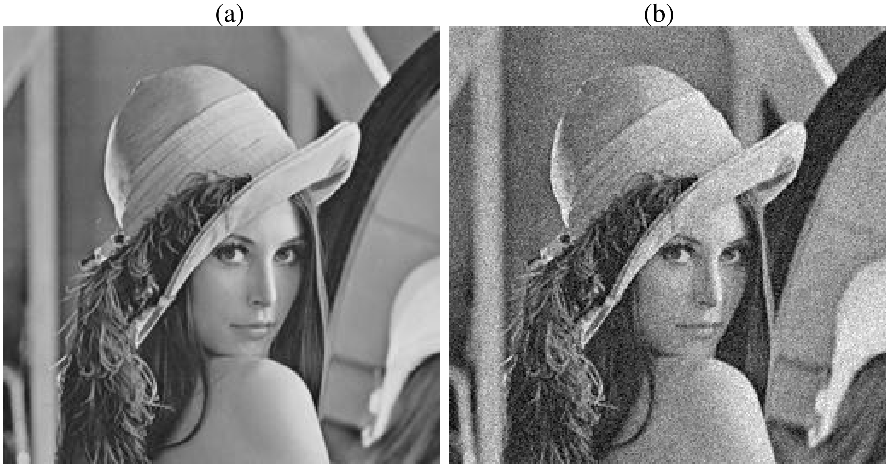


Fig. 1: Lena: (a) The original image in 256×256 and (b) a noisy image of PSNR$= 24.4$ (dB).

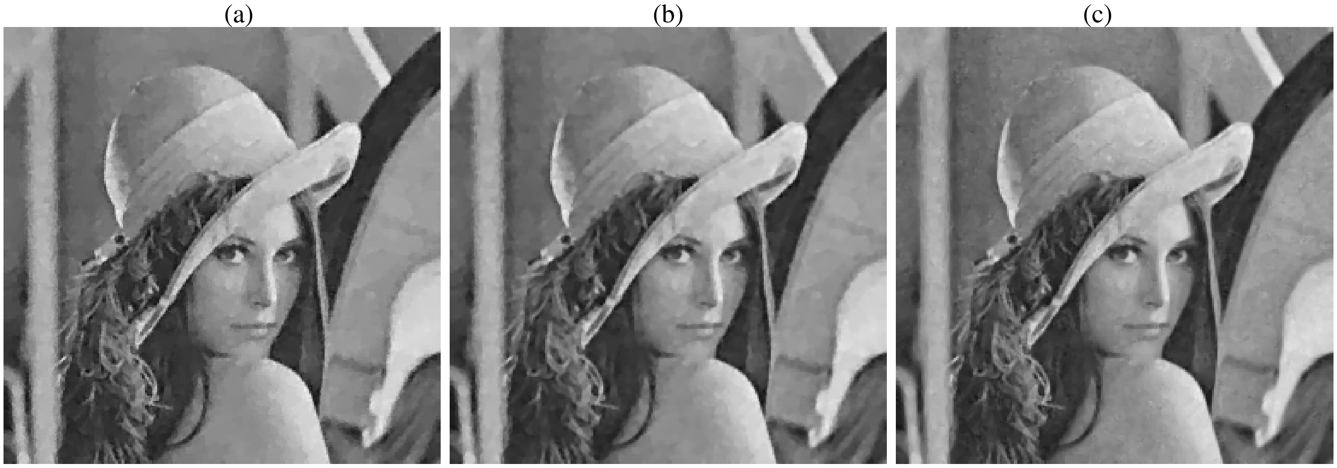


Fig. 2: Lena: Restored images by (a) $\mathcal{M}_1$, (b) $\mathcal{M}_2$, and (c) $\mathcal{M}_3$. The restored images show PSNRs of 31.1, 31.1, and 31.3 respectively for $\mathcal{M}_1$, $\mathcal{M}_2$, and $\mathcal{M}_3$.

environment which allows them to make significant contributions to interdisciplinary research in computer sciences and applied mathematics. A research project has been launched by introducing a couple of new PDE-based denoising models which are variants of the Bregman iterative refinement. The method of subject modularization is employed to partition the mathematical subjects into easy-to-solve small modules, which has helped the students learn fundamentals of the denoising models. Then, in order for the students to learn C++ effectively, a basic software for the Bregman refinement model has been implemented in Matlab/Octave and C++. The students are asked to implement a code for each of the new models by modifying/adding functions in the Matlab basic software first and then functions in the C++ basic software. It has been found from the experiment that HS students can learn C++ more easily when they begin with an easy-to-use computational language such as Matlab/Octave and that they can make significant contributions to real-world research when the subjects are modularized appropriately and a basic software is provided.

Acknowledgment

Jiho Kim is a HS student who is involved in the computing education research and has implemented required functions very effectively in Octave and C++. S. Kim's work is supported in part by the NSF grant DMS-1228337.

References

[1] A. Mehrizi-Sani, "Everyday electrical engineering: A one-week summer academy course for high school students," *IEEE Trans. Educ.*, vol. 55, no. 4, pp. 488–494, 2012.

[2] R. Mayer, "Learners as information processors: Legacies and limitations of educational psychology's second metaphor," *Educational Psychologist*, vol. 31, pp. 151–161, 1996.

[3] T. Shuell, "Cognitive conceptions of learning," *Review of Educational Research*, vol. 56, pp. 411–436, 1986.

[4] J. Bruner, *Child's Talk: Learning to Use Language*. New York: Norton, W. W. & Company, Inc., 1983.

[5] A. Luria, *Cognitive Development: Its Cultural and Social Foundations*. Cambridge, MA: Harvard University Press, 1976.

[6] L. Vygotsky, *Mind in Society: The Development of Higher Psychological Processes*. Cambridge, MA: Harvard University Press, 1978.

[7] L. Bregman, "The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming," *USSR Computational Mathematics and Mathematical Physics*, vol. 7, no. 3, pp. 200–217, 1967.

[8] A. Marquina and S. Osher, "Image super-resolution by TV-regularization and Bregman iteration," *J. Sci. Comput.*, vol. 37, pp. 367–382, 2008.

[9] R. Weinstock, *Calculus of Variations*. New York: Dover Pubilcations, Inc., 1974.

[10] L. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D*, vol. 60, pp. 259–268, 1992.

[11] A. Marquina and S. Osher, "Explicit algorithms for a new time dependent model based on level set motion for nonlinear deblurring and noise removal," *SIAM J. Sci. Comput.*, vol. 22, pp. 387–405, 2000.

[12] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*. New York: Springer-Verlag, 2003.

[13] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin, "An iterative regularization method for total variation-based image restoration," *Multiscale Model. Simul.*, vol. 4, no. 2, pp. 460–489, 2005.

[14] S. Kim, "Image denoising via diffusion modulation," *International Journal of Pure and Applied Mathematics*, vol. 30, no. 1, pp. 71–92, 2006.

[15] ——, "PDE-based image restoration: A hybrid model and color image denoising," *IEEE Trans. Image Processing*, vol. 15, no. 5, pp. 1163–1170, 2006.

[16] S. Kim and H. Lim, "Fourth-order partial differential equations for effective image denoising," *Electronic Journal of Differential Equations, Conference Special Issue*, vol. 17, pp. 107–121, 2009.

[17] S. Kim, H. Lim, D. Kim, and M. Tynan, "Subject modularization and research projects with high school students on mathematical image processing," in *Proceedings of the IASTED International Conference on Education and Technology*, 2005, pp. 247–252.

[18] W. T. Cordell, Hakran Kim, J. L. Willers, and S. Kim, "Image reconstruction for arbitrarily spaced data using curvature interpolation," in *Proceedings of The 2013 International Conference on Image Processing, Computer Vision, and Pattern Recognition*, Las Vegas, Nevada, USA, July 22-25, 2013.

[19] A. Gautam, J. Kim, D. Kwak, and S. Kim, "Iterative refinement by smooth curvature correction for PDE-based image restoration," in *Proceedings of The 2013 International Conference on Image Processing, Computer Vision, and Pattern Recognition*, Las Vegas, Nevada, USA, July 22-25, 2013.

[20] J. Moore, Y. Eum, and S. Kim, "Efficient edge-forming procedures for real-time image interpolation," in *Proceedings of The 2013 International Conference on Image Processing, Computer Vision, and Pattern Recognition*, Las Vegas, Nevada, USA, July 22-25, 2013.

Incorporating Output Interface in Studying Computer Graphics in Computer Science

Hassan Farhat
University of Nebraska at Omaha, Omaha, NE, USA

Abstract

The concepts of digital design in the hardware track of computer science often skip the input and output interface requirements. These are due to constraints placed on the number of classes offered in the track. As a result, the coverage is often limited to discussions at the gates and registers levels. In order to study input and output requirements, a student may need additional classes from electrical and computer engineering.

In computer graphics, many of the existing computer science textbooks give a brief description of input and output devices. We feel a more detailed discussion of hardware devices is beneficial to students understanding of the graphics topic. In this paper we propose a set of topics that can be included in teaching such a course. These can be added without compromising the coverage to the additional traditional topics covered in such a course. We accomplish this by integrating knowledge of digital logic with a small set of topics from electrical and computer engineering. The topics we present are absent from the many textbooks available on computer graphics. We apply the topics coverage to the graphics-utility-kit library extension to Open GL. The contribution of the paper is in compiling the needed topics that can be covered under curricula constraints.

1 Introduction

The hardware track in computer science and computer engineering spans coverage of topics from simple digital designs to complex designs as in designs of central processing units. The topics include Very Large Scale Integration [1, 2], digital design [3] through [8], computer organization [9, 10], and computer architecture [11] through [13]. Recent popularity of hardware languages introduced flexibility in the hardware track. At one end of the spectrum, one can model an entire computer as a finite-state machine. A program written in VHDL or Verilog can be used to describe the computer at the behavioral level, Fig. 1 [14]. In the figure the computer is modeled as a finite state machine with datapath (FSMD). In Fig. 2 a general VHDL structure of the design is given at the behavioral level. In [15] an AC-based computer is written as a large case statement.

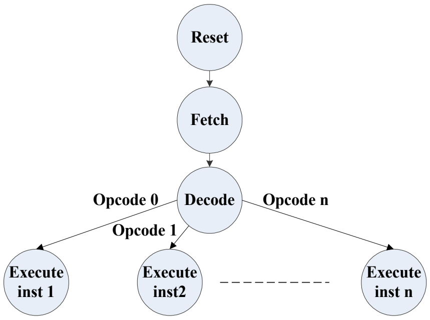


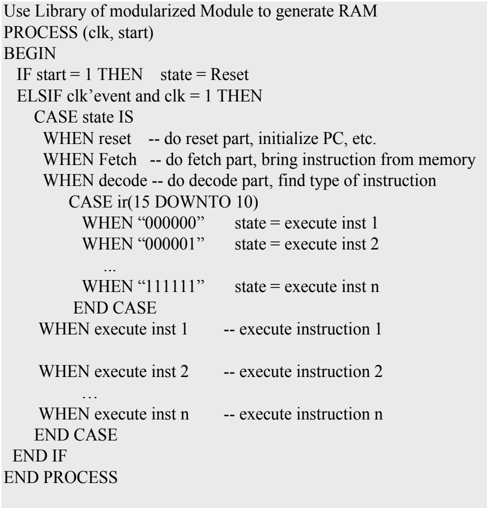
Figure 1: Simple computer as an FSMD

In computer science, unlike computer engineering, due to curricula constraints, the required hardware topics are normally constrained to the gate and register levels. As a result, additional courses and textbooks normally ignore the electrical lower level discussions needed in the study of input and output, for example. This applies as well to the study of computer graphics.

Given the user interaction with displays in computer graphics, we feel a more detailed coverage of the hardware aspects of computer graphics is needed. This can be accomplished without compromising coverage of traditional topics found in graphics textbooks, [17] trough [22], and can be beneficial to better understanding of the software topics.

We propose a minimal set of electronics topics in the discussion of monitor design. Simple knowledge of current and voltage computations is needed in the discussion (this topic can also be covered in a one lecture discussion). To speed the coverage process, we use the Multisim software package in circuit simulation [16].

The paper is organized as follows. In section 2, we discuss LEDs and LCDs as basic output interface elements. In section 3 we discuss CMOS transistors as elements that makeup today's monitors. Section 4 discusses the construction of an LCD/LED monitor. In section 5 we look at a simple construction of a graphics output system with a frame buffer, video graphics controller and a monitor.



```
Use Library of modularized Module to generate RAM
PROCESS (clk, start)
BEGIN
  IF start = 1 THEN    state = Reset
  ELSIF clk'event and clk = 1 THEN
     CASE state IS
       WHEN reset    -- do reset part, initialize PC, etc.
       WHEN Fetch   -- do fetch part, bring instruction from memory
       WHEN decode -- do decode part, find type of instruction
           CASE ir(15 DOWNTO 10)
              WHEN "000000"      state = execute inst 1
              WHEN "000001"      state = execute inst 2
                …
              WHEN "111111"      state = execute inst n
           END CASE
       WHEN execute inst 1        -- execute instruction 1

       WHEN execute inst 2        -- execute instruction 2
          …
       WHEN execute inst n        -- execute instruction n
     END CASE
  END IF
END PROCESS
```

Figure 2: Behavioral realization of simple computer in VHDL

Section 6 includes the need for double buffering in computer animation and show how double buffering works in the context of the Open GL API. The conclusions are given in section 7.

2 The LED and LCD Output Interface

The Light-Emitting-Diode (LED) and the Liquid-Crystal-Display (LCD) are among the most common building blocks for visual output interface. The elements are used to build outputs that range from seven-segment-displays to LCD and LED monitors. We present the LED and LCD topics as non-linear devices that have simple first-order approximations. In order for the LED to work properly, the current passing through the LED must fall within a range of minimum and maximum values. Students can be asked to study the current range as a simple application of Kirchhoff's voltage (KVL) and current (KCL) laws; the study can be related to the current sourcing and sinking capabilities of a given logic gate. An example using Multisim is shown in Fig. 3.

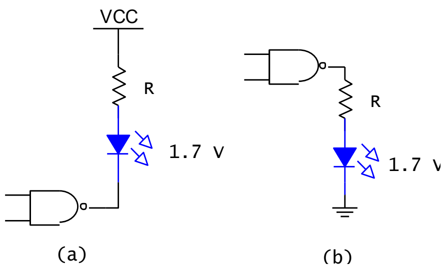


Figure 3: LED current-limiting resistors

KVL can be applied to determine the range of the resistor values; an LED requires a current in the range of 10 to 16 mA with a voltage drop of 1.7 V across the diode. For part (a) KVL results in VCC – iR – 1.7 – VOL = 0, where VOL is the gate LOW output voltage. This gives a range of the values for the current limiting resistor. An emphasis can then be placed in the above circuit on the concepts of the output sinking and sourcing current capabilities of a logic gate; in TTL, the NAND gate is able to sink the needed current but unable to source it. Hence one can't connect the LED as shown in Fig. 4b).

The same concepts of current limiting resistors can then be applied to seven-segment displays as a special case of multiple LEDs. Here the concepts of common anode and common cathode can be explained. Fig. 4 shows an illustration.

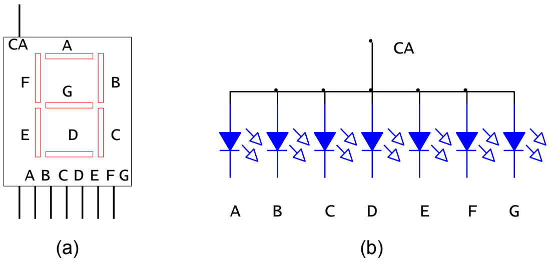


Figure 4: Seven-segment displays as 7 LEDs

From the figure it is illustrated that the same rules applied to LEDs are applied to the seven-segment display. Each of the 7 segments must have a current limiting resistor connected to it. The interface with BCD-to-seven-segment-decoders can then be discussed.

Similar discussions can be carried about LCDs. Following the discussion, the transistor element is discussed. The element in combination with LCDs and LEDs form the pixels and colors on a transistor-based flat-panel monitor.

3 The Transistor and LED

Among the building blocks of a monitor is an array of Thin-Film-Transistors (TFTs) that, in combination with LCDs and LEDs, form the pixels of a monitor. As a result, a brief discussion of transistor LED and LCD construction is helpful. Fig. 5 shows a common type of transistor in use, the Enhancement-Mode Metal-Oxide-Semiconductor Field-Effect- Transistor (E-MOSFET).

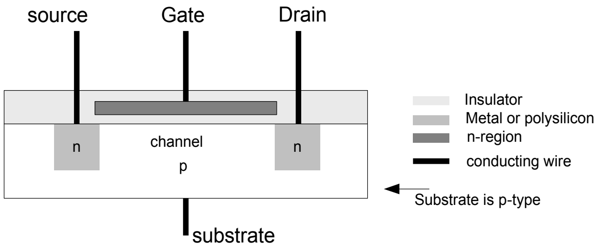


Figure 5: MOSFET Transistor

The circuit is composed of three inputs (source, gate and drain). The figure is a 2D view of the transistor. A 3D view of the gate part with the surrounding insulator (SiO_2, glass) is shown in Fig. 6.

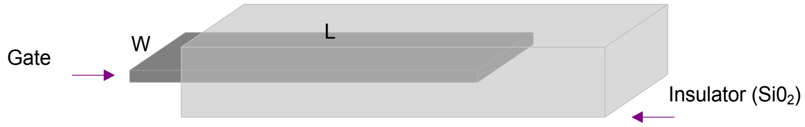


Figure 6: 3D view of gate part of MOSFET transistor

A brief explanation can be made about the transistor acting as a controlled switch connecting the source to the drain (with the gate input functioning as the switch control), Fig. 7 (part (b) shows the transistor as a switch).

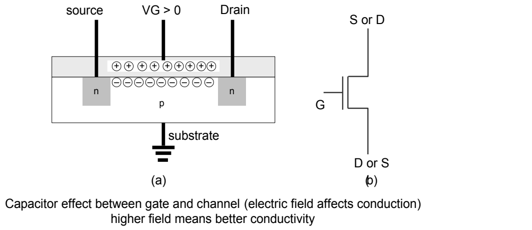

source
VG > 0
Drain
n
p
n
substrate
(a)
S or D
G
D or S
(b)
Capacitor effect between gate and channel (electric field affects conduction)
higher field means better conductivity

Figure 7: (a) MOSFET with conducting channel, (b) MOSFET as a switch

In the above, a positive voltage (of certain threshold VGS) applied on G will induce a negative channel of electrons; which results in a conduction path from source to drain. This is due to the capacitor effect resulting from the insulator (Fig. 6). Similarly, the absence of a positive voltage will cause the transistor to act as an open switch (source not connected to drain).

4 The Graphics LCD Monitor

With the above brief discussion, students can then study the LCD or LED graphics monitors. The basic pixel construction of the monitor is based on connecting a TFT to an LCD. For colored monitors three TFTs can be used to control the intensities of the three primary colors (RGB colors). Fig. 7 shows a simplified pixel construction using TFT technology.

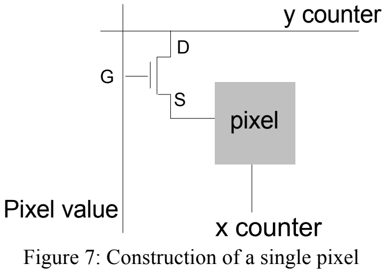


Figure 7: Construction of a single pixel

In the figure, the x and y values correspond to the pixel location on the screen and are connected to the source and drain part of the transistor. An LCD is shown connected as a small square, labeled pixel. Similar to an LED, in order for the LCD to turn on, a voltage difference must exist between the lines labeled s and x counter. The voltage is controlled by the voltage values on the G input and the x and y counters. In order for the switch to work in normal mode, the y (D) voltage value must be positive with respect to the S value. With the switch in normal mode, the gate input controls the switch (open/closed). A positive voltage on gate closes the switch and creates a voltage difference on the terminals of the LCD (similar to LED) causing it to appear lit. A 0 voltage on the gate input turns the pixel off.
A simplified view of a 4 by 4 monitor based on TFT technology is shown in Fig. 8.

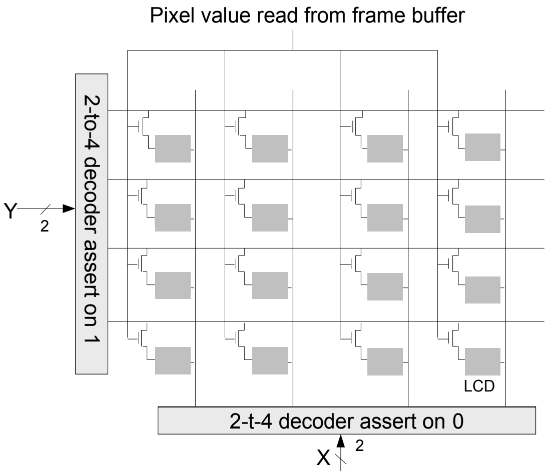

Pixel value read from frame buffer
2-to-4 decoder assert on 1
Y
2
LCD
2-t-4 decoder assert on 0
X
2

Figure 8: A 4 by 4 monitor based on Thin-Film-Transistors

The X and Y inputs are from counters built as part of the monitor. The counters are connected to the inputs of a row decoder (Y decoder) and the inputs of a column decoder (X decoder). The counters count clock pulses of certain period and can be reset to 0 synchronously. The Y decoder outputs are asserted on HIGH (1) while the X decoder outputs are asserted on LOW (0). Hence at any instance of time, only one TFT is functioning in normal mode (all others are turned off since the source and drain are not held at proper voltages). For the transistor functioning in normal mode, the input to the gate determines the pixel value. This value is read from the frame buffer. Following the construction, the X and Y counters sizes can be discussed in the context of a monitor's native resolution (number of pixels on the screen).

6 The Frame Buffer, Video Graphics Controller and Monitor

With the above discussion, more detailed coverage of the monitor interface can be given. The coverage can include a simple video graphics controller and a frame buffer. A schematic of a simple video graphics controller is shown in Fig. 9.

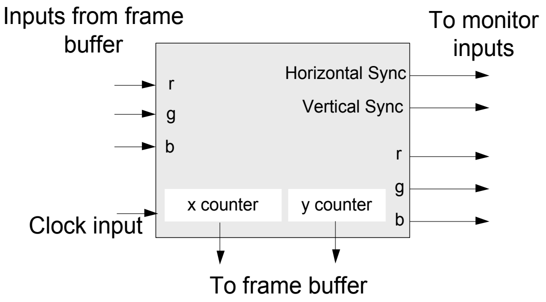


Figure 9: Video graphics controller in block diagram

The controller has two groups of outputs: 1) counter outputs that are used to access the frame buffer, and 2) outputs connected to the monitor.

The x and y counter outputs are used to read the current frame buffer memory pixel value. The pixel values are used as inputs to the video controller, which in turn buffers the pixel values and passes them to the monitor on its RGB outputs.

The outputs connected to the monitor are the three RGB pixel color outputs, and two additional synchronizing signals (horizontal and vertical counter syncs). The video controller counters and the monitor counters must be synched so that the proper frame buffer value at a given (x, y) coordinates is placed on the correct screen pixel position. Fig. 10 shows a simplified combined construction that includes a monitor, a video controller and a frame buffer.

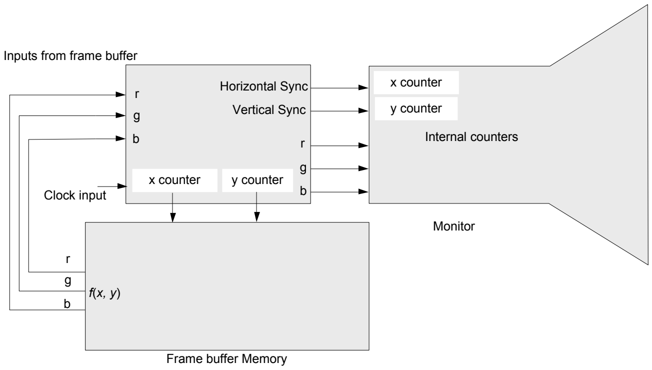


Figure 10: Frame buffer, VGA, and monitor construction *f*(*x*, *y*) is pixel value

Here discussions on resolution, frame buffer access time and refresh rate can be made. Constraints on access time computations results in the need to read multiple pixels simultaneously (entire row). We illustrate this in Fig. 11.

In the figure, the three frame buffers represent a single buffer containing the RGB colors. The frame buffer has a single read address bus, Y, which reads an entire row of pixels and places the pixels at the inputs of the multiplexers. Using the x counter, the pixel values can be then multiplexed out at the speed of the multiplexer, thus reducing memory read demands.

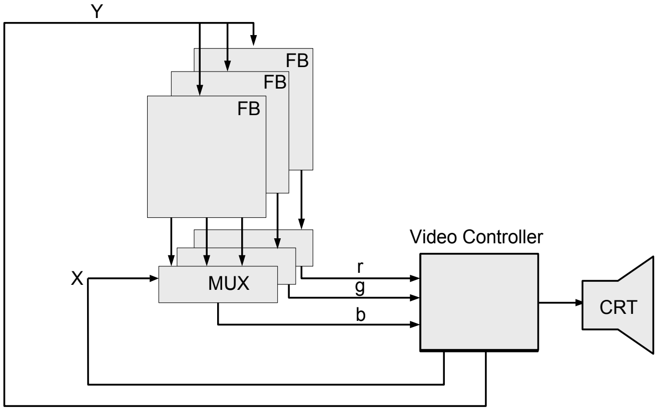


Figure 11: The need for multiplexing in reading fame buffer

6 From LUT to Double Frame Buffers and Computer Animation

The discussion on monitor interface may conclude with a brief review of the earlier buffers construction to today's inclusion of double-buffering used in computer animation. For this we can use today's Open GL API and an earlier popular graphics instruction set by Digital Equipment Corporation.

In Open GL, using the windows operating system and C++, we instruct the hardware to use double-buffering with the statement "glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB)" language binding. The function constants are set in such a way to accommodate incorporating graphics hardware from earlier frame buffers constructions to today's flexible options (GLUT_DOUBLE versus GLUT_SINGLE). The second argument in above function sets the type of data stored in the frame buffers, actual color values or index values into a color table (GLUT_RGB versus GLUT_INDEX).

In some earlier computer graphics classes, a graphics instruction set called ReGIS (Remote Graphics Instruction Set) was used. The set was provided by Digital Equipment Corporation (DEC), and was tailored to be used on special graphics terminals such the VT240. Due to the cost of hardware then, the frame buffer was set so as to hold 2 bits per pixel, resulting in a maximum of four simultaneous colors per frame.

To increase the pool of total possible colors to select from, actual colors were stored in a 4-row table. Each row in the table is 24 bits, which can be set by the user. Fig. 12 shows a schematic. Today this option is seldom used.

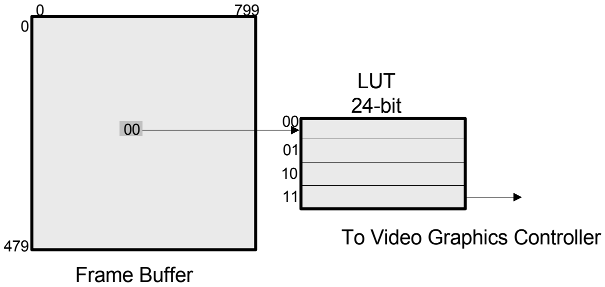


Figure 12: Frame buffer with LUT table

Instead, actual colors are stored in frame buffers and, further, double buffering can be used. Fig. 13 shows a simplified construction of a double frame buffer construction. The buffers are characterized as front and back buffers. The front buffer is used to display the current scene while the additional buffer (back buffer) is used to generate a next scene. When the next scene computation is completed, the graphics package is instructed to swap the roles of the front and back buffers. The video controller reads the pixel values from the new declared front buffer.

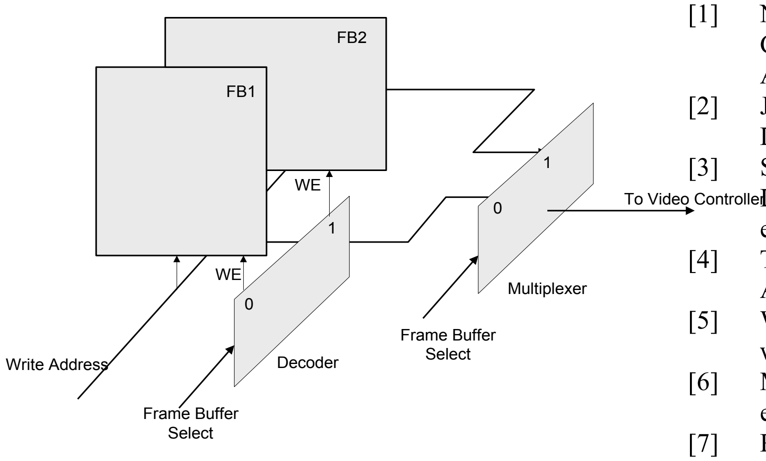


Figure 13: A simplified construction of a double frame buffer

Once a scene is completed and ready to be rendered, we instruct OpenGL to swap the roles of the buffers (the front buffer becomes the back buffer and vice versa). To accomplish this we use “glutSwapBuffers()”.

We illustrate how this can be accomplished in hardware by referring to Fig. 14. In the figure, the decoder outputs are used as write enables to the frame buffers. Hence writing is done to one frame buffer only. The Open GL statement glutSwapBuffer() can then be gated with additional hardware signals and used as input to the decoder select line of the buffer. Similar arguments are made about the multiplexer select lines. In the discussion we note to the students, that it is necessary to have unequal select values on the decoder and multiplexer select lines. This insures the frame read is different from the frame that is currently being updated.

6 Conclusion

In this paper we considered extending computer graphics topics coverage to include early introduction of hardware topics. In particular, we proposed including more detailed study of the monitor interface as part of studying computer graphics. The study of monitor interface includes incorporating a frame buffer and a video controller. While the study is more detailed than found in computer graphics textbooks in computer science, it can be covered without compromising the study of traditional topics found in the textbooks. We feel such a study is beneficial to students and provides better understanding of the input/output aspects of computer graphics.

References

[1] N. Weste and D. Harris. “CMOS VLSI Design: A Circuits and Systems Perspective 4th edition”. Addison Wesley, 2011.

[2] J. Vyemura. “A First Course in Digital Systems Design an Integrated approach”. Brooks/Cole, 2000.

[3] S. Brown and Z. Vranesic. “Fundamentals of Digital Logic with VHDL Design with CD-ROM, 3rd edition”. McGraw-Hill, 2009.

[4] T. Floyd. “Digital Fundamentals: A System Approach”. Prentice Hall, 2013.

[5] W. Kleitz. “Digital Electronics A Practical Approach with VHDL. 9th edition”, Prentice Hall, 2012.

[6] M. Mano and M. Cilette. “Digital Design, 5th edition”. Prentice Hall, 2013.

[7] R. Tocci. N. Widmer and G. Moss, “Digital Systems Principles and Applications, 11th edition”. Prentice Hall, 2011.

[8] J. Wakerly. “Digital Design Principles and Practices 4th edition”, Prentice Hall. 2006.

[9] D. Patterson and J. Hennessy. “Computer Organization and Design, Revised 4th edition”. Morgan Kaufmann, 2011.

[10] W. Stallings. “Computer Organization and Architecture, 9th Edition”. Prentice Hall, 2013.

[11] D. Harris and S. Harris. “Digital Design and computer Architecture”. 2nd edition, Morgan Kaufman, 2013.

[12] J. Hayes. “Computer Architecture and Organization 3rd edition”. McGraw-Hill, 1998.

[13] J. Hennessy and D. Patterson. “Computer Architecture: A Quantitative Approach 5th Edition”. Morgan Kaufmann, 2011.

[14] H. Farhat. “Integrating Electronics in Computer Science under Curricula Constraints”. Proc. IEEE EIT, 140-144, Ames, Iowa, USA, 2008.

[15] J. Hamblen and M. Furman. “Rapid Prototyping of Digital Systems, a tutorial approach”. Kluwer Academic Publishing, 2001.

[16] Electronics Workbench, www.electronicsworkbench.com.

[17] E. Angel. “Interactive Computer Graphics, A Top-Down Approach Using OpenGL, 6th edition”. Addison Wesley, 2012.

[18] E. Angel. “OpenGL, A Primer, 2nd edition”. Addison Wesley, 2005.

[19] Foley, van Dam, Feiner, Hughes. “Computer Graphics Principles and Practice, 2nd edition”. Addison Wesley, 1990.

[20] D. Hearn, M. Baker, W. Carithers. “Computer Graphics with OpenGL, 4th edition”. Prentice Hall, 2011.

[21] F. Hill, S. Kelly. “Computer Graphics using OpenGL, 3rd edition”. Prentice Hall, 2007.

[22] D. Rogers, J. Adams. “Mathematical Elements for Computer Graphics, 2nd edition”. McGraw Hill, 1990.

A Real-World Project to Apply Discrete Structures

Anja Remshagen
Department of Computer Science
University of West Georgia
Carrollton, Georgia, USA

Abstract - *There is a rising effort to design interesting and appealing applications to teach data structures and discrete structures. Common themes include games, animations, and robotics. While these applications are fun to work with, many students perceive these applications as toy problems that do not relate to real-world applications as they occur in a commercial setting. We describe a course project that is concerned with access control. Access control is of crucial importance in commercial applications. The project requires applying a variety of data structures as well as concepts from discrete mathematics. Thus, the project also addresses the increasing aspiration to integrate these areas in the classroom. An instructor can adopt the project in different ways: as a single course project, as a sequence of several projects, or as individual, independent assignments.*

Keywords: Computer science education; discrete mathematics; data structures; context-based learning; assessment;

1 Introduction

Discrete mathematics and data structures are part of the core body of knowledge of computer science according to the Ironman Draft for the Computer Science Curricula 2013 [1]. In many computer science programs, discrete mathematics is taught by the mathematics department of the corresponding institution. In this case, the content is typically delivered without context and without relation to computing. Many computer science students do not see the relevance of discrete mathematics for their major. As an additional obstacle, the material is more abstract than many other areas of computer science and thus perceived as more difficult. In case of data structures, students typically have to implement small sample applications using the data structures. There has been an increased effort to design interesting applications that appeal to students, using for example, games [3, 14, 16, 27, 28] and animations [8, 13, 17]. While these applications are fun to work with, many students still perceive these applications as toy problems that do not relate to real-world applications as they occur in a commercial setting, and they do not see the need to use advanced data structures in their career as computing professionals. In hands-on assignments for other courses, many students resort to linear data structures provided by the application programming interface of the corresponding programming language if they need to implement a collection. As the underlying data of an application in an educational setting is typically small, students do not recognize the need for a more efficient data structures in those courses either.

In this paper, we describe a course project that requires the implementation of an access control system using a simplified version of role-based access control. Access control is of increasing importance in the commercial and government sectors as the growing use of social networks, cloud computing, and geolocation, for example, raises privacy concerns and worries about identity theft. The European legislation, for example, is in the process of reforming its data protection regulations to address these concerns [10]. Obviously, it is not possible to address all these concerns in a course project and to expect students to develop solution approaches. Indeed, access control is a very active research area that is continuously challenged to tackle new issues. However, we can give the students a realistic view into how access to data can be regulated.

The project covers a variety of topics of discrete mathematics and advanced data structures including sets, relations, functions, propositional logic, hash tables, and graphs. The project shows the interaction among data structures, the operations on a structure, and discrete mathematics. The project has been designed as a final project for the two-course sequence Data Structures and Discrete Mathematics at the University of West Georgia. As data structures and discrete mathematics are closely intertwined, the computer science department at the University of West Georgia teaches these topics together in one two-course sequence, instead of separating the two topics into a data structures and algorithm course and a discrete mathematics course. This project did not only serve to assess the students' performance but also to assess the learning outcomes for accreditation purposes. The learning outcomes that can be assessed though this project are as follows:

- Apply mathematical foundation to data structures, algorithms, and other concepts in computer science.
- Select an appropriate data structure and algorithm to solve a problem.
- Implement data structures and related algorithms.

The described project can be adopted in different ways in a computer science course that teaches data structures, algorithms, or discrete structures. Parts of the project might also be used in a discrete mathematics course for computer science students that have already programming skills. The project can be used as an open-ended final project. The instructor can chose the level of provided support and guidance. The project can also be split into several smaller individual assignments to practice the implementation of a specific data structure, for example.

The remainder of this paper is structured as follows. Section 2 describes related work. Section 3 introduces the role-based access control model on which the course project is based. Section 4 describes the project. Section 5 discusses how the project can be modularized and split into individual, independent parts. Section 6 concludes with a summary.

2 Prior Work

Access control is an important and continuously evolving field as new regulations result in the development of new access control models. Many new access control models are based on *Role-Based Access Control (RBAC)*. To name only a few examples, consider temporal role-based access control [4], privacy-aware role-based access control [21] or event-driven RBAC [6]. For details about role-based access control, see Ferraiolo et al. [11] or Sandhu et al. [24], for example. The course project proposed in this paper is based on a simplified version of role-based access control.

Studies have shown that students are more motivated if material and concepts are presented in context [25, 26]. Games have been frequently used in computer science education to implement context-based learning as games seem to excite students [3, 14, 16, 27, 28]. Other themes used to teach computer science in context are educational robots, like the Scribbler [2] or the Finch robots [15], media computation [12], animations, like Alice [8], Greenfoot [13] or Scratch [17], and sustainability [9]. Most computer science students at the University of West Georgia will work in a commercial setting at software companies and information technology departments of other companies. Very few students will work in an area that was used to excite students for computing during their education, like gaming, graphics, or robotics. We present a project that is more relevant to our students in respect to their future career.

The student project and assignments presented here combine topics from data structures and discrete mathematics. There has been rising awareness that these topics are closely intertwined and should be integrated. There are efforts to explicitly teach mathematics, in particular, discrete mathematics, in computer science courses, and to offer mathematics courses for computer science students in which the math concepts are linked to the related computing topics. For example, Coffey [7] introduces a two-course data structures and algorithm sequence that integrates relevant mathematics, data structures and the analysis of algorithms. McMaster, Anderson, and Rague [19] demonstrate how to teach discrete mathematics with programming. Neff [20] describes a course that teaches mathematics related to regular languages. Oxley [22] describes how to teach discrete mathematics in context, in particular, employing artificial intelligence and software agents. Sahami [23] has developed a course on probability theory for computer scientists.

3 Role-Based Access Control

We describe the role-based access control model that is implemented in the course project. It is a simplified version of the model investigated by Ferraiolo et al. [11] and Sandhu et al. [24]. In almost any commercial application that involves the storage and management of data, access to that data needs to be controlled. Different users of a database have different access rights. A nurse might be allowed to read the drug information of a patient, but only a physician might be allowed to change the drug information. The manager of a large software project might have read access to all project related data. The programmer might only have access to some parts of the project, but is allowed to change particular parts of the programs. The number of possible users that need access to some of the information in a database is typically very large, and thus storing the access rights for each user individually can result in a huge amount of data itself. In role-based access control, the users are grouped into subsets of users with the same access rights. Thus only the permissions for the specific subsets, called *roles*, and the mapping that assigns roles to users need to be stored. These subsets of users are not necessarily disjoint since a user might be associated with several roles. Consider, for example, a RBAC system of a hospital that has the roles nurse, physician, and supervisor. The users Alice and Bob are nurses, Kim is a physician, and Alice and Kim are also supervisors. Hence both, the role nurse and supervisor, include the user Alice, and both, the role physician and supervisor, include the user Kim. If a user is associated with several roles, then that user obtains all the access permissions associated with each role. Assume in the above example that a nurse is allowed to read a patient's drug information, a physician is allowed to read and update the drug information, and a supervisor is allowed to read the time card of a hospital staff member. Then Alice is allowed to read the drug information of patients and to read the time cards of the staff since she has the role nurse and supervisor. Bob can read the drug information, but is not allowed to read time cards. Kim can read and update the drug information and read time cards.

Some roles are more powerful and should include the permissions of a more junior role. For example in a website development project, there might be the roles team member, web designer, graphics designer, and project manager. Web designers, graphics designers, and project managers are all team members and should inherit the permissions that are associated with the role team member. In addition, the project manager might need to access the parts of the project managed by the web designer and graphics designer and therefore inherits the corresponding permissions. The role hierarchy can be visualized graphically where the more senior roles are drawn towards the bottom. See Figure 1 for the graphical representation of the above role hierarchy. An edge from role r_1 to role r_2 means that r_1 inherits the permissions from r_2. The role hierarchy is transitive. That is, if there is a path from role r_1 to role r_2, then r_1 inherits the permissions from r_2 as well. A role hierarchy is also reflexive and antisymmetric. Therefore, it is a partial order. Note that the graphical representation does not display all edges of the actual graph of the partial order. The graphical representation corresponds basically to the Hasse diagram. The difference to the Hasse diagram of a partial order is that the Hasse diagram displays the minimal elements towards the bottom and the maximal elements towards the top. The edges in a Hasse diagram are undirected. The correct Hasse diagram of the relation is shown in Figure 2. Obviously there is a one-to-one correspondence to the graphical representation shown in Figure 1 and the Hasse diagram in Figure 2.

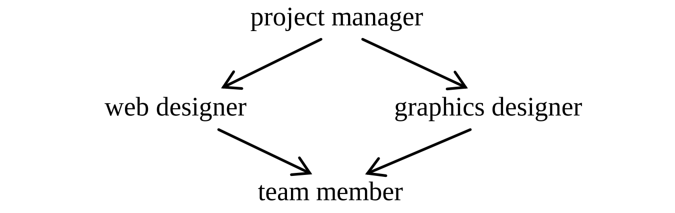


Fig. 1. Graphical representation of a role-hierarchy

Depending on whether students are more familiar with Hasse diagrams (a representation typically taught in a math-oriented course) or directed graphs (a representation typically used in a data structures course), the instructor may choose either one of the two possible representations. In the following, we will refer to the graphical representation of the role hierarchy as the *Hasse-like graph*.

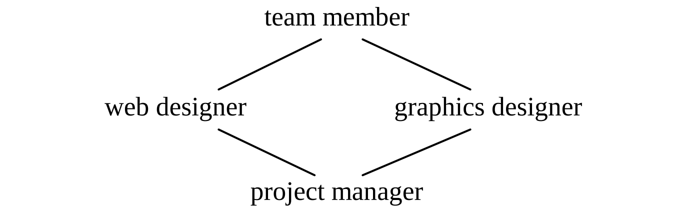


Fig. 2. Hasse diagram of a role-hierarchy

Role-based access control is composed of the following components:

- A set of users U; in an application, each person corresponds to a user. Users are assigned a unique *userid*.
- A set of roles R where the set U of users and set R of roles are disjoint sets.
- A partial order H on the set of roles called *role hierarchy*; the role hierarchy specifies if a role inherits the permissions of another role.
- A relation $U_{Roles} \subseteq U \times R$ that maps users to their roles. A user can have multiple roles. There are typically many users assigned to each role.
- A set D of data objects.
- A set A of actions; The set of actions specifies the kinds of different access rights. For example, a set of actions might be {read, create, update, delete, deny}. In the proposed project we consider the set {read, write, deny} of actions.
- A set of permission assignments that specifies which data objects can be accessed by the users with a particular role. The set of permission assignments is a subset of $R \times A \times D$.

The set of permission assignments is at the heart of a RBAC system since it specifies the permissions. A permission assignment is a triple of the form

(*role, action, data*)

where *role* ∈ R is a role, *action* ∈ A is an action, and *data* ∈ D is a data object. For example, the permission assignment (nurse, read, drug) assigns read access to the drug data for all users with the role nurse.

An access request is a triple of the form (*user, action, data*) where *user* ∈ U is a user, *action* ∈ $A\backslash$\{deny\} is an action, and *data* ∈ D is a data object. Generally, a specific access request (usr, act, dta) is granted if there exists a permission assignment of the form (rol, act, dta) and the user usr is assigned the role rol or the user usr is assigned a role that inherits all permissions from the role rol. We say that a user usr *is associated with the role* rol, if and only if the user usr is assigned the role rol or if the user usr is assigned a role that inherits all permissions from the role rol. In other words, an access request (usr, act, dta) is generally granted if the user usr is associated with the role rol. However, the decision whether access is granted depends also on the access policies that are specified for the access control system. There are two types of access policies that need to be specified:

(1) The first policy specifies how to handle permission assignments that include the deny actions. A system applies either the policy DENY_OVERRIDES or the policy PERMIT_OVERRIDES. In case of DENY_OVERRIDES, an access request (usr, act, dta) will not be granted if a permission assignment of the form (rol, deny, dta) exists where the user usr is associated with the role rol. It does not matter if an addition a permission assignment of the form (rol, act, dta) applies as well – access will not be granted. In case of PERMIT_OVERRIDES, an access request (usr, act, dta) will be granted if a permission assignment of the form (rol, act, dta) exists where the user usr is associated with the role rol. It does not matter if a permission assignment of the form (rol, act, dta) exists. Note that the permission assignments of the form (*role*, deny, *data*) are irrelevant if the policy PERMIT_OVERRIDES holds.

(2) The second policy specifies whether a write permission entails a read permission to a specific data object. Either the policy WRITE_IMPLIES_READ or WRITE_WITHOUT_READ applies. In case of WRITE_IMPLIES_READ, the permission assignment (rol, write, dta) implies that also read access is granted to the data object dta by a user who is associated with the role rol. In other words, an access request (usr, read, dta) is granted if a permission assignment of the form (rol, read, dta) or (rol, write, dta) exists where the user usr is associated with the role rol. Of course, if in addition the policy DENY_OVERRIDES holds, the access control system has still to verify that no permission assignment with the deny action applies. In case of WRITE_WIHTOUT_READ, read access is

not granted if only permission assignments with the write rule apply. An access requests (usr, act, dta) where act is not equal to read is handled as before in both cases.

4 The RBAC Project

In the course project proposed in this paper, the students have to implement an access control system that undergoes two phases: the setup phase (see Section 4.1) and the execution phase (see Section 4.2). In the setup phase, the system reads and stores all data that is relevant in order to decide whether an access request should be granted from a file. In the execution phase, the access control system receives and decides on access requests.

4.1 The Setup Phase

The access control system should read in the data of a specific example system from one or more files. The file(s) with the relevant information should be provided by the instructor. The following needs to be included in the file(s):

- The relation U_{Roles} that maps userids to roles; the set of all userids and roles can be constructed from that relation. However, a list of all userids and all roles may be provided in addition.
- The role hierarchy; the file should only specify the edges of the Hasse-like graph. The actual role hierarchy is the transitive closure of the specified relation. See Section 3 for details about the graphical representation of the role hierarchy.
- The set of permission assignments
- The specification of the two access policies: DENY_OVERRIDES or PERMIT_OVERRIDES, and WRITE_IMPLIES_READ or WRITE_WITHOUT_READ.

We strongly recommend providing the data of several different access control systems to the students that test different scenarios. At least one of the given scenarios should include a very large application.

In order to decide on how to store the data in the setup phase, students have to determine the operations that will be performed on the data in the execution phase. Then they can decide on the data structures and algorithms that are most suitable to effectively determine whether given access requests should be granted. We discuss the possible data structure options and associated algorithms. If the students are not given any help or hints on this task, we recommend that the students are required to turn in a preliminary report that discusses their choice of data structures and related operations before starting the implementation. That will encourage the students to go through the design process before starting the implementation. The instructor can provide feedback on the students' report.

In the following, we discuss each collection that needs to be implemented.

4.1.1 User roles

The students have to implement a data structure that stores the roles of each user. Users are represented by a unique userid.

The main operation on this data structure is a lookup operation to determine the roles of a given userid. As userids are unique, the abstract data type map or, equivalently, dictionary, needs to be implemented where the keys are the userids and each item in the map is a collection of roles. The map should be implemented through a hash table. Since the application needs only to iterate through the collection of roles for a given userid, a linear data structure, like a linked list, for example, is sufficient to implement each item in the map.

4.1.2 Role Hierarchy

The next data structure to be determined and implemented is the role hierarchy *H*. Recall that *H* is a partial order. In the execution phase, the access control system needs to determine all roles whose permission assignments are inherited by a given role. In other words, for a given role, all roles that are related to that role according to the relation *H* have to be determined. Recall that the input file only stores the edges of the Hasse-like graph, and therefore does not store explicitly all edges of the graph of the partial order. For example, consider the role hierarchy represented in Figure 1. The node project manager has only an outgoing edge to web designer and an outgoing edge to graphics designer. However, the project manager also inherits the permissions from team member, and thus project manager relates to team member in the order *H*. We can compute the transitive closure of the graphical representation to determine all edges of the graph of the partial order *H* between two different nodes.

We have two options to store the role hierarchy: (1) The RBAC system determines and stores the transitive closure of the given graph in the setup phase. In the execution phase, the outgoing edges of a given node are traversed. The corresponding end nodes represent all the roles whose permission assignments are inherited by the given node. Warshall's algorithm can be used to determine the transitive closure. Note that Warshall's algorithm is typically applied to the adjacency matrix representation of the graph. However, it is most efficient to store the resulting graph as an adjacency-list. (2) In the second option, we only store the specified edges of the given Hasse-like graph. Whenever the roles whose permissions are inherited by a given role need to be determined, the system computes all nodes that can be reached from that given node. This can be accomplished by breadth-first search. Again, an adjacency-list representation is most efficient in this case. Breadth-first search is linear in the number of edges among reachable nodes. Assuming that the given Hasse-like graph is sparse, the operations of both options have the same time complexity in the execution phase. However, the first option has a lower constant factor.

The advantage of the second option is a less time-consuming setup phase and lower space complexity.

The role hierarchy offers an opportunity to apply graphs and Warshall's algorithm. If graphs have not yet been covered in a course, the instructor can reduce the role hierarchy to a tree structure and students can implement option (2), i.e. breadth-first search.

4.1.3 Set of Permission Assignments

The last data structure to be implemented in the setup phase stores the permission assignments. The main operation on this structure requires to determine whether a given permission assignment exists. A hash table can be used again where the entire permission assignment constitutes the key. As the key consists of the three components userid, action, and data object, the students will not be able to resort to a suitable default hash function that might be provided by an application programming interface. They will have to apply the taught techniques used to design a hash function, like folding, for example.

Students might decide to implement additional operations on the collection of permission assignments. For example, recall that access requests are made by individual users and users are associated with a set of roles. Thus, one might want to determines whether for a given set *R'* of roles, an action act, and a data object dta, at least one permission assignment of the form (rol, act, dta) with rol ∈ *R'* exists. This operation can be directly implemented with the help of the previous operation.

4.1.4 Access Policies

The access policies don't require a data structure. Two variables are sufficient to store the corresponding values of each of the two policies.

4.2 The Execution Phase

In the execution phase, the RBAC system should read in access requests (usr, act, dta) from users of the system and determine whether the requested access should be granted. A simple user interface is sufficient to allow users of the system to enter an access requests. Figure 3 outlines a possible algorithm. Step 1-3 compute the set *R* of roles that are associated with a user. Step 4-8 determine whether access should be granted. The access policy choices need to be considered in these steps.

A cruicial component of the project is to provide the studens with a sample application that contains a very large set of roles and permission assignments. For a set of 10,000 permission assignments, for example, a find operation will not noticebly differ in respect to execution time if the permission assignments are stored in a linked list or a hash table. But for large sample applications students will experience long execution times if they choose an unsuitable data structure in their solution.

5 Extracting Individual Subprojects

For many courses, the scope of the proposed project will be too large. In addition, a course may not include all the topics covered by the project. We discuss how the project can be split into three individual smaller projects or assignments. Each assignment covers a particular topic. An instructor can choose an assignment for a particular topic without having to include the other assignments in the course. We discuss each of the three possible assignments or subprojects.

```
Input: userid usr, action act, data object dta
1. Use $U_{Roles}$ to determine the set $R_{usr}$ of roles of the user the
   userid usr.
2. Set $R = R_{usr}$.
3. For each role r in $R_{usr}$:
      Use order H to determine the set $R_r$ of roles that can be
      reached from r.
      Set $R = R \cup R_r$.
4. Set permission = false.
5. For each role r in R:
      If (the permission assignment (r, act, dta) exists)
         Set permission = true.
         Go to Step 5.
6. If (NOT permission AND  act == read AND
     WRITE_IMPLIES_READ)
      For each role r in R:
         If (the permission assignment (r, write, dta) exists)
            Set permission = true.
            Go to Step 6.
7. If (permission AND DENY_OVERRIDES)
      For each role r in R:
         If (permission assignment (r, deny, dta) exists)
            Set permission = false.
            Go to Step 8.
8. Return permission;
```

Fig. 3. Algorithm for the Execution Phase

Project 1: Hash Tables and Hash Functions

In this assignment, we consider a further simplified version of RBAC where deny actions are excluded. Thus each permission assignment is of the form (*role*, read, *data*) or (*role*, write, *data*) only. Further we assume that the policy WRITE_WITHOUT_READ applies. There is no need to specify how to handle deny actions since there are no permission assignments with the deny action. In addition, we omit the role hierarchy. In this case, a hash table is needed to store the roles associated with each user and a second hash table to store the permission assignments. For an access request (usr, act, dta), we need to determine all roles associated with the user usr by accessing the first hash table. Then, for each role rol associated with the user usr, we check whether the permission assignment (rol, act, dta) is present in the second hash table. The interesting task is the design of a hash function for both hash tables. We impose a specific format on the userids, which are the keys in the first hash table. For example, we may require that all userids start with the string “usr” and end with the string “01” or “02”. The students should recognize that the first three characters and the second last character do not help to distinguish individual userids. Hence these characters should not be used by the hash functions. In case of the hash function for the table with permission assignments, each key is a triple of the form (rol, act, dta). In this case folding has to be applied to construct an individual number that is then turned into a valid table index.

One can further reduce the assignment to a single hash table implementation by omitting roles. In this case, the first component of each permission assignment is a userid. For an access request (usr, act, dta), we need to determine whether the permission assignment (usr, act, dta) exists. If students are allowed to use an existing hash table class, as it is typically offered through the application programming interface of most common programming languages, then the project can be reduced to a small assignment, that requires designing and implementing a suitable hash function.

Project 2: Graphs, Partial Orders, and Warshall’s Algorithm or Trees and BFS

In the second project, a role hierarchy is added to the access control system. As discussed before, the role hierarchy H is a partial order. The relation is defined as follows: role r_1 relates to role r_2 if r_1 inherits all permissions that hold for role r_2. Project 2 demands to store the role hierarchy and to determine all roles that are related to a given role. Note that only the edges of the Hasse-like graph are given to specify the role hierarchy. As we need to determine the roles again and again, the assignment requires computing the transitive closure of the graph specified by the Hasse-like graph. Warshall’s algorithm can be used to determine the transitive closure.

Alternatively, the assignment can be modified to practice tree implementations and breadth-first search. In this case, only the edges of the Hasse-like graph are stored without computing the transitive closure. Each time a role r_0 is given, the roles r such that r_0 relates to r are calculated. These roles are exactly the roles r where a path from r_0 to r exists. Breadth-first search can be used to solve this problem.

If Project 1 has not been implemented, one can either provide a solution for Project 1 as a starter application for Project 2. In this case, one can simplify the mapping U_{Roles} of users to roles such that each user is assigned exactly on role. The students have to implement a method that returns all roles associated with a given user.

Project 3: Propositional Logic

In the last project, the different access policies are added. The user interface should be revised to allow entering the two relevant access policies: (1) WRITE_IMPLIES_READ or WRITE_WITHOUT_ READ and (2) PERMIT_OVERRIDES or DENY_OVERRIDES. In order to determine whether access should be granted, propositional logic needs to be applied. Students will need to implement the logic as displayed in Figure 3, for example.

If the students did not implement Project 1, a starter project should be provided that reads in the mapping U_{Roles} of users to roles and the permission assignments from a file. The starter project stores the data in an apropriate form. In addition, two methods should be provided: a method that returns all roles of a user and a method that determines whether a given permission assignment exists. In the assignment, students create a user interface to read in an access request of the form (*user*, *action*, *data*) and the access policies. Then they develop and implement the logic to determine whether the access request should be granted.

6 Summary

We have described a course project that is suitable for a data structures and algorithm course, a discrete structures course, or a discrete mathematics course for computer science students with a solid programming foundation. The project not only provides a context for the rather abstract material, but also provides a real-world application that students may encounter in the corporate world. Students can experience the importance of efficient data structures and algorithms and the proper application of discrete mathematics when testing their solutions on large input.

The project covers a range of data structures, including lists, hash tables, graphs or trees, and it integrates some topics of the discrete mathematics, like relations and functions. The project is not intended as a comprehensive coverage of all core topics of data structure and discrete mathematics specified in the draft of the Computer Science Curricular 2013 [1]. Indeed, this might be very difficult to accomplish. For example, the 2007 SIGCSE Committee Report on the Implementation of a Discrete Mathematics Course found that it was “not possible to develop a coherent package and cover all of the core topics” [18]. Thus we cannot expect to design a single course project that covers the majority of discrete mathematics concepts and most data structures that a computer science student should be exposed to.

The project can be used as a final course project, or it can be split into three subprojects or individual, independent assignments. The instructor can choose the level of support to be provided to the students. For example, the instructor may require and specify the data structure that needs to be applied, or the instructor may ask the students to select or design a data structure on their own.

The project or assignments assess the following learning objectives:

- Apply mathematical foundation to data structures, algorithms, and other concepts in computer science.
- Select an appropriate data structure and algorithm to solve a problem.
- Implement data structures and related algorithms.

Thus, the project can be used as an assessment tool to assess higher-level course outcomes and to evaluate the students’ performance.

7 References

[1] ACM/IEEE Joint Task Force on Computing Curricular (February 2013) Computer Science Curricula 2013, The Ironman Draft (Version 1.0). Available: http://ai.stanford.edu/users/sahami/CS2013//ironman-draft/cs2013-ironman-v1.0.pdf.

[2] T. Balch, J. Summet, D. Blank, D. Kumar, M. Guzdial, K. O'Hara, et al., "Designing Personal Robots for Education: Hardware, Software, and Curriculum," IEEE Pervasive Computing 7(2), 2008, pp. 5–9.

[3] J. D. Bayliss, "Using Games in introductory courses: tips from the trenches," in Proceedings of the ACM SIGCSE '09, 2009, pp. 337–341

[4] E. Bertino, P. Bonatti, and E. Ferrari, "TRBAC: A temporal role based access control model," ACM Transactions on Information and System Security, 4(3), 2001, pp. 191–233.

[5] I. Bezáková, J. E. Heliotis, and S. P. Strout, "Board Game Strategies in Computer Science," in Proceedings of the ACM SIGCSE '13, 2013, pp.17–22.

[6] P. Bonatti, C. Galdi, and D. Torres, "ERBAC: Event-Driven RBAC," in Proceedings of the ACM Symposium on Access Control Models nad Technologies, 2013.

[7] J. Coffey, "Integrating Theoreticaland Empirical Computer Science in a Data Structures Course," In Proceedings of the ACM SIGCSE '13, 2013, pp. 23–27.

[8] S. Cooper, W. Dann, and R. Pausch, "Teaching objects-first in introductory computer science," in Proceedings of the 34th SIGCSE technical symposium on Computer Science Education. Reno, Navada, USA, ACM, 2003, pp. 191–195.

[9] A. Erkan, T. Pfaff, J. Hamilton, and M. Rogers, "Sustainability Themed Problem Solving In Data Structures and Algorithms," In Proceedings of the ACM SIGCSE '12, 2012, pp. 9–14.

[10] European Commission (2013, February 20) "EU Data Protection: European Parliament's Industry committee backs uniform data protection rules," Press release, Brussels. Available: http://europa.eu/rapid/press-release_MEMO-13-124_en.htm.

[11] D. F. Ferraiolo, R. Sandhu, G. Serban, D. R. Kuhn, R. Chandramouli, "Proposed NIST standard for role-based access control," ACM Trans. on Information and System Security (TISSEC) 4(3), August 2001, pp. 224–274.

[12] M. Guzdial, "A Media Computation Course for Non-Majors," ACM SIGCSE Bulletin, 2003.

[13] M. Kölling, "The Greenfoot Programming Environment," ACM Transactions on Computing Education (TOCE) 10(4), 2010.

[14] M. Kölling and P. Henriksen, "Game programming inintroductory courses with direct state manipulation," In Proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education (ITiCSE '05), 2005, pp. 59–63.

[15] T. Lauwers and I. Nourbakhsh, "Designing the Finch: Creating a Robot Aligned to Computer Science Concepts," in Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10), 2010, pp.1902–1907.

[16] S. T. Leutenegger and J. Edgington, "A games first approach to teaching introductory programming," in Proceedings of the ACM SIGCSE '07, 2007, pp. 115–118.

[17] J. Maloney, M. Resnick, N. Rusk, B. Silverman, and E. Eastmond, "The Scratch Programming Language and Environment," ACM Transactions on Computing Education 10(4), November 2010.

[18] B. Marion, and D. Baldwin. (April 2007) SIGCSE Committee Report on the Implementation of a Discrete Mathematics Course. Available: http://www.sigcse.org/resources/documents/pdfs/DiscreteMathReport.pdf.

[19] K. McMaster, N. Anderson, and B. Rague, "Discrete Math with Programming: Better Together," in Proceedings of the ACM SE '07, 2007, pp. 100–104.

[20] N. Neff, "Problem-Directed Discrete Structures Course," in Proceedings of the ACM SIGCSE '10, 2010, pp. 148–151.

[21] Q. Ni, E. Bertino, J. Lobo, C. Brodie, C. Karat, J. Karat, and A. Trombetta, "Privacy-aware role-based access control," ACM Trans. Inf. Syst. Secur. 13(3), July 2010, pp. 1–31.

[22] A. Oxley, "Discrete Mathematics and its Applications," Teaching Mathematics and Its Applications: An International Journal of the IMA 29, no. 3, 2010, pp. 155–163.

[23] M. Sahami, "A Course on Probability Theory for Computer Scientists," in Proceedings of the ACM SIGCSE '11, 2011, pp. 263–268.

[24] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-based access control models," IEEE Computer Society 29(2), February 1996, pp. 38–47.

[25] M. Savin-Baden, *Facilitating Problem-based Learning*. McGraw-Hill, Berkshire, UK, 2003.

[26] M. Savin-Baden and C. H. Major, *Foundations of Problem-Based Learning*. McGraw-Hill, Berkshire, UK, 2004.

[27] D. L. Schuster, "CS1, arcade games and the free java book," in Proceedings of the ACM SIGCSE '10, 2010, pp. 549–553.

[28] K. Sung, M. Panitz, S. A. Wallace, R. Anderson, and J. Nordlinger, "Game-themed programming assignments: the faculty perspective," in Proceedings of the ACM SIGCSE '08, 2008, pp. 300–304.

Implementing Strategies To Improve A Video Game Programming Course

Antonio Javier Cavazos Mejorado
ITESM – Monterrey Campus, México

Summary

This paper shows the interest on the students to develop videogames in object-oriented programming and the way they were motivated.

The results were obtained through the use of strategies developed during different semesters in a videogame class.

The improvements made to have a better course evaluation indicator during five years.

This specific course is a project to develop a computer application using different concepts in mathematics, programming and physics of the three previous semesters in which the subject is taught.

This is a difficult course because the students must integrate their previous knowledge and they require individual and team preparation and several tools to collaborate effectively.

The purpose of this paper is to show how the students can be motivated to program when the project is a videogame and others recognize their work out of the classroom.

1. Introduction

This paper shows the changes made in a professional course in the area of Computer Science at the ITESM campus Monterrey Mexico, for several years, and the way students were motivated to develop a computer application in an object oriented programming paradigm.

The course is a software development project baseline in Java, with students of fourth semester of Computer Science major. This course has to integrate the knowledge gained in the three previous semesters in different materials for the development of an application in a collaborative way.

This work serves as an example of experience, for those professors whose want to teach a software development project in a colaborative way.

2. Background

Before the fourth semester project, the students get some knowledge from mathematics, physics, programming and user interface, between som other subjects. The main goal of this course is to integrate knowledge of these materials (see example Annex 1: Map of career ITC 05, ITESM).

When I was assigned for this class, I thought that a videogame could be the best option for the students to integrate their prevoius knowledge in mathematics, physics, programming and user interface.

The first thing I did was to prepare the analythic program and the activities to develop during the semester.

Below are the changes defined year by year for this class and the course acceptance indicator (CAI) from students .

3. Development

3.1 First year

This year the students were asked to develop three month small projects, besides the final project, which made the course very difficult for them and for me, counseling hours were about 10 hours per section, having projects with different game objectives, in addition to the monthly projects.

The idea of the course at the time was that the students did research, with basic explanations given by the teacher, which was actually a lot more work, because it required more assistance outside the classroom.

All work was performed in teams of 3-4 people maximum, individual assessment to students was minimal, as there were quizzes that were assigned to check readings.

Furthermore students used drawings designs without copyrigth, and although this was an academic project, that was wrong , they had to have approval of the use of the designs so that they can include them in a videogame. At the end of the semester the CAI obtained from both groups was 1.90 and 1.85[1]

At the end of the first term several books were gotten for in-class support: "Developing Games in Java" [1], which has been so helpful for this course. Also got the book "Beginning Java 5 Game Programming" [2], which is more basic, but also served a great help, especially for the initial course.

For the second term, there was a single group, having more lecture during class, but definitely out of class counseling remained high, 12 hours a

[1] CAI indicator range varies from 1 for excellent to 5 for bad

week of counseling for these students belonging to a single section. At the end of the term the section CAI was 1.7

Another important goal besides the main objective of the class is the use of well recognized documentation techniques, we used java documentation “javadoc” [3] and the Java code convention [4] as well as a good internal documentation.

At the end of the term the teams have a final presentation with some professors, students, the dean of the major and a professor recognized for each team as a “team godfather” all of them as an audience. (See Annex 2: Presentation of Final Papers)

The motivation of the semester students to the final project showed an exponential increase, it was a pleasure to see how their work exhibited and recognized their effort and dedication, through colleagues in the area, as teachers from other areas , sponsors and the dean of the major.

Students also deliver technical documentation of the project, delivery of the game made like commercial dvd in a box with advertising (see Annex 3: Example of a Final Project and Game), one of the “team godfathers” coments was: "this is amazing is the same videogame I buy for my grandchildren".

3.2 Second year

The second year there were major changes in the course, among which there were only two monthly projects and the final project, students had more time for the final project, the time of counseling per week down to 7 hours per section and between different actions now emphasize support tools for teamwork and explanations focused on developing work, it was still difficult for the teacher to handle the advice so varied, every project, every idea that he was difficult to manage in the time of counseling, it was difficult to find fault, the time it took to focus on a project, review errors and then switch to another project, more difficult to advice, especially by different programming styles.

The use of godparents was important to recognize the work of students at the end of the semester, but otherwise students began to have trouble getting teachers who could be free within hours they had assigned the review of the course to attend the presentation of the final work.

At the end of the first term CAI was obtained as 1.62 and 1.67 in the two sections and 1.88 semester the second term there was only one group.

One of the comments from students was that they wanted to be given more classes and less research, but to expose their level of work satisfaction was very large because they felt they had advanced so far that they had developed a product that initially did not expect.

An ongoing project is very complicated, especially when students have not had an approach to managing a project before, when they are not using the same development tools, and some of them have better programming skills than others, and even harder when they have not developed a complete product and no prior art has an abstract programming language.

The challenge of this class is to provide guidance in planning, proposing collaborative tools, provide guidelines for project management and programming advice at all times.

Students in this course are facing what they will experience at the end of his career, such as making decisions at all times, working together in a collaborative way, and researching to get better programming solutions.

During the summer the professor developed a series of tutorials for this course, word documents with explanations and code pieces to try.

The purpose of these tutorials was to get students with basic programming skills get an incemental knowledge in videogame programming.

3.3 Third year

This year the students used the tutorials developed to program videogames for applets and frames.

The dinamic of the work changed and the students had individual and team work, 8 weeks to work with individual assingments and 6 weeks to the final project.

The professor lectures in class using the tutorials and there was a sense of satisfaction in students feeling more guided .

For the second term the professor used a game enging and framework template [5] for the the students to develop the videogame, having the game more standarized.

In that year the CAI down significantly from previous years in August 2008 having 1.59 and 1.16 in the two groups and in Jan 2009 semester 1.25 and 1.41 in the other two groups.

Unquestionably the CAI had dropped considerably, from the first year of delivery of the subject.

The tradition of the final project was a team, students had to be dressed formally for presentation, all equipment must be present and in the end was typical session of questions and answers and could not miss sandwiches and soda to celebrate success of the work.

The teacher felt more relaxed with the development of the class, but I saw that some students were using some designs that were not public domain or taking some designs and adjusted without permission from the authors, on the other hand saw that time was much investing in developing these designs and had even sent teams to make their own designs and it was decided to recruit students from other disciplines to develop scenarios designs and drawings for use by students in this class.

3.4 Fourth year

When you start the semester in August 2009 further changes were made to the previously defined between the first changes was to establish specific requirements for equipment so the game that everyone should have developed a specific feature, such as a disability support, teach feeding, teaching mathematics, safety, among others.

Another change that was experienced was that students had to sign a confidentiality agreement and assignment of the product.

The idea was to make the students feel in a real company to be working in the classroom and to think about the importance of their work for the development of applications that may be at the level of many applications that are sold.

Most agreed to this and they all signed. Another important change was that the teacher and director of the company was only with the team leaders, ie from the beginning of the project development, any doubt that the team had on the development of the project was the only leader conversed with the teacher.

Added to this, it is suggested the game's development in particular the selection of characters and settings, so that each team should focus the requirement and the resources provided for the development of the game in particular.

This semester was peculiar, because there was a small group of 15 students and decided to innovate with the development of video games but in the cell, which meant a lot more work for teachers and students.

Working with these students was higher, as they had to investigate in order to be prepared before going to classes, students worked more than all those who have taken this matter, since they saw the game development Applets in MIDlets Frames and finally, to make the game on the phone.

The motivation of this group was very large, since the greater incentive for them to bring in his cell was an application developed by themselves.

It was based on the book "Creating Mobile Games" [6] and developed tutorials to support students from the basics.

Presenting the final draft of this group, the students showed much more motivation than the rest.

By January 2010 some changes were implemented due to feedback from surveys of students, including Deleted the signing of confidentiality agreements and the sale of the rights to work, as there were several negative comments about it, where did see that they did not want to feel so pressured and were not yet working in an enterprise.

In August 2009 CAI was obtained 1.46, 1.40 and 1.20 in the three groups that took the field and in January 2010 was obtained 1.56 and 1.93.

The generation of January 2010 had been a difficult generation, discussing this with colleagues and found that there was little consistency between OMB and comments so high because all the comments were very positive.

It is noteworthy that during all semesters that this matter has offered students receive timely feedback on assignments and tests by the platform and its partial and final qualification can be found there before it is published in the official grading platform They already know their total weights, this has been one of the objectives of the teacher, make the most of the facilities offered by the Institute's official platform.

During the last semester of the fourth year of offering the subject the teacher creates a Web page for the query in the tutorials and the use of resources focused designs directly serve students in the class, with its own resources and the desire a higher level of recognition of student work. It would page sections: 1. Tutorials, 2. Resources 3. Gallery Games.

3.5 Fifth year

The first change experienced by August 2010 was the use of the site for the development of individual and team work [7], which caused them a lot of admiration for the students.

On the other hand experienced a very clear easy explanations and clarifications as each tutorial references were much more agile and able to observe the tutorials applications running well, as they are immersed in the same page, this helped a lot more students.

A second change was the students remove the stress of getting a sponsor for presentation because they knew of the existence of this requirement.

A third change and I think this was beneficial, was the mention of a "management" as responsible for the work entrusted to them, where the teacher only was responsible for a department and the students who work for him would be support for provide better service.

The management is defining the changes are for students to develop their proposals and awards himself if there is management who defines this.

Students were always very attentive to these changes and more interested in knowing who formed the management.

The last change came to the end of the semester and found it much more motivating, is that the teacher started up the previous semesters games to the Web, so that students see on the page in the gallery section games, team picture and can use the game developed by the team.

This is something that I really liked, because now they could present the work developed in team quality, showing through the Web, so that if a student is interviewed by Microsoft, Google, or any other company that wants to hire , has a very fast way to make this reference.

The CAI obtained in this semester for the three groups that offered matter was 1.14, 1.00 and 1.17 in the three groups in which the subject is offered.

4. Future Changes

Among the changes you want to implement in the future are:

• testimonials from students who are already working and that passed through the development of this project,

• real users, businesses or homes have some games that support game development for real

• competitions, so as to reward the best games.

5. Conclusions

The improve opportunities for student learning in the development of a class is difficult, even more when it comes to a project, but is more challenging precisely the power to do it a way that is innovative.

It is important to use templates, formats, standards and tools that support the entire software development in a collaborative way, we must be in continuous research to spend less time on what is not productive.

Learning in teaching is a very rewarding process that makes them see the continuous improvement process of a course as a more agile.

The motivation for the student is the most important factor in the learning process and their satisfaction, we must be alert to these issues in order to enhance learning.

Definitely changes developed in a class for the continuous improvement process significantly affect the way in which the students realize the service delivery of the course and thus affects the teacher's evaluation.

The design initially defined for this class, consider activities that helped to solve problems in isolation, so that students have the practice of dealing with a problem and a solution, and then develop a solution to a problem posed by the end of the project class, among the changes were defined problems aimed at better understanding of the concepts required for the development of the final project.

Definitely the recognition of the efforts of the students of this class is something that has impacted more acceptance, because the work they do in class Project Problem Solving with Programming is another project is the third project , which gives a great idea of what they will do in their professional lives and can access it from anywhere, as I have it on the internet and be a pride for them, for the teacher and the institution.

References

[1] Brackeen, D. (2003). Developing Games in Java. USA: New Riders Pub

[2] Harbour, Jonathan S. (2006). Beginning Java 5 Game Programming. USA: Thomson. Course Technology.

[3] How to Write Doc Comments for Javadoc Tool Available: http://www.oracle.com/technetwork/java/javase/documentation/index-137868.html

[4] Code Conventions for the Java Programming Language. Available at: http://www.oracle.com/technetwork/java/codeconv-138413.html

[5] Java Game Programming Tutorial - Game Engine & Framework. Available at: http://www3.ntu.edu.sg/home/ehchua/programming/java/J8d_Game_Framework.html

[6] Hamer, C. (2007). Creating Mobile Games: Using Java ME Platform to Put the Fun into Your Mobile Device and Cell Phone (Technology in Action). USA: Technology in Action Press.

[7] Playing with Java http://www.jugandoconjava.co

A Monitored Student Testing Application Using Cloud Computing

R. Mullapudi and G. Hsieh
Department of Computer Science, Norfolk State University, Norfolk, Virginia, USA
r.mullapudi@spartans.nsu.edu, ghsieh@nsu.edu

Abstract – *This paper presents the design and implementation of a prototype online monitored testing application which has been deployed in the Amazon Web Services Cloud using Amazon's Elastic Compute Cloud platform. The interactive application allows students to take tests online for their courses while the instructors are logged in simultaneously and conduct the tests. The application allows the students to request hints from the responsible instructor during the test and also enables the instructor to provide the hints if so desired. The prototype application was developed using Apache Tomcat web server, Adobe Flash Builder for the user front-ends, and MySQL for the back-end database. The application logic was implemented primarily in Java 1.6 using Eclipse IDE for Java EE Developers. The application was developed and runs on Microsoft Windows platforms.*

Keywords: cloud computing; online testing; monitored testing

1 Introduction

To enhance the teaching and student learning in information technology and related computing disciplines, especially in gateway or remedial courses, it is important to conduct quizzes, tests, and exams frequently and meaningfully to gain insights into students' achieving the desired learning outcomes or not, on both collective and individual basis.

Online testing applications and systems are becoming increasingly widely used due to the major benefit of convenience that they bring to the participants. Students can take the tests from anywhere with internet access. Instructors can schedule the tests at times that are convenient to the students. The convenience factor is especially important for distance-learning or online courses.

The conventional online testing applications generally do not provide the instructors with capability to control the pace of the test in real time. Also they generally do not allow the students to ask for help from the instructors, or the instructors to provide appropriate help during the tests interactively. An instructor generally finds out the performance of a student on a test only after the test is complete, without a great deal of insight as to the difficulties or problems that prevented the student from being able to correctly answer a question. Hence, these testing applications are useful for evaluation or assessment purposes. However, they do not generally provide a collaborative problem-solving capability between an instructor and a student during a scheduled test.

The monitored student testing application described in this paper attempts to add such a collaborative problem-solving capability into online testing applications by allowing the students to ask for help and the responsible instructor to provide hints if so desired, all during the testing interactively.

In addition, cloud computing (1) is gaining momentum as a critical technological and business solution in both public and private sectors, including the education and training segments. The major business benefits of cloud computing include: business agility, new business models, less operational issues, better use of resources, and less capital expense (2). The application described in this paper is designed to take advantage of the benefits offered by cloud computing.

The initial design and prototype implementation of such a monitored student testing application was completed through a M.S. Project research (3) at Norfolk State University.

The prototype application was developed using Apache Tomcat web server (4), Adobe Flash Builder (5) for the user front-ends, and MySQL (6) for the back-end database. The application logic was implemented primarily in Java 1.6 using Eclipse IDE for Java EE Developers (7). The application was developed and runs on Microsoft Windows platforms.

Once the application was developed and tested on a local Windows 7 Enterprise system, it was then deployed to the Amazon Elastic Compute Cloud (EC2) platform (8) which is a product offered by Amazon Web Services (AWS) (9). We chose to use the AWS because it is a low-cost public cloud service which offers a high level of flexibility and performance. We chose to use the EC2 because it is a well-known service that provides the application developers with complete control of their computing resources and lets the developed applications run on Amazon's proven computing environment. Furthermore, the AWS offers an attractive Free Tier option that is very useful for prototype development and deployment purposes. For this research project, we used the AWS Free Tier service.

This paper is organized as follows. Section 2 discusses the user scenarios for the initial design and prototype implementation of a monitored student testing application.

Section 3 presents the high-level design of the prototype application. Section 4 describes how the prototype application was developed and deployed on a local system. Section 5 discusses how the prototype application was deployed to the EC2 platform in AWS cloud so it could be accessed from outside of the cloud. A summary and discussion on future work are presented in Section 6.

2 User Scenarios

A new user needs to first register with the application as a Professor or a Student. A registered Professor can create test papers each containing a list of questions and correct answers. The Professor then specifies the date/time for the test. Upon the scheduled time for a test, the responsible Professor can log into the application, start the test, monitor the progress of the test, and supply the hints/suggestions if requested by students and the Professor deems it appropriate to do so. The Professor also determines when to post the next question to all participating Students. The Professor can also track the students' performance as the test progresses.

A registered Student can log into the application to find out what tests are scheduled. At the scheduled time for a test, the Student can log in and take the test as each question is posted by the responsible Professor. If the Student needs help, he/she can ask the Professor for hints in real time. A Student can also track his/her own performance and the Student's standing in comparison to the class as the testing progresses.

3 Software Design

This section presents an overview of the software design and components used for the prototype monitored student testing application that has been deployed on AWS EC2 platform.

3.1 Required Software

The off-the-shelf software packages required for the development and deployment of the prototype monitored student testing application are:

1) Adobe Flash Builder 4.5/4.7;
2) Eclipse IDE for Java EE Developers;
3) Apache Tomcat 6.0 web server;
4) MySQL 5.1 database server; and
5) MySQL 5.1 GUI Tools.

3.2 Application Software Architecture

To provide the desired level of performance, maintainability, and reusability, the best practices for developing web applications call for using a three-tiered architecture consisting of client, application server, and back-end databases.

For our application, we choose Apache Tomcat as the Web Application Server and MySQL as the relational database management system. Both are open source and high performance implementations which have been widely used for development and deployment worldwide.

With this architecture, user requests are transmitted by the client to Apache Tomcat Web Application Server which processes the user requests based on the application logic implemented by the server. The Apache Tomcat server also accesses the MySQL databases to store, modify, or retrieve data.

The responses to user requests are transmitted from the Apache Tomcat server to the client that in turn presents the results to the users.

Figure 1 depicts the software architecture of the application as viewed by the key components: Professors, Students, MySQL databases, and Apache Tomcat application logic.

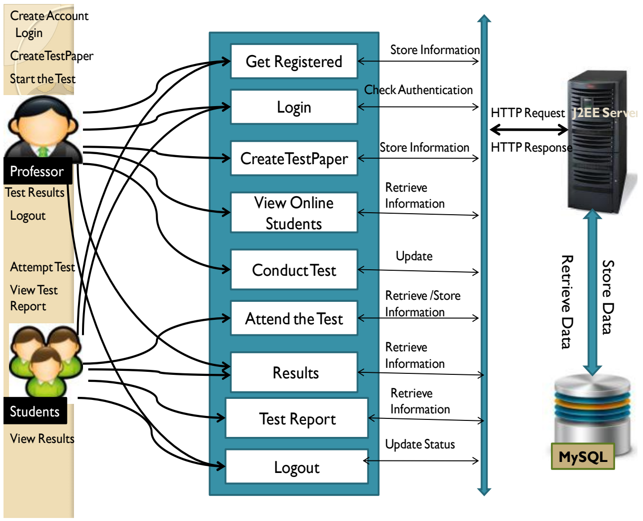


Figure 1. Application Software Architecture

3.3 UML Representations

Unified Modeling Language (UML) is a standardized specification language for object modeling (10). UML is a general-purpose modeling language that includes a graphical notation used to create an abstract model of a system, referred to as a UML model. It's a refinement of earlier Object Oriented Design and Object Oriented Analysis methodologies. These diagrams can be divided into two groups: Structural Diagrams, which model the organization of the solution, and Behavioral Diagrams, which model the functionality of the solution.

3.3.1 Class Diagram

Class diagrams are the structural diagrams used to identify the class structure of a system, including the properties and methods of each class. Also depicted are the various relationships that can exist between classes, such as an inheritance relationship. The class diagram is one of the most widely used diagrams from the UML specification.

Figure 2 depicts the class diagram for two classes: Professor and Student for our application.

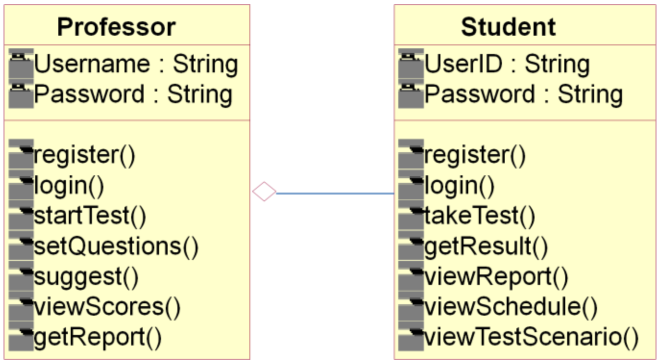


Figure 2. Class Diagram

3.3.2 Use Case Diagrams

Use case diagrams are Behavior Diagrams and are used to identify the functionality provided by the system (use cases), the users who interact with the system (actors), and the association between the users and the functionality. Use cases are used in the analysis phase of software development to articulate the high-level requirements of the system. The primary goals of use case diagrams include:

1) Providing a high-level view of what the system does;
2) Identifying the users ("actors") of the system; and
3) Determining areas needing human-computer interfaces.

Figure 3 shows the use case diagram for the Professor and Figure 4 shows the use case diagram for the Student.

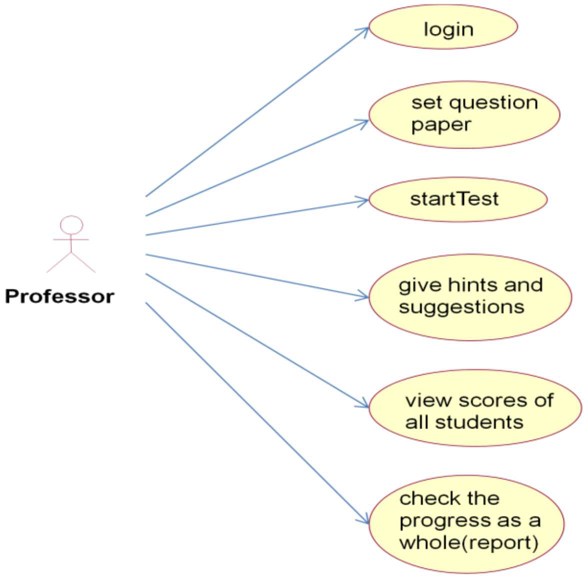

Professor
login
set question paper
startTest
give hints and suggestions
view scores of all students
check the progress as a whole(report)

Figure 3. Use Case Diagram for Professor

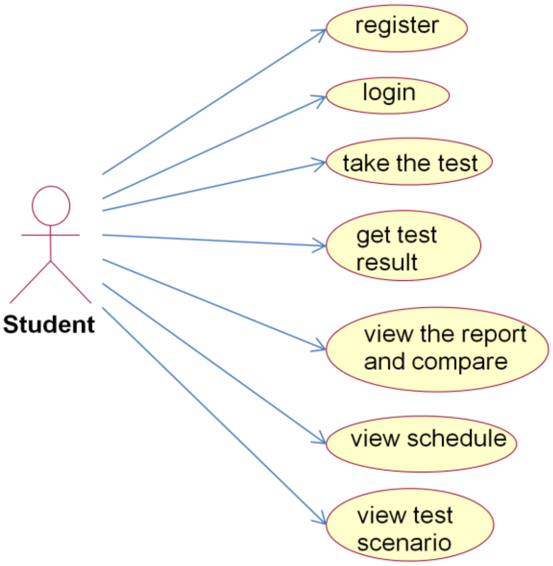


Figure 4. Use Case Diagram for Student

3.3.3 Sequence Diagram

Sequence diagrams are also Behavior Diagrams and are used to document the interactions between classes to achieve a result, such as a use case. The sequence diagram lists objects horizontally, and time vertically, and models these messages over time.

Figure 5 is the sequence diagram depicting the interactions among the key components of our application.

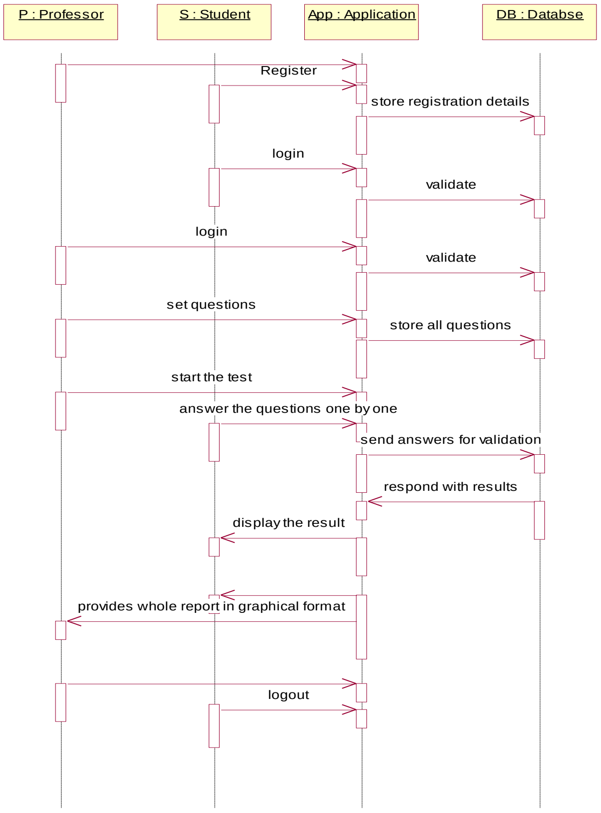


Figure 5. Sequence Diagram for Professor and Student

3.3.4 Collaboration Diagram

Collaboration diagrams are also Behavior Diagrams and are designed for modeling the interactions between objects. This type of diagram is a cross between an object diagram and a sequence diagram. It uses a free-form arrangement of objects which makes it easier to see all iterations involving a particular object.

Figure 6 is the collaboration diagram depicting the interactions among the key components of our application as a sequence of events and messages.

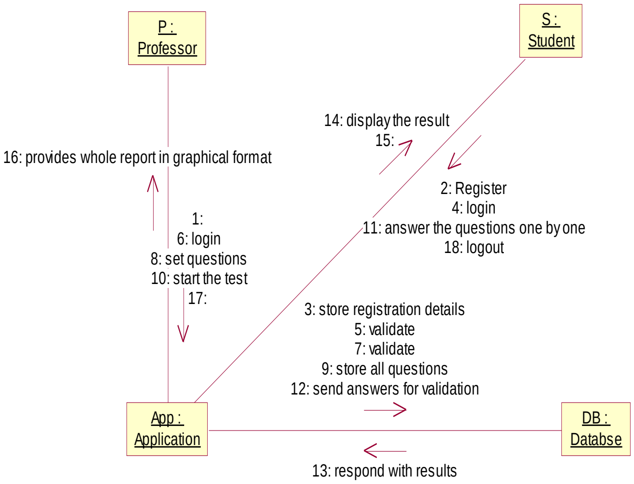


Figure 6. Collaboration Diagram

3.4 Database Tables

In order to execute the project, several MySQL tables are created to store the relevant information:

1) Student Details table for storing information for each Student registered in the system.
2) Professor Details table for storing information for each Professor registered in the system.
3) Exam Schedule table for storing information for each exam created and scheduled by a Professor.
4) Test Papers table for storing the questions and correct answers for each exam created by a Professor.
5) Test Scenario table for storing the status of an exam and the Professor and Students who are participating in an exam.
6) Student Questions table for storing the status of each Student's progress on answering the questions in an exam.
7) Student Result table for storing the answers given by each Student to the questions in an exam.
8) Total Results table for storing the results for all students taking an exam.
9) Student Marks table for storing the scores for each Student.

3.5 User Front-Ends

To develop the GUI front-ends with cross-platform support, we choose to use Adobe Flash Builder (Version 4.5 and 4.7) which is a popular Integrated Development Environment (IDE) for building games and applications. The Flash Builder supports the ActionScript® language and the open source Flex framework (11). It also supports MXML which is an XML-based user interface markup language first introduced by Macromedia in March 2004.

We use MXML to declaratively lay out the GUI interface of our application, provide tags that correspond to classes in the Flex framework, and implement the foundational logic and application behaviors. We also use ActionScript-based scripts embedded in MXML files to provide flow control and object manipulation features that are not available in MXML. The Flash Builder IDE is used to compile the MXML into standard binary SWF files which can then be used by Adobe Flash Player. The Flash Player is "a web browser plug-in that delivers expressive cross-platform, browser-based applications, content, and video that work consistently across browsers on a broad range of devices, including smartphones, tablets, personal computers, and TVs." (12)

4 Prototype Application Development

The prototype application was developed on Microsoft Windows platform using the Eclipse IDE for Java EE Developers (Juno release). The developed code was first installed and tested on a local Windows platform (Windows 7 Enterprise version). Once proven operational in a local environment, the developed code was then installed and tested on a Windows Server 2008 virtual machine running on Amazon's EC2 platform.

4.1 Software Modules

The prototype software is organized into four modules:

1) Users Module which provides the functionality to authenticate and register the users (Professors and Students) who wish to use the application.
2) Professor Module which provides the functionality for a Professor to create and schedule a test, log in at the scheduled test time, conduct the test, give hints and suggestions to students who ask for help, view test scores, and compare a student's test scores with other students in the class.
3) Student Module which provides the functionality for students to find out test schedules, take a test, ask for hints if so desired, and view the individual's test scores in graphical form.
4) Cloud Module which supports the deployment of the entire application on a cloud and the ongoing monitoring of its operation in the cloud.

4.2 Local Development

There are five major steps involved in developing, deploying, and using the prototype application in a local environment (on a Windows 7 Enterprise system):

1) Set up the system.
2) Develop the application.

3) Deploy the application.
4) Run the application.
5) Access the application locally.

4.2.1 System setup

To set up the local development and execution environments, five major software packages are required:

1) Install Eclipse IDE for Java EE Developers.
2) Install Adobe Flash Builder 4.7 Premium release.
3) Install MySQL Server (5.1.67).
4) Install MySQL GUI Tools (5.0.r17).
5) Install Apache Tomcat Server (6.0.36).

4.2.2 Application development

The main application logic is implemented primarily in a Java program called MonitoredTestingService, using the Eclipse IDE. The GUI front end is implemented in MXML and ActionScript code using the Flash Builder IDE. The database tables for the local MySQL Server are created using SQL scripts and MySQL GUI tools.

4.2.3 Application deployment

The following major steps are used to deploy the application software on a local Windows 7 system:

1) Create a local Apache Tomcat v6.0 server in Eclipse's workspace using Eclipse IDE.
2) Start the local Apache Tomcat v6.0 Server from within the Eclipse IDE.
3) Deploy the application MonitoredTestingService onto the local Apache Tomcat v6.0 Server within the Eclipse workspace.
4) Deploy the bin-release code produced by the Flash Builder IDE for this application, at the same location for the main application MonitoredTestingService, onto the local Apache Tomcat Server within the Eclipse workspace.
5) Configure the local Apache Tomcat Server within the Eclipse workspace for a different TCP port number (using 9090 instead of the default value of 8080).
6) Configure the local MySQL Server and create the database tables required for the application.

4.2.4 Application execution

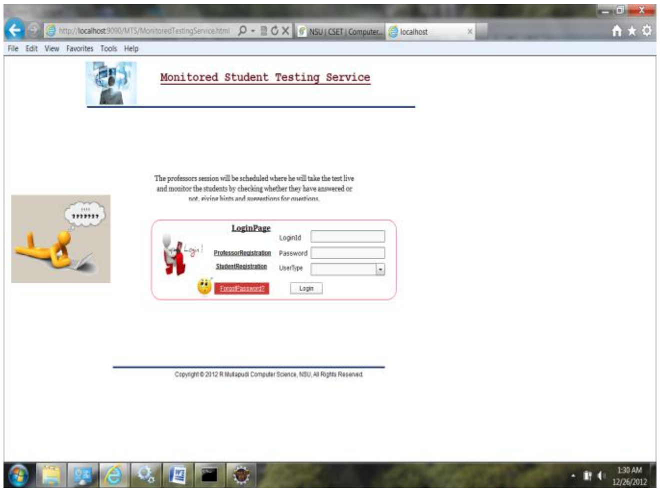
To run the application, the following major software components need to be running:

1) The local MySQL Server with the database tables appropriately created and initialized for the application.
2) The local Apache Tomcat Server within the Eclipse workspace with the required application software deployed and configuration changes made.

4.2.5 Application access

To access the application locally, start a web browser (e.g., Microsoft Internet Explorer) with the URL set to: http://localhost:9090/MTS/MonitoredTestingService.html.

This will bring up the "Monitored Student Testing Service" homepage with a LoginPage dialog box at the center of the page, as illustrated in Figure 7 below.



Monitored Student Testing Service

The professors session will be scheduled where he will take the test live and monitor the students by checking whether they have answered or not, giving hints and suggestions for questions.

LoginPage

LoginId

Password

UserType

Copyright © 2012 R Mullapudi Computer Science, NSU, All Rights Reserved

Figure 7. Homepage Display for Local Access

5 Cloud Deployment

Once the application had been developed and tested on a local Windows 7 Enterprise system, the application was next deployed to the Amazon EC2 cloud.

The major steps in deploying and using the application in Amazon EC2 cloud include:

1) Set up an EC2 Instance pre-loaded with a standard Windows Server 2008 R2 operating system.
2) Set up the EC2 Instance with the prerequisite software.
3) Deploy the application onto the EC2 Instance.
4) Run the application on the EC2 Instance.
5) Access the application using a web browser executed on the EC2 Instance.
6) Access the application using a web browser executed on a remote system.

5.1 Set up Cloud Instance

To simplify the set up process, we choose to load the instance with a standard Windows Server 2008 R2 operating system.

The security policy for the EC2 instance needs to be modified to permit the following TCP ports through the built-in firewall for the EC2 instance: HTTP, MYSQL, RDP (for remote desktop), and 9090 (for Apache Tomcat server).

Next, a RDP connection is set up between a remote Windows 7 system and the EC2 instance running Windows Server 2008 R2 operating system. The RDP connection is used for deploying the application onto the EC2 instance and administering/monitoring the application subsequently.

For the ease of application deployment and future maintenance, we choose to install all the pre-requisite software on the EC2 instance: Eclipse IDE for Java EE Developers; Adobe Flash Builder 4.7; MySQL Server and MySQL GUI Tools; and Apache Tomcat Server.

Next, copy the application software onto the EC2 instance. Then, deploy and run the application on the EC2 instance as described in the previous section.

5.2 Accessing Application in Cloud

Once the application is deployed onto the EC2 instance, we first test the application by accessing it through a web browser session launched from the EC2 instance.

Next we access the application from a web browser launched on a remote system with the URL set to: http://*x*:9090/MTS/MonitoredTestingService.html, where x is the public DNS address (e.g., ec2-184-73-116-127.compute-1.amazonaws.com) assigned to the EC2 instance. Figure 8 below shows the homepage displayed for remote access to the EC2 instance.

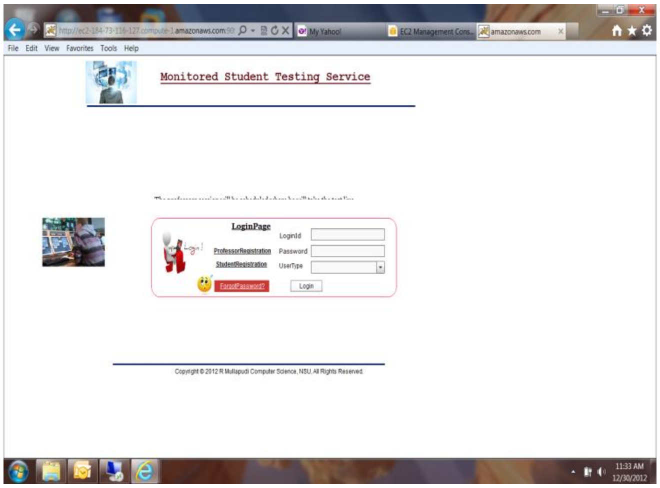

http://ec2-184-73-116-127.compute-1.amazonaws.com:90
My Yahoo!
EC2 Management Cons...
amazonaws.com
File Edit View Favorites Tools Help
Monitored Student Testing Service
LoginPage
Login!
LoginId
ProfessorRegistration
Password
StudentRegistration
UserType
ForgotPassword?
Login
Copyright © 2012 R Mullapudi Computer Science, NSU, All Rights Reserved.
11:33 AM
12/30/2012

Figure 8. Homepage Display for Remote Access

6 Summary and Future Work

This paper presents the design and prototype implementation of a cloud-based monitored student testing application.

This application is designed to allow the instructors to conduct and monitor the tests in real-time. It also allows the students to ask the instructors for hints during the tests, and the professors to supply hints or suggestions if they deem appropriate to do so. The pace of each test is also controlled by the responsible instructor in real-time based on the performance of the students as they answer each question. A student can also display in real-time his/her test score and its standing among the student's cohorts.

This monitored approach for student testing can be very useful for undergraduate gateway or remedial courses which are critical for laying the foundation for students' success in more advanced courses down the road. It is very important for the instructors in these courses to make sure that students learn the required topics successfully. The traditional online testing approach does not provide the instructors real-time control of the pace of the test, or a mechanism for students to ask for and receive hints from the instructor in real-time. This type of real-time interaction between the instructor and the students can be very useful when the instructor wants to make sure that a certain percentage of students who are taking the test have correctly answered a question.

With a cloud-based implementation, this type of application can offer a high degree of convenience for instructors and students as they can work from anywhere. It can also leverage the benefits of cloud computing in cost efficiency and improved availability.

A tradeoff of this monitored approach is that the responsible instructor needs to be online and conduct the testing in an active manner. This is different from the traditional online testing approach which does not require the instructor to be online and actively involved with the testing during the time the testing is scheduled.

Another constraint of the initial prototype implementation of the monitored student testing application is that all students taking the same test are working on the same questions at the same pace. This may be too restrictive for courses with students of very diverse abilities and preparations. In the future versions of the prototype implementation, we plan to allow students to move forward at their own pace, while still allowing the instructors to actively monitor the performance of students and provide hints when requested by students interactively during the testing.

The initial prototype implementation of the application only supports the questions and answers in simple text and multiple choice formats. We plan to add capabilities to support questions and answers in more flexible and multimedia formats.

We also plan to conduct experiments with a small number of gateway and remedial courses in the computer science and information technology curricula to collect feedback from the participating instructors and students on the efficacy of this monitored student testing approach.

7 Acknowledgement

G. Hsieh's research was supported in part by U.S. Army Research Office, under contract no. W911NF-12-1-0081, and U.S. Department of Energy, under grant no. DE-FG52-09NA29516/A000.

8 References

1. Mell, Peter and Grance, Timothy. *The NIST Definition of Cloud Computing.* NIST. 2011. NIST Special Publication 800-145.

2. Schouten, Edwin. 5 Cloud Business Benefits. *Wired.* October 16, 2012.

3. Mullapudi, RamaKrishna. *Monitored Student Testing Service using Cloud Computing.* Department of Computer Science, Norfolk State University. Norfolk, Virginia, 2013. MS Project.

4. Apache Tomcat. [Online] The Apache Software Foundation. [Cited: April 5, 2013.] http://tomcat.apache.org/.

5. Adobe Flash Builder Family. [Online] Adobe Systems Inc. [Cited: April 5, 2013.] http://www.adobe.com/products/flash-builder.html.

6. *MySQL.* [Online] [Cited: April 5, 2013.] http://www.mysql.com/.

7. *Eclipse Downloads.* [Online] Eclipse Foundation. [Cited: April 5, 2013.] http://www.eclipse.org/downloads/.

8. *Amazon Elastic Compute Cloud (Amazon EC2).* [Online] Amazon.com Inc. [Cited: April 5, 2013.] http://aws.amazon.com/ec2/.

9. *Amazon Web Services.* [Online] Amazon.com Inc. [Cited: April 5, 2013.] http://aws.amazon.com/.

10. *Unified Modeling Language.* [Online] Object Management Group. [Cited: April 5, 2013.] http://www.uml.org/.

11. Flex. [Online] Adobe Systems Inc. [Cited: April 5, 2013.] http://www.adobe.com/products/flex.html.

12. Adobe Flash runtimes. [Online] Adobe Systems Inc. [Cited: April 5, 2013.] http://www.adobe.com/products/flashplayer.html.

Research of Computational Thinking-driven Teaching and Innovative Practice Pattern

Zhanquan Wang, Jiang Liu, Chunhua Gu,Qingchun Hu,Xinxiu Wen
Computer Teaching & Experiment Center, East China University of Science and Technology, Shanghai, China

Abstract -*With the improvement of computer technology, the previous methods of teaching and innovation practice aren't conformable in current information age . It is difficult for students to understand computational thinking (CT) and improve their innovation ability. The study propose a new pattern of computer course system based on computational thinking, which consists of modules, including computational thinking module, foundation teaching module, innovation practice module, ACM train module and information literacture module. The results of the study indicate that the new teaching system is very effective and validate. Student engaging in this study achieve good performance in the application ability contest of the computer in China and Shanghai city.*

Keywords: Computational Thinking; Iinnovative Module; Information Platform;

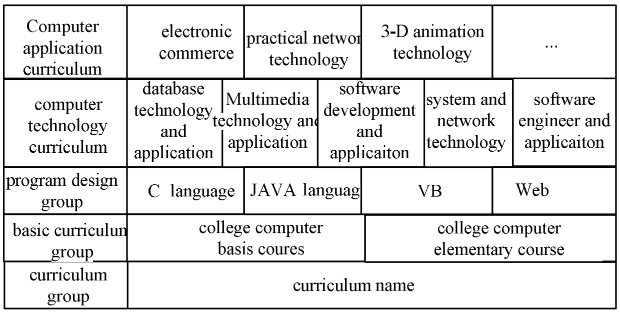
1 Introduction

computer technology and computer knowledge is becoming increasingly popular. For the instructor, how to make the students use computer to create the professional product, how to popularize computational thinking and how to make students truly use computer to accelerate the cultivation of innovation ability, is a long-term problem for computer education. At the second national university undergraduate teaching working conference, Ji Zhou, the previous minister of education, pointed out that "We must strengthen teaching work and improve teaching quality." [1] One of the major work is to vigorously strengthen the cultivation of college students' learning ability, practical ability and innovative ability. According to the characteristics of the current basic course system, Guoliang Chen [2] [3] who is academician of the Academy of Science of China, introduced the concept of computational thinking to apply the basic concepts of computer science to problem solving and system design, and a series of thinking activities about computer science [11].Academician Chen penetrates computational thinking and creative ability penetrate into the architecture of computer basic education and makes the ordinary students really grasp how to do work and create more schemes with computer through basic computer teaching mode change, which aims to cultivate students with computer teaching and innovation practice.

2 Current computer education teaching

Currently, the university computer basic courses prefer more basis and focus on credits, less application and ignore capacity. After the completion of the university computer basic course [4] [5] [6], such as fundamentals of computers, programming, application courses, most of non-computer professionals courses for students tend to the end, without the support of follow-up professional practical and innovational course system. The students learn professional courses directly, the computer knowledge learned is rarely practical to solve practical problems, especially for the professional areas. The computer is not combined well with the no-computer profession due to the lack of thinking about how to think for the computer. Although the non-computer professional undergraduate will use some computer knowledge when doing graduation design, however, the students' computer application ability need be further improved because of a period of no practical time. In addition, students also have some other problems in the process of learning computer knowledge, such as the teaching just for teaching without using innovation and practice exercises, and don't grasp of computational thinking deeply. For students, they don't know how to use what they learn, which does not reflect the ability of the computer to solve problems. Due to the existence of these phenomena, teachers feel that so much basic computer knowledge can't be well pass on to students, and students can't get a good absorption, the key spirit of innovation and innovation practice projects is neglectful. This situation restricts the development of the students' computer knowledge, and affects the application and expansion of basic computer knowledge in the field of non-computer professional, which is not conducive to cultivating innovative talents for the whole country. With the development of information technology, the computer is an essential tool, like the invention of the printing in our country, as we can hardly move an inch without computer. So it is extremely important to foster the thinking and innovative ability of students. Response to these phenomena, firstly we start from the teaching mode set of basic computer course. The purpose of building basic tutorial system about developing computational thinking innovative capacity is to promote and foster the students' ability of using computer to improve the level of expertise and innovative ability.

After discussion and literature review, our computer basic teaching system mode has great changes. Figure 1. shows our earlier basic computer teaching system model. Initially, this mode greatly promotes the development of basic computer teaching. However, it has encountered some problems under the rapid development of computer technology. The following starts from architecture for next analysis.



Computer application curriculum	electronic commerce	practical networ technology	3-D animation technology	...	
computer technology curriculum	database technology and application	Multimedia technology an application	software development and applicaiton	system and network technology	software engineer and applicaiton
program design group	C language	JAVA languag	VB	Web	
basic curriculun group	college computer basis coures		college computer elementary course		
curriculum group	curriculum name				

Figure 1. Teaching curriculum group architecture

The architecture is divided into four levels: basic curriculum group, program design group, technology curriculum group and the application curriculum group, and does not reflect the concept of computational thinking. In the lessons, students prefer to learn and apply, rather than analyze and solve problems from a higher level. The information interaction and knowledge sharing of the course is not perfect. In this case, the pattern of course architecture based on computational thinking is proposed, which takes computational thinking as the main line. In addition to the co-coordinated development of various modules, the construction of information-sharing platform will help a variety of knowledge sharing.

3 Teaching Innovative Practice Platform of Computational Thinking driven

According to our situation and the development of computer technology, in conjunction with other universities [2] [6] [7] [8], the original architecture has been improved. As shown in Figure 2: 1)Huge homework in the course, where it is a combination of the feature of no-computer professionals and practical projects related to the curriculum, such as the students development the system related chemical engineering simulation. At the end of terms, students should complete the homework; 2)Take the concelt of computational thinking as the driver, and make the coordinated development of various modules, while focusing on training students' innovation ability; 3)Optimize original architecture of the basic course group t; 4)Add or replace some courses to make the computer basic course group more reasonable, such as C#, VB.NET, Pyhton, and network attack and defense etc..

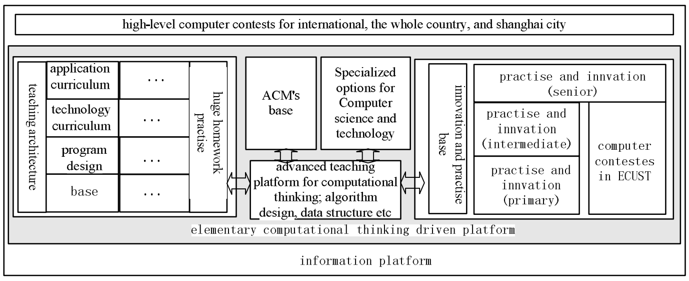

high-level computer contests for international, the whole country, and shanghai city
teaching architecture
application curriculum
technology curriculum
program design
base
...
huge homework practise
ACM's base
Specialized options for Computer science and technology
advanced teaching platform for computational thinking; algorithm design, data structure etc
innovation and practise base
practise and innvation (senior)
practise and innvation (intermediate)
practise and innvation (primary)
computer contestes in ECUST
elementary computational thinking driven platform
information platform

Figure 2. Computational Thinking- driven Teaching and Innovative Practice architecture

The improved architecture is mainly divided into teaching system module, innovation practice base module, computational thinking driving component, ACM training, and information teaching platform module. Furthermore, the computer basic course group architecture should be taught for every course managed by Computer Teaching & Experiment Center, which mainly aims at talking about preliminary knowledge about computational thinking, and also making students know the place where the basic computer course they are learning in the course architecture. So that they should know what they have learned, what they will learn and which computer practical innovation and intramural competitions they can participate in. Each part will be described in the following.

3.1 The opening chapter of the computer architecture

The opening chapter: non-computer science students often feel quite confused about what they can learn on basic computer courses. They also have no idea about what teachers should talk about, either how to solve the problem from a computer perspective. Furthermore, they are confused about the level where they are and the use of it? Therefore, the teachers need to tell students that the position of the course they are learning, what they can do when they eventually finish them, the importance of computational thinking and the things we can do in conjunction with professional knowledge and how to conduct innovative practice activities with the use of computational thinking. After this guide, the students will put some loss and know the level of computer knowledge they have learned and what they need to do, how to use computational thinking to solve the problems in the applications. All in all, the opening chapter is very important, because we have to make the non-computer science students know that what we are training is, the future computer talents of non-computer science, the creative talents using computational thinking to solve problems in this field.

3.2 Strengthen Computational Thinking Efforts and Improve Computer Connotation Construction

This module emphasizes the idea of computational thinking can make students better grasp it, for the drive platform of primary computational thinking (showed in Figure 2). We should strengthen the understanding of the basic computer concepts through experiment and practice to improve the computational thinking ability of students.

Firstly, we must understand the definition of computational thinking, be conscious of what computational thinking and computer knowledge connotation are. Secondly, we try to understand the connotation of computational thinking, because of computational thinking far more than meaning for computer programming, but also requires the ability to think in abstract multilevel. Thus, to develop computational thinking is to broaden everyone's way of thinking, open the ideas of everyone. Especially, to the classroom or innovative practice, we request: first of all, teachers should often lead students to think, conduct more and teach more how to analyze and solve practical problems with the computer perspective . For example, how to solve (using mathematical equations) it from the perspective of mathematical thinking and with computational thinking (using iterative approximation), how to use the computer to solve it is an application of computational thinking. Secondly, teachers should also put forward questions appropriately and talk more about original computer knowledge to make students know how the computer is to solve the problem use computer, then it can strengthen students' abilities and interests of computational thinking. In specific practices, we will evaluate the ability of computational thinking through construing and lectures. However, also less the research of the quantitative assessment for computational thinking. So we will study and design system to evaluate computational thinking and connotation construction in this project, further to specific the evaluation of computational thinking to the key points of teaching, to make quantitative assessment and to improve connotation construction, accordingly to make it more scientific through researching and improving the project, and form an iterative rising spiral of continuous improvement process.

3.3 The Optimization of Innovative Practice Base Modules

In Figure 3, we have done some appropriate changes for innovative practice base modules. Due to the strong practicality of computer knowledge, if we only learn computer theoretical knowledge, what students master is not well. Therefore, according to the university’s requirements, we make some further reform for innovative practice module, such as increasing the cultivation of computational thinking, letting students know how computer works and why it set such a variable rather than as set. Thus, it allows students to improve computational thinking skills, and facilitates them better conduct computer innovative practices; Another example is that increasing excitation measures and leading into improvements based on computer-interest, including the primary intermediate and senior innovative practice activities, ACM and the knowledge of a variety of computer knowledge contests etc.. At the same time, we evaluate innovative practice base construction quality through innovative practice performance (such as student evaluation, competition, etc.).

3.4 Teaching Curriculum Group architecture

In teaching architecture module in Figure 2, the professional and enhancing classified guidance teaching could be talked. For example, students are encouraged to participate in the innovative practice activities in the form of huge homework, the specific method is to mix lesson comprehensive training (practice activities) into course groups, allow students to freely compose a group of 2-3 to finish a characteristic professional engineering work, and then guide them into the intermediate practice and innovation of practical activities according to the completion situation, so as to lay a foundation for the next senior innovation activities and cultivate good seedlings. In addition, because of the majority of high school students have access to computers, so we detect whether students need to learn the course of college computer elementary course through the examination. Taking the learning interest and the combination with professional practice into account, we will introduce C#, VB.NET, python and other courses. In the practice, we assess the teaching quality by lectures, student interview, test scores and the projects. For both the humanities and science, we take classified teaching to better enhance students' learning interest and enthusiasm. Specific description is in Figure 3. Simultaneously, in the classroom, we also explain the application method of computational thinking, particularly:1) in program design group, process-oriented variables, find and sort(type, recursive, optimization and compromise);data structure(abstract, recursive, compromise decomposition); expressions, control statements, function (abstraction, recursion, decomposition, reduction); System building, code reuse, and module building (system recovery, planning, parallel, simulation, protection). 2) in network course, due to the variety of the different nature of network courses, we will explain different content depending on different levels. In general, we will talk about: finding transmission problems, such as transmission error (retransmission), CRC(protection, redundancy, error correction and system recovery); the OSI hierarchical structure and the TCP/IP structure (reduction, decomposition, scheduling, SoC, etc.) network protocol / function (abstraction, decomposition, compromise etc.) etc..

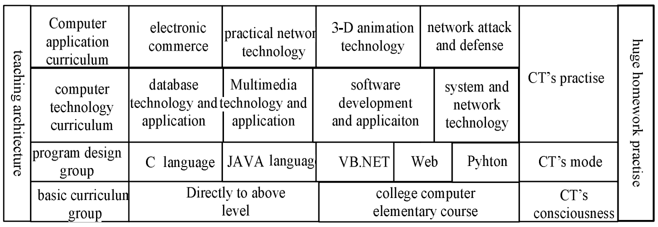

teaching architecture
Computer application curriculum
electronic commerce
practical network technology
3-D animation technology
network attack and defense
CT's practise
computer technology curriculum
database technology and application
Multimedia technology and application
software development and applicaiton
system and network technology
program design group
C language
JAVA language
VB.NET
Web
Pyhton
CT's mode
basic curriculum group
Directly to above level
college computer elementary course
CT's consciousness
huge homework practise

Figure 3. Teaching curriculum group architecture improved

3.5 Computational Thinking Driver

This part is very important under the teaching of primary computational thinking, the students have had some ideas about the computational thinking. On this basis, this part will focuse on how to use the knowledge of computer commonality. For example, for the 20 key concepts of computational thinking (including calculation, communication, coordination, automation, design reduction, embedding, transformation, simulation, recursive, parallel, abstract, decomposition, protection, redundancy, fault tolerance, error correction, system-recovery, heuristics, scheduling). For different students, we use different emphases and strategies when explaining specifically, such as ACM programming contest, we will explain all of it; for junior students, we will explain communication, coordination, automation, design, reduction, embedding, transformation, simulation, recursive, parallel, abstract, decomposition, protection and redundancy; for other innovation practices and teaching, we will explain some concepts selectively according to the characteristics of the course. For example, in C programming course, firstly, we will explain the concept of computational thinking, and require students to review literature bout computational thinking. Then, we will ask students to show their understanding and reflection of computational thinking with regard to the knowledge points of each chapter, and request students to give examples of their use in real life. Finally, they should complete a project which is relative their majors. All the activities of students can be learned in the teaching platform through background management functions. In the aspects of practical application, interested students are encouraged to submit their own design jobs, then further to participate in innovative practices in computer contests at all levels. The practice shows that the method works well, it is worth further improved.

Additionally, in the ACM construction driven by computational thinking, in connection with the characteristics of ACM students(we have counted for each recruiting, more than 90 percent of students are proficient in C language and data structure), in this case, we will let them master more computational thinking concepts, while obtaining preliminary computational thinking by themselves. We emphasize on instilling the 20 technical points of computational thinking at thinking training platform and then apply it into various practical modeling. Such as focusing on coordination, design, reduction, embedding, transformation, simulation, parallel, abstract, decomposition, protect, redundancy, fault tolerance, error correction, heuristics and scheduling, some other basic concepts can be acquired through self-study. By way of emphasis training, we obtained very good results in the ACM contest, which is shown in the table-1. At the same time, ACM team members act as teaching assistant role, they can help us organize various computational thinking lectures and train student computational thinking ability. For example, for the minimum cost maximum flow analysis, the minimum spanning tree problem, etc., which involves reduction transformation, simulation, recursion, type, SoC, planning, scheduling, compromise, redundant, fault tolerant, heuristics, etc., they will explain well to innovation team members and the other groups.

3.6 The Information Teaching Platform

An information technology teaching platform is developed due to the diversification of communications between teachers and students in our university. The platform includes homework upload system and forum (including innovative practice sub-forums) etc..

4 Practice and conclusions

By the improvement of computer teaching architecture based on computational thinking, the peer teachers of many universities such as Tong Ji university, ShangHai university, East China Normal University etc approve our work. We made great progress in the aspects of teaching group architecture, and strengthen students' computational thinking ability and practical innovation. After the implementation of the new teaching innovation platform, the level of theory and practice of our students greatly increase, especially computational thinking ability. Students can learn to analyze and solve problems from the perspective of computer, and these improvements directly reflect into the practice innovative module. After the teaching of innovation practice, our students' ability has been greatly improved, the mutual development of theory and practice levels directly promote the improvement of the level of innovative practice, and achieved good results. For example, in the past year, more than 300 people have participated in computer knowledge contest. In classroom and innovation practice, our teachers use computational thinking ideas to impel students to analyze and solve problems from computer perspective with computational thinking issues, and achieved very good results. For instance, in Shanghai computer application ability contest in 2010,2011and 2012,we have achieved very good results(table 1); in Chinese college students computer design contest, we have won many prizes, and we have obtained the outstanding organization award in Chinese college students computer design contest in 2012; Since our students possess a high level of theory, computational thinking and practical level; the East China University of Science and Technology also achieved very good results in ACM finals is also obtained.

TABLE I. RESULTS OF CONTESTS

years	quantity of students enrolled in practise	quantity of student passed(the credit is 0.5)	Shanghai's prize（ the first prize）	Nation prize(the first prize)	ACM's prize(gold medal)
2009-2010	800	305	3（1）	1（0）	There are three Gold medals, seven silver medals, sixteen bronze medals.
2010-2011	1000	360	12（2）	3（1）	
2011-2012	1200	532	13（2）	8（1）	

All in all, the new computer teaching platform has achieved good results. At the same time, computational thinking drives to increase innovation practice, which greatly improved the students' enthusiasm, initiative, practical ability, not only make students master computer knowledge well, but also deepen the mastery of basic computer knowledge through theory teaching, practice, competition and innovation practice and other methods, so that students can better use computational thinking ideas to think about how to use computer tools to solve problems in the real world. Thus it can better trains students' abilities to identify problems, solve problems. But each system mode has two sides, that whether it will bring good results in the implementation process needs to be adjusted according to the actual situation of our university and local conditions. Only in this way can it achieve the best effect.

5 References

[1] Wing J M.Computational Thinking[J].Communications of the ACM,2006,49(3):33-35.

[2] Yizhen Zhou. Computational Thinking[J].China Computer Federation Communications,2007,3(11):83-85.

[3] Jiehua Chen.The Practice and Exploration of Enhancing Computational Thinking Training In Promgramming Courses[J].Computer Education,2009(20):84-85.

[4] Wenhu Wu, Jiande Wang. World Collegiate Programming Contest(ACM/ICPC) Advanced tutorial The Computational Thinking Method Used in Programming design, 2009(7):1-3.

[5] Guoliang Chen, Rongsheng Dong. Computational Thinking and University Basic Computer Education[J]. China University Education ,2011(1):7-11.

[6] Qingming He, Hanquan Lu, Boqin Feng. The Core Mission of Basic Computer Teaching is The Training of Computational Thinking Ability: “ C9 Joint Statement of Basic Computer Teaching Development Strategy "Interpretation[J]. China University Teaching , 2010(9):22-32.

[7] Ministry of Education Basic Computer Courses Teaching Steering Committee.Institutions of Higher Learning Basic Computer Education Development Strategy Study Cum Computer Basic Courses Teaching Basic Requirements[M].BeiJing:Higher Education Press,2009:16-28.

[8] Dejiang Zhang.Reform of Teaching Methods and Cultivate Innovative Talents[J].China University Teaching,2009(5):7-10.

[9] Zongli Jiang.With the Ability to Cultivate To Improve the Level of Computer Teaching Oriented With Ability Training.[J].China University Teaching,2008(8):35-37.

[10] Dunrong Bie.University Teaching Methods Innovation and Improve The Quality of Higher Education[J].Journal of Education,2009,30(40):95-101.

[11] Xiongchun Duan. Colleges and University Modernization of Teaching Methods[J]. Higher Education Exploration,2009(4):133-134.

[12] Yong Li,Bin Wu,Youqing Luo.Thinking about University Rational Reform of Teaching Methods:Based on The Principle of The Institutional Environment Perspective[J].China Higher Education Research,2011(5):83-85.

SESSION

POSTERS AND SHORT PAPERS

Chair(s)

TBA

An Overview on Big Data

Qiming Niu [12], Feng Liu [1], Chun Zhang [1], and Jie Liu [3]
[1] School of Computer Science, Beijing Jiaotong University, Beijing, China
[2] Computing center, Hebei University, Baoding, China
[3] Department of Computer Science, Western Oregon University, Monmouth, Oregon, USA

Abstract - *With the development of the Internet, cloud computing, Internet of Things and mobile devices, Big Data has become very important and crucial. Hence, in this paper, we Introduce the basic application, define Big Data ,describe the characteristics of Big Data and processing steps, discuss the development trend of Big Data, and summarize the major representative data application platform. Finally, we believe that the Big Data will benefit mankind in various fields.*

Keywords: Big Data, Cloud computing, Data mining.

1 Introduction

Ubiquitous wireless sensors, mobile devices, the RFID have continued to produce large amounts of data every minute. At the same time, various Internet services Such as, cameras, microphones, software logs always produce a huge amount of data. In recent years, with the rapid development of the Internet, cloud computing, Internet of Things and mobile devices, Big Data is received people's attention increasingly. The world's technological per-capita capacity to store information has roughly doubled every 40 months since the 1980s; as of 2012, every day 2.5 quintillion (2.5×1018) bytes of data were created [1] .

2 Application

Now we are living in a big data environment, and more and more dependent on it. The data from a hundred thousand sensors on a plane is Big Data. If there are one hundred thousand sensors on a plane, each producing an eight byte reading every second, these sensors would produce less than 3GB of data in an hour of flying (100,000 sensors x 60 minutes x 60 seconds x 8 bytes) [2] .

At present many countries and organizations all over the world have established data open web portal. For example, U.S. government data open website is Data.Gov, the British government's open Data site is Data.Gov.uk. In 2012, the U.S. National Institutes of Health announced the 1000 Genomes Project data is free to the public. These data totaled 200TB. And this is just the tip of the iceberg of the expanding biological data.

American Democratic politician Barack Obama's re-election of the President of the United States successfully. Big Data applications play a significant role in Obama's re-election. Basketball fans around the world can access NBA.com and query vast amounts of official statistics in the support of SAP HANA real-time data analysis platform, and they can analysis of the data and information of the history of NBA games, teams and players. Weather Bureau using DataCloud technology provide meteorological content to all the TV, Internet, mobile terminals and automobile products. DataCloud can provide more valuable, more timely meteorological information through the analysis of flood of meteorological data. Relying on Big Data, people can accurately forecast the trend of the air tickets price, and analyze flight delays. After the earthquake in Japan on March 11, 2011, the U.S. National Oceanic and Atmospheric Administration (NOAA) has successfully predicted the tsunami in Japan. Some large companies, such as: Wal-Mart, eBay, Taobao benefited greatly from Big Data. we can get real-time traffic information from vast GPS data captured by the taxi or bus. Google self-driving car application thanks to a lot of geographic information. Alibaba company will be able to know the merchants of funding problems through analysis of a large number of commercial data, so they know who need to borrow money. The commodities recall phone inquiry system has become a boon for consumers. From the use of data mashups, the people see the benefit of open data.

The successful application of Big Data in various fields let us see the power of the Big Data. It showed that the era of Big Data has arrived. Therefore, people need to face the challenge from the collected data, open data, use data in the era of Big Data.

3 Definitions, characteristics, and trends

In the early 1980s, American people put forward the concept of Big Data. Ever since then the amount of data in various fields are in rapid growth, Big Data is becoming more and more popular, more and more important. So what is the definition of Big Data? What are the characteristics of Big Data itself? How do we manage Big Data，and how to apply it? What is the trend of development of the Big Data?

3.1 Definitions

There are a lot of Big Data definition from a different perspective. Such as Big Data usually includes data sets with sizes beyond the ability of commonly used software tools to capture, curate, manage, and process the data within a tolerable elapsed time [3] . In 2012, Gartner defines Big Data as follows: "Big Data are high volume, high velocity, and/or high variety information assets that require new forms of processing to enable enhanced decision making, insight discovery and process optimization."[4]”Big Data” refers to datasets whose size is beyond the ability of typical database software tools to capture, store, manage, and analyze [5] . CCID Consulting present the following concepts: Large data refers to the rapid acquisition, processing, analysis to extract value from massive, diverse transaction data, interactive data and sensor data [6] . All in all, Big Data is described as very large data sets. Specifically how much data can be referred to as Big Data, there is no universal definition. With the development of technology and the changes of actual situation, The definition of the data does not need to be unified.

3.2 Characteristics

Big Data mainly involve four aspects: volume, velocity, variety and value [1,7] .

Volume –Volume define a capacity of Big Data. The size of data is in terabytes and petabytes, even exabytes.

Velocity –Speed is the demand for Big Data of acquisition, processing and analysis. The emphasis of velocity is the need of real-time analysis. The efficiency of processing data is the life of the enterprise.

Variety –Variety define data types of Big Data. Big Data include structured and unstructured data such as text, audio, video, sensor data, posts, log files and more.

Value –Value is the purpose of the analysis of Big Data. Only data combined with the appropriate business analysis, can data create value.

It should be noted that the significance of Big Data is that humans can analyze large amounts of data, use these data, discover new knowledge, create new value and bring the rapid development.

3.3 Processing flow

The processing flow of Big Data mainly include: acquisition, preprocessing, analysis and data mining.

First, the collected data is stored in the database. MySQL, Oracle, HBase or MongoDB is the common database. Second, If you want to effectively analyze these massive data, you should import the data from the front-end to a concentration of large-scale distributed database, distributed storage cluster, and can import based on some simple cleaning and pretreatment. Third, the user can get the analysis of the most common requirements through to the huge amounts of data query summary and classification. Fourth, Data mining mainly get some prediction based on the calculation of all kinds of algorithm in the existing data in order to meet the demand of some high-level data analysis.

3.4 Trends

The data is the information carrier, is the source of knowledge. Therefore, the data can create value and profits. It can be predicted that the knowledge-based competition will be mainly reflected in the data-based competition, the competition of Big Data is the inevitable result of economic development. The future trends of Big Data are mainly the following three points:

Firstly, the data is becoming an important asset. Information department turn into a profit center from a cost center. In the era of Big Data, data penetrate in various industries, and gradually become strategic assets. Data size, activity, ability to collect, and using data will be the core competitiveness of enterprises.

Secondly, the decision becomes more intelligent. Government policies and services will be more accurate. The strategy of corporate will shift to data-driven from business-driven. Traffic routes which people choose will be more rational. In short, That we promote the application of big data will improve the level of social intelligence.

Thirdly, The emergence of visualization technology has formed a the integrity system for a data acquisition, data integration, data analysis, data mining, data show. The essence of visualization is to show the data in an intuitive way. The user can find the nature of the underlying problem of unstructured, non-geometric abstract data.

4 Market-oriented Big Data platform

Big Data platform is able to quickly integrate and manage different types of large capacity data. It can provide a set of tools for large data analysis and mining technology to accelerate discovery. The following introduce some Big Data platform of well-known IT vendors.

IBM's Big Data analytics platform is InfoSphere. It includes the BigInsights and Streams. The product is released in May 2011. Biglnsights can analyze large-scale static data. It provides multi-node distributed computing. It can add nodes to enhance the ability of data processing when necessary; Streams using the the memory calculation method analyze real-time data.

Oracle provides a data integration solutions through optimization of software and hardware integration. Oracle released Oracle Big Data Appliance in January 2012. The hardware part of Big Data Appliance includes server, CPU, memory, disk space and network. the software part of Oracle includes Linux, Oracle JDK, Cloudera Hadoop Distribution, Cloudera Manager, NoSQL DataBase , etc.

In December 2011, EMC released Big Data analysis platform Greenplum. This platform can handle structured, semi-structured and unstructured data. It realizes the seamless integration in various types of data.

SAP’s Big Data platform is mainly composed of real-time data platform HANA, analytical database Sybase IQ and transactional database Sybase ASE. In December 2010, SAP released the SAP HANA.

Microsoft has HDInsight Server and HDInsight Service two types of Big Data platform. Users can read Big Data through Hadoop platform. In the Excel environment, the results are stored in Excel in the form of a Table or Cube.

Intel released Intel Hadoop Manager2.0. According to Intel's hardware, the platform enhance and optimize the Hbase, HDFS. Platform performance is significantly improved.

All in all, Hadoop has been acknowledged as the basic platform of the new generation of data processing. EMC, IBM, Microsoft, and Oracle use the basic platform.

5 Future thoughts and Conclusion

At present, there will be a lot of problems and conflicts in the field of big data. For example, How do official statistics deal with the challenges of Big Data? Disclosure of sensitive data will result in very serious consequences, which makes us very worried about data privacy issues. Is there enough protection? How to solve the contradictions of user privacy and convenience? To many users, the key problem of Big Data is the price, ease of use and speed. How can we solve these problems?

All in all, Big Data is in its infancy. The Big Data will promote the development of the society, economy, and technology. From the perspective of the information technology, Big Data promote IT technology and product innovation.

6 Acknowledgements

This work is supported by The National High Technology Research and Development Program of China(Grant No.2012AA040912). The authors would like to thank attendees of China Communications Standards Association, China Unicom, CHINA TELECOM, and China Mobile Group Design Institute Co., Ltd. for their review and helpful comments on a draft of this paper

7 References

[1] WWW Link: http://www-1.ibm.com/software/data/bigdata/

[2] WWW Link: http://mike2.openmethodology.org/wiki/Big_Data_Definition

[3] WWW Link: http://en.wikipedia.org/wiki/Big_data

[4] Douglas, Laney. "The Importance of 'Big Data': A Definition". Gartner. Retrieved 21 June 2012.

[5] Manyika, James; Chui, Michael; Bughin, Jaques; Brown, Brad; Dobbs, Richard; Roxburgh, Charles; Byers (May 2011). Big Data: The next frontier for innovation, competition, and productivity McKinsey Global Institute, May, 2011 p6

[6] WWW Link: http://tech.cnr.cn/list/201207/t20120711_510180269.html

[7] Sachchidanand Singh. Big data analytics, 2012 Internati - onal Conference on Communication, Information Computing Technology (ICCICT 2012), p 4, 2012

Using an Insurance Company Database Example to Teach Database Design and Data Mining Principles

Julia Bao (U. of Phoenix)
juliabao@email.phoenix.edu

and
Chris Helwig (EMCC)
WORLDCOMP'13

ABSTRACT

This project involves using an insurance company's data to illustrate database design principles. Students are told to suppose they are designing a new database system for an insurance company. The insurance company provides auto, homeowners, renters, and life insurance policies to its customers. Students must identify what kinds of data to place in database tables, what data types to use, what primary and secondary keys to use, and what relationships to enforce between database tables. To complete this task, it is necessary to understand the insurance company's business model, as well as to consider how the database might later be used as an analytical tool and as a data repository.

General Terms

Computer Science Education, Database Design, Microsoft Access.

Keywords

Insurance Company Database, Database Schema.

1. The database could include the following tables

1. Customer Info, with fields for ID, Last Name, First Name, Address, City, State, Zip, Phone, Credit Card Number, Exp Date, Multi Policy Discount, DOB.
2. Auto, with fields for Auto Policy Number, ID, Make, Model, Year, Liability Amount, UM, UIM, Med Pay, Premium Amount, Collision Damage Amount, Named Insured, Additional Driver.
3. Homeowners, with fields for Homeowners Policy Number, ID, Liability Amount, Property Damage Amount, Premium.
4. Renters, with fields for Renters Policy Number, ID, Liability Amount, Property Damage Amount, Premium.
5. Life Ins, with fields for Life Ins Policy Number, ID, Premium amount, Benefit Amount, Beneficiary.

The Customer Info table would contain basic customer contact, biographical information, and payment information. The Auto table would contain information related to a customer's automobile policy. UM refers to uninsured motorist coverage and UIM refers to under-insured motorist coverage. This table contains as a foreign key the Customer Info table ID field. The Homeowners table contains information related to Homeowners insurance coverage, the Renters table contains information related to Renters insurance, and the Life Ins table contains information related to Life insurance.

Also needed would be transaction tables for transaction related data:

1. Payments, with fields for Payment Transaction ID, ID, Due Date, Amount Due, Amount Paid, Paid on Time.
2. Claims, with fields for Claim Transaction ID, ID, Amount of Claim, Claim Approved.

The database schema should avoid data redundancy, maintain data integrity, and organize data in a logical manner.

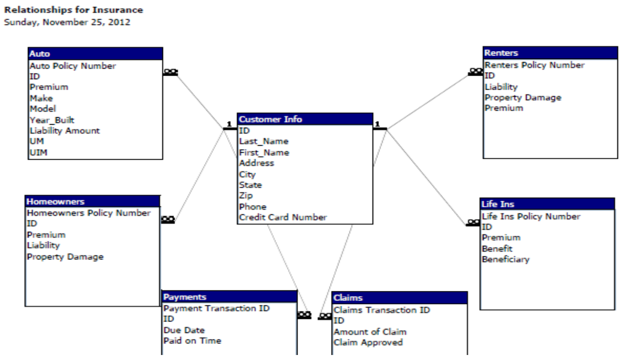
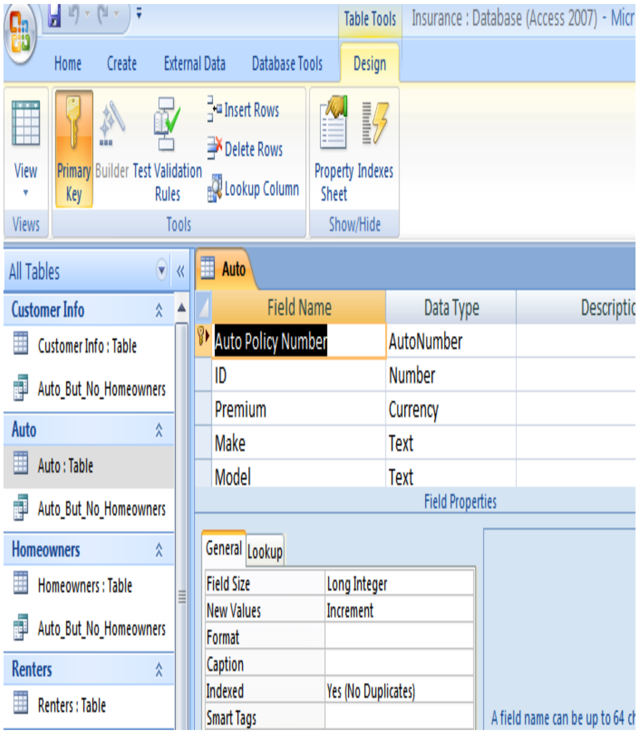


Figure 1. Screen shots of what this database would look like in Microsoft Access.

2. Further Database Details

This relationship report above was generated in Access under Database Tools | Relationships and Relationship Tools | Design | Relationship Report. A one to many relationship with referential integrity was enforced between the ID column in Customer Info and the ID columns in the other tables.



Field Name	Data Type	Descriptio
Auto Policy Number	AutoNumber	
ID	Number	
Premium	Currency	
Make	Text	
Model	Text	

Field Size	Long Integer
New Values	Increment
Format	
Caption	
Indexed	Yes (No Duplicates)
Smart Tags	

Figure 2. Column names for the Auto database table.

Appropriate data types for each table column were added in design view using the Data Type drop down arrow.

How might datamining be used to analyze this data? One way would be to identify customers who have Auto policies but not homeowners policies, and then target such customers with marketing efforts to sell them homeowners policies.

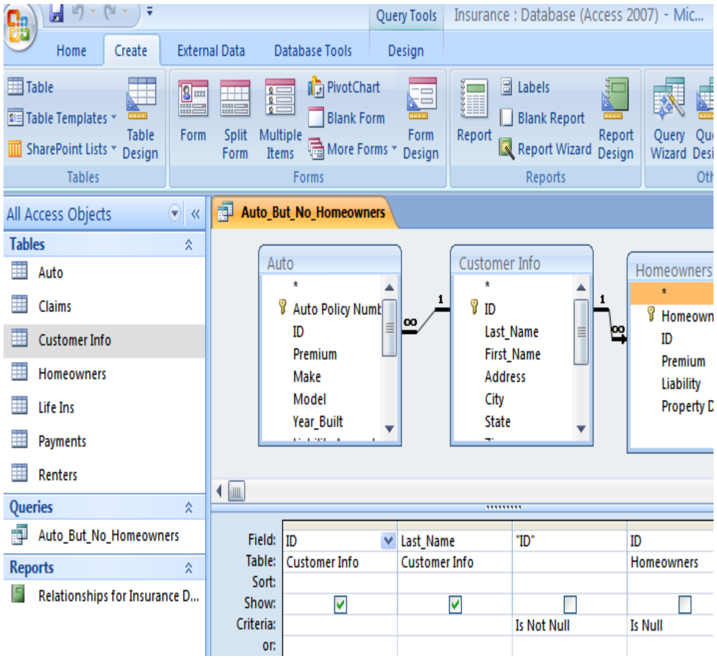
In SQL, the command would be:

SELECT [Customer Info].ID, [Customer Info].Last_Name

FROM ([Customer Info] INNER JOIN Auto ON [Customer Info].ID = Auto.ID) LEFT JOIN Homeowners ON [Customer Info].ID = Homeowners.ID

WHERE ((("ID") Is Not Null) AND ((Homeowners.ID) Is Null));

This query was created under Create | Query Design. The SQL was generated by selecting SQL View.



Field:	ID	Last_Name	"ID"	ID
Table:	Customer Info	Customer Info		Homeowners
Sort:				
Show:	☑	☑	□	□
Criteria:			Is Not Null	Is Null
or:				

Figure 3. Building a query in Access.

Other uses of the database information would include finding the optimal setting for premium amount, amount of discount for multiple policy holders, and probability of a policy holder making a claim. Data miners might consider if likelihood of claim varies by region, climate, or driver's age or driving record.

There are several other database design exercises students might be asked to complete. Students could be asked to design database tables for an online retailer such as Amazon or for an online movie review website or an airline reservation website. Students could be asked to design a database for a credit card company, and include tables for customer credit history, bankruptcy history, credit applications, potential customers with their credit scores, formulas for the amount of credit to extend, formulas for which potential customers to contact with marketing literature, and transaction data on credit card balance payment records. Students could also be asked to design a database for a computer chip manufacturer, and include database tables for manufacturing data on yield ratios for various chip designs, the use of feed forward and feed backward process control, and total volume of chips produced compared with fab utilization rates. This data could be combined with sales data for various computer chips, cost data for supplies, and pricing data on various products.

.

SESSION

DATA STRUCTURES, RESEARCH EXPERIENCE, ABET AND ASSESSMENT

Chair(s)

Prof. Hamid Arabnia
University of Georgia

An Analytical Study of Data Structure, Algorithm and Link Utilization of Perfect Difference Network

Rakesh Kumar Katare
Dept of Computer Science
A.P.S. University, Rewa,
M.P. India - 486003
katare_rakesh@yahoo.com

N.S. Chaudhari
Dept. of CSE,
IIT, Indore,
M.P. India - 452017
narendra@iiti.ac.in

Shashi Kant Verma
Dept of MCA, ITM
Universe, Gwalior
M.P., India – 474001
skv.shashi@gmail.com

Rajesh Roshan Raina
Dept of Computer Science
A.P.S. University, Rewa,
M.P. India - 486003
luckyrajmn@gmail.com

Abstract : A study of topology and its use in PDN architecture is the purpose of this paper. Graph theory methodology has been followed for network communication.

The rich connectivity of PDN's made it possible to circumvent faulty nodes and links with little or no increase in routing distance with less or negligible performance impact. The topological properties of PDN are reviewed and some lemmas are developed which may be beneficial for the future research. The simulation result of network interconnection for link utilization is decisive for the cost and profitability of installation of such network for parallel and serial computation. A parallel data structure for PDN architecture has been developed for perfection to test the robustness of PDN the link utilization during normal course and link failure has been properly studied. An approach is devised to form chordal PDN and Bipartite PDN and various interconnection between different nodes. A technique to form cyclic code PDN mapping of nodes in the algorithm form and drawing of simulation in pictorial form has been presented for consideration.

We used OPNET Modeler-14 Simulator [1], [15], [18] to study the behavior of the PDN. OPNET is an event driven discrete network simulator which contains a comprehensive development environment supporting the modeling and performance evaluation of communication networks and distributed systems. OPNET uses models that are specified in terms of objects, each with configurable set of attributes. Models are hierarchical to the naturally parallel the structure of actual communication networks. OPNET includes a tool for graphical representation and processing of simulation output. [10]

Keywords: Perfect Difference Code(PDC), Perfect Difference Graph (PDG,) Perfect Difference Set (PDS), Perfect Difference Network (PDN), Link Utilization.

Paper ID-FES3824

1 Introduction

An analytical study of data structure of algorithm and link utilization in a perfect Difference Network (PDN) is presented here. A higher performance study of heavy load transfer for the parallel and distributed systems has been done to understand connectivity and complexities in PDN architecture. The topological properties and characteristics of PDN are presented in the form of lemmas. A parallel data structure and link utilization studies are done to study the merits of PDN. An alternate approach to form chordal PDN and Bipartite PDN and interconnection of nodes are presented here. A technique to form cyclic code PDN, mapping of nodes in the form of algorithms and simulation results in pictorial form has been presented for easy understanding of the outcome

1.1. Perfect Difference Set

The Perfect Difference Sets were first discussed by J. Singer in 1938 in terms of points and lines in a projective plane of a Galois Field (GF) [17], [19].

Definition 1: *Perfect Difference Set* – If the set S of $\delta+1$ distinct integers S_0, S_1 ..., S_δ has the property that the $\delta^2 + \delta$ differences S_i - S_j ($0\leq$ i, j $\leq \delta$, i$\neq$j) are distinct modulo $\delta^2 + \delta$ +l, S is called a Perfect Difference Set mod $\delta^2 + \delta$ +l.

The existence of Perfect Difference Sets seems intuitively improbable, at any rate for large δ, but in 1938 J. Singer proved that, whenever δ is a prime or power of prime; say $\delta = p^n$, a perfect difference set mod p^{2n} $+p^{n\ +1}$ exists. [7], [3], [4]

From now on, let δ denote pn and we write that $n = \delta^2 + \delta + l = p^{2n} + p^{n+1}$, S = {s: $|s_i - s_j|$ mod n, where $0\leq i$, $j\leq \delta$, $i \neq j$, δ is a prime or power of prime and $n = \delta^2 + \delta + l$}[7].

1.2. Perfect Difference Network

Perfect Difference Network architecture, based on a PDS, is designed where each ith node is connected via directed links to nodes i ± 1 and i ± s_j (mod n), for $2 \leq j \leq \delta$. Each link is bidirectional and the preceding connectivity leads to a chordal ring of δ in-degree and δ out-degree (total degree of a node $d(v) = 2\delta$) and diameter D = 2 [2], [4].PDN has already been studied for, high-performance communication and parallel processing network [4] and some topological properties of PDNs and parallel algorithms [7], [8], [9], [16] were suggested. It was shown that an n-node PDN can emulate the complete network with optimal slowdown and balanced message traffic.

Although other interconnection architectures with topological and performance characteristics similar to PDNs exist, we view PDNs as worthy additions to the repertoire of computer system designers.

Alternative network topologies offer additional design points that can be exploited to accommodate the needs of new and emerging technologies. Further study is needed to resolve some open questions and to derive cost/performance comparisons for PDNs.

1.3. Perfect Difference Graph

In a PDN, for each link from node i to node j, the reverse link j to i also exists, so the network corresponds to an undirected Perfect Difference Graph (PDG), G = (V, E) consisting of a set of vertices V and a set of edges E connecting pairs of vertices in V [13], [14].

2 *Topological Properties of Perfect Difference Graph (PDG)*

Since, PDG is a chordal ring with set of vertices & edges; we have derived new results in the form of following lemmas in this section:

Lemma 2.1: The total number of members in a PDS is equal to δ+ 1.

Proof **–** As we know from the definition of PDS that δ is prime or power of prime,

$n = \delta^2 + \delta + 1$, (i.e., total number of nodes in PDN) and

Let m = total number of members of a perfect difference set.

We can write as-

$R = (\delta^2 + \delta + 1) \bmod \delta$ or

$R = (\delta^2 + \delta + 1) – m * \delta$

Here 'm' is quotient, 'δ' is denominator and 'R' is remainder and is always equal to 1 for total number of nodes in PDN.

Therefore,

$$1= (\delta^2 + \delta + 1) – m * \delta$$
$$m * \delta = \delta^2 + \delta + 1 – 1$$
$$m = (\delta^2 + \delta)/ \delta$$
$$m = (\delta^2 / \delta) + (\delta/\delta)$$
$$m = \delta+ 1$$

Hence it is proved that the number of members in a PDS is equal to δ+ 1(which is quotient).

We formulate the following properties of PDN as:

1. Total degree of a node = 2δ.
2. Total number of chordal diagonal of a single node= 2δ- 2.
3. Total number of chordal diagonals in a PDN
   = n(2 δ-2)/2 = n(δ - 1).
4. Total number of edges in PDN
   = (n (2δ-2)+2n)/2
   = (2nδ-2n+2n)/2
   = nδ.
5. Also total number of diagonals in a PDN is always even.

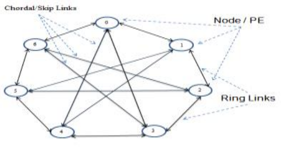


Fig. 1: Perfect Difference Network for PDS {0, 1, 3} and δ=2

*Lemma 2.2***:** *The total number of vertices in a PDG is always odd.*

Proof **–** Consider a PDS is of order n, (where n = δ (δ+l) +1 and δ is a prime or power of prime) then it gives the following conclusions.

Since, a PDG has n = δ (δ+l) +1 vertices.

Case 1: if δ is even then δ (δ+1) result in an even integer

Case 2: if δ is odd then δ (δ+1) again result an even integer

As, we know that adding 1 in even integer always turns a odd integer

So, the total number of vertices in PDG is always odd.

Lemma 2.3: Each vertex of PDG has even number of edges.

Proof **–** Since each ith node of PDG is connected to (i±1) mod n and (i±s_j) mod n node, where $2 \leq j \leq \delta$ and any two vertex having one edge between them.

Therefore d (v) = 2.δ, where δ is also equal to total number of non-zero elements in a PDS.

So each node of PDG has even number of edges

Lemma 2.4: The total number of edges in a PDG is equal to nδ.

Proof – Since the connectivity of the PDG leads to a degree d = 2δ.

Therefore the total degree of all vertices = $\sum_{v \in V(G)} d(v) = n.2\delta$

Thus actual edges in PDG = (n. 2.δ)/2= n.δ

Lemma 2.5: The PDG architecture forms an Eulerian Circuit and Path.

Proof – Since, perfect difference graph is an undirected connected graph and the degree of every vertex in PDG is always even (Lemma 4).

Therefore, perfect difference graph has an Eulerian Circuit. (Since, an undirected connected graph possesses an Eulerian circuit if and only if its vertices are all of even degree.)

*Lemma 2.6***:** *We can find corresponding vertices number of left/right half vertices by complementing its vertices number in the right/left half.*

Proof **–** Since, PDG is presentation of perfect difference set S = {s : $|s_i - s_j|$ mod n, where $0 \leq i, j \leq \delta$, $i \neq j$, δ is a prime or power of prime and $n = \delta^2 + \delta$ +l}, mod n makes it possible to assign integer addresses 0, 1, 2, ..., n-1 to vertices and arrange all vertices of PDN on a Chordal ring.

Mod n of nth complement vertex numbered i is also in PDG.

As we know that

$$s_i- s_j = (\delta^2 + \delta) \bmod (\delta^2 + \delta+1)$$
$$=> (\delta^2 + \delta)= (s_i- s_j) + (\delta^2 + \delta+1) * \text{Quotient}$$
$$=> \text{Quotient} = ((\delta^2 + \delta)- (s_i- s_j))/(\delta^2 + \delta+1)$$

Here,

Quotient = 1

('Quotient' is always equal to 1).

Therefore,

$$(\delta^2 + \delta)- (s_i- s_j) = \delta^2 + \delta+1$$
$$=> - (s_i- s_j) = \delta^2 + \delta+1-\delta^2 –\delta$$
$$=> s_i- s_j = -1 \qquad (1)$$
$$=> s_j = s_i + 1 \text{ (for initial node).e.g.}$$

when $s_i = 1$, $s_j = 0$, $\delta = 2$ and n = 7,

Therefore,

1-0 = -1

=> 1 = -1

This means -1 is the left node of the node 0 and 1 is the right node of the node 0. In other words, -1 is the complement of the node 1.

Now we can write Remainder as

Remainder = si- sj

This can be also written as

Remainder + n = $s_i - s_j + n$

=> Remainder = $(s_i - s_j + n) \bmod (n * Q)$

(Here Q = 1)

Therefore, in general we can write

Remainder =$(s_i - s_j + n) \bmod n$ … (2)

Where $(s_i - s_j)$ is the Difference Set, ‘Remainder’ is the node number in chordal ring and ‘n’ is the total number of nodes in a PDN.

Lemma 2.7: There is no direct loop between odd skip vertices in a PDG.

Proof – Since, the maximum distance of PDG is 2; if we skip an odd number of vertices then we do not find any loop in skip nodes.

Lemma 2.8: *Connection between vertices in a PDG is always hierarchical (so it is a spanning tree).*

Proof – PDSs are closely related to perfect partitions [7]. We can take any PDS in normal form and find the mod n differences $s_{i+1} - s_i$ between consecutive numbers in it, including $s_0 - s_\delta$. So, we conclude that a PDS allows us to express a large set of integers via a set of much smaller in size.

This property of a PDS motivates us to develop a hierarchical balanced tree structure of PDG.

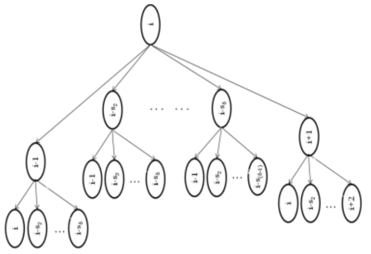


Fig. 2 : Hierarchical structure of a PDG

Lemma 2.9: PDN is the union of n balance trees.

Proof – In a PDG, a node i connected to (i ±1) mod n and $(i \pm s_j)$ mod n node, where $2 \leq j \leq \delta$ in PDS $\{0, 1, s_2, ..., s_\delta\}$.

$$\text{PDN} = \bigcup_{i=0}^{\delta^2+\delta} N_i$$

where N_i is a set of nodes connected to vertex i of the PDG of PDS $\{0, 1, s_2, . . ., s_\delta\}$

Lemma 2.10: Half of PDN is mirror image of the remaining half of the PDN.

Proof – In order to prove half of the PDN is mirror image of the second half of the PDN, let ‘n’, the total number of nodes in PDN , be divided into two parts

i.e. i = n /2

where $0 \leq i \leq n$ and n is the total number of nodes in PDN.

=> 2i = n

=> i + i = n

=> i = n-i

=> n-i = i

=> n – i = i – n + n

=> (n-i)=(i- n)+ n

=> (i- n)= (n-i)mod n (According to the remainder theorem)

But here

1) The value of quotient will be 1.

2) The value of quotient will depend upon the value of i.

Hence the above equation can be written as

i = (n- i) mod n + n (3)

Equation (3) is called the mirror image equation of the PDN.

e.g.

(1) let i = 1 , n =7 , the above equation can be written as

1 = (7 – 1) mod 7 + 7

=> 1 = 6 mod 7 + 7

=> 1 = -1 + 7

=> 1 = 6

i.e., 6 is the mirror image of 1 When n = 7.

Now its reciprocal i.e., when i = 6 , and n = 7 then we have

6 = (7 - 6) mod 7 + 7

=> 6 = 1 mod 7 + 7

=> 6 = -6 + 7

=> 6 = 1

i.e., 1 is the mirror image of 6 when n = 7.

Hence, it is proved.

Lemma 2.11: ‘0’ is the mirror image of itself in a PDN.

Proof – To prove this, Let’s consider equation (3) of Lemma 2.10

i.e.

i = (n- i) mod n + n.

Where ‘i’ is the node number, ‘n’ is the total number of nodes in PDN.

Here, i=0 and let n=7.

Therefore the above equation can be written as

0 = ((7-0) mod 7) + 7

=> 0 = ((7) mod 7) + 7

=> 0 = 0 + 7

=> 0 = 7

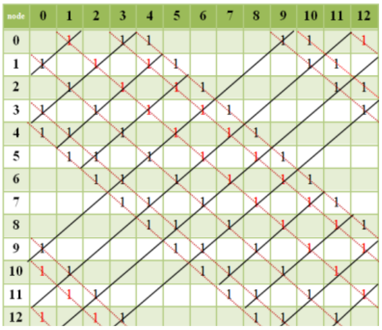
Since node 0 can be represented as node 7. Therefore it is proved that node 0 is the mirror image of itself.

3 Parallel Symmetric Neighbor Discovery

An adjacency table (Table 1) of PDG for a PDS {0, 1,3, 9}, clearly shows that the nodes are at the symmetric distance from any node of the graph so PDG is a good bipartite graph [3].

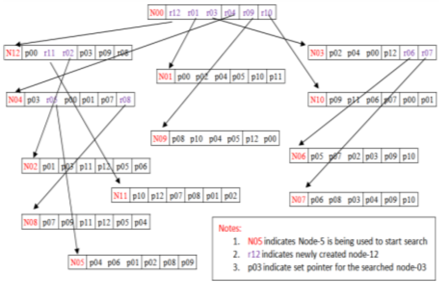
TABLE 1

DEPENDENCY TABLE FOR PDS {0, 1, 3, 9}



node	0	1	2	3	4	5	6	7	8	9	10	11	12
0		1		1	1					1	1		1
1	1		1		1	1					1	1	
2		1		1		1	1					1	1
3	1		1		1		1	1					1
4	1	1		1		1		1	1				
5		1	1		1		1		1	1			
6			1	1		1		1		1	1		
7				1	1		1		1		1	1	
8					1	1		1		1		1	1
9	1					1	1		1		1		1
10	1	1					1	1		1		1	
11		1	1					1	1		1		1
12	1		1	1					1	1		1	

An adjacency matrices "Table 1" show '1' for vertices j; if vertices j is the neighbor of vertices i in PDG otherwise it is left blank [11],[12]. An adjacency matrix for a PDN can be used to search/create n-way-tree data structure. For example, following Fig-3 shows such a tree for PDS {0, 1, 3} using adjacency matrix "Table 2". We start creating the tree from PDN node '0' (shown as 'N00' in red color) and find ring and chordal nodes- 12, 1, 3, 4, 9 and 10 as its neighboring nodes to create/search (shown as 'r12', 'r01' etc. in purple color), by implementing BFS algorithm for the tree, we get some already created/searched nodes- these nodes can be treated as a previous node links (shown as 'p00', 'p03' etc. in black color), finally we get the tree structure shown in Fig. 3.


N00 r12 r01 r03 r04 r09 r10
N12 p00 r11 r02 p03 p09 r08
N01 p00 p02 p04 p05 p10 p11
N03 p02 p04 p00 p12 r06 r07
N04 p03 r05 p00 p01 p07 r08
N10 p09 p11 p06 p07 p00 p01
N09 p08 p10 p04 p05 p12 p00
N06 p05 p07 p02 p03 p09 p10
N02 p01 p03 p11 p12 p05 p06
N11 p10 p12 p07 p08 p01 p02
N07 p06 p08 p03 p04 p09 p10
N08 p07 p09 p11 p12 p05 p04
N05 p04 p06 p01 p02 p08 p09

Notes:
1. N05 indicates Node-5 is being used to start search
2. r12 indicates newly created node-12
3. p03 indicate set pointer for the searched node-03

Fig. 3: n-tree of PDG for δ=3 and PDS={0,1,3,9}

PDN can be traversed recursively using any method of graph traversal for the various purposes. We can also represent a PDG as a threaded n-tree data structure of depth 2, e.g. Fig. 4 shows BFS traversal of the PDG as a balanced threaded 6-tree of depth 2 from vertices '0' (dark and straight edges connect unvisited vertices directly, whereas dotted lines are links to traverse recursively.)

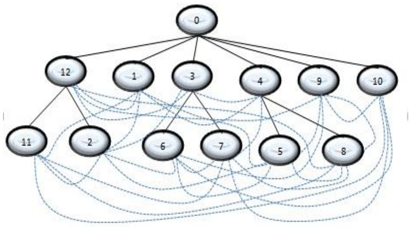


Fig. 4: Thread-tree of PDG for PDS = {0, 1, 3, 9}

4 *A Perfect Difference Shift Cycle Code for Parallel Symmetric Neighbor Discovery*

The best of the known constructive cyclic codes, for a difference set, known as random-error-correcting cyclic codes, are given by Bose–Chaudhuri–Hocquenghem (BCH) [5] , and by E.J. Weldon [19] however, we present a cyclic code, for the node of PDN, which can be used to identify its neighbors in symmetric network.

Let a set V={0,1, ..., n-1} mod n and a set D = { d_0, d_1, d_2, ..., d_{l-1}} is a l-subset of V. For every $d \neq 0$ mod n, there are exactly λ ordered pairs (v_i, v_j), $i \neq j$ such that $v_i - v_j \equiv d$ mod n. A set fulfilling these requirements is called (v, n, λ) - Cyclic Difference Set modulo n.

Each ith node of PDN is connected to (i ± 1) mod n and (i ± dj) mod n node, where $2 \leq j \leq \delta$ and $i \epsilon V$, of the network. We construct a code $N_i = (c_0, c_1, \ldots, c_{n-1})$ where $i \epsilon V$, for each node with the rule e.g., consider a PDN with 13 node for the perfect difference set {0, 1, 3, 9}, where each node has been identified by a unique positive integer from 0 to12. These codes can be used to calculate neighbor nodes addresses of a node in PDN by δ-circular consecutive left shifts and right shifts of the node code. These shifts need to be done in mod 2.

```
Algorithm: FindNeighbors (nodeAddress $N_i$)
// each node of PDN has 2δ neighbors
// half nodes on left side of $N_i$ and
// half nodes on right side of $N_i$
BEGIN
k=0
$N_k = N_i$
for all j=1 to δ    in parallel do
        $N_{k+1}$ = LeftCircularShift(Nk, j mod 3)
        k=k+1
done
for all j=1 to δ    in parallel do
         $N_{k+1}$ = RightCircularShift($N_k$, j mod 3)
        k=k+1
done
END
```

Thus the number of nodes N, which can send/receive the data within a unit time, can be written as-

$$N_i = \sum_{j=0}^{v-1} c_j$$

Where c_j belongs to {0, 1} denotes the binary bit data. Without loss of generality, we assume the active node with assigned code N_i is the desired node and c_j is the desired data bit.

5 *Methods of Interconnection of Nodes to Form Bipartite PDN using Chordal Diagonals*

We can construct a direct interconnection network with $n=\delta^2 + \delta + 1$ nodes based on the normal form of perfect difference set { $s_0, s_1, \ldots, s_\delta$ } with order δ.

Take the difference set { $s_0, s_1, . . . , s_\delta$ }, then we find out the difference between two consecutive members of the set and add it to the current member/node until it forms a new set. Once the new set has been constructed, then the remaining sets can be constructed by the increment of 1 to each element in the set to form a new set; this way all the n sets can be constructed. This is shown below in the tables.

Now let d_i be the difference between two consecutive members, n_i be the current member/node and n_{i+1} be the next member/node after the n_i

Then according to above assumption, we have

```
Algorithm GenerateChordalPDN()
// In PDS , there are $\delta$+1 members.
 // $d_i$ is the difference between consecutive
      members
// $n_i$ is the current member of the PDS.
// $n_{i+1}$ is the next consecutive member of the PDS
BEGIN
           $d_i = n_{i+1} - n_i$            where  $0 < i < n$
          if di > 1 then
 $n_i = n_i + d_i$
          else
$n_i = n_i$
          if $n_{i+1} = 0$   then        //i.e., empty/null.
              $n_i = n_i + 1$
END
```

For example;

Let δ= 2, n = 7, elements in PDS = δ + 1, and PDS = {0,1, 3}

Here,

$n_i = 0$, $n_{i+1} = 1$

Therefore,

$d_i = 1 – 0$

=> $d_i = 1$

Hence d_i is equal to 1 and is discarded according to the above algorithm

=> $n_1 = 0$

i.e., the first element of the new set is 0.

Now $n_i = 1$, $n_{i+1} = 3$

Therefore,

$d_i = 3 – 1$

=> $d_i = 2$

since $d_i > 1$

=> $n_i = 1 + 2$

=> $n_2 = 3$

i.e., the second element of the new set is 3

Again, now

$n_i = 3$, n_{i+1} = NULL

=> ‘if’ condition gets executed.

=> $n_i = n_i + 1$

=> $n_i = 3 + 1$

=> $n_3 = 4$

i.e., the third element of the new set is 4.

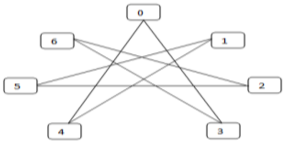


Fig. 5: PDN of Chordal Diagonals only

The new difference set is {0, 3, 4} and with this set the remaining n-1 sets can be constructed by increment of 1 as shown below in the Table 2:

Table 2:THE INTERCONNECTION OF CHORDAL DIAGONALS FOR
δ = 2, n= 7 and PDS={0,1,3}

J(Node number)	1(First element)	2(Second element)	3(Third element)
0	0	3	4
1	1	4	5
2	2	5	6
3	3	6	0
4	4	0	1
5	5	1	2

From the above table (i.e. Table 2) it is clear that J Node is connected to the other nodes through the chordal diagonals that is, ‘0’ node’s first node is ‘0 ’itself , its second node is ‘3’ and its third node is ‘4’. Similarly for node ‘1’, the first node is ‘1’ itself, its second node is ‘4’and its third node is ‘5’. This way we can find out all the chordal diagonals. The above table also shows that the nodes are circular in nature. This is achieved by saying that either from the second column or third column or from the fourth column; all the nodes shown in the table are in the increasing order with a difference of 1. The last row of these columns are just one step far from the second row of their concerning column, which means that they are also circular in nature, hence follows the property of PDN. Once the above Table 2 is formed, the bipartite PDN can be formed easily in this way. First we connect the nodes through the chordal diagonals as shown below in the Fig. 5.

The next step is to form the bipartite PDN by connecting a connector to each one of the node pairs such as {0,1},{1,2},{2,3},...........{6,0}. Then according to the above Fig. 5, connect each node to its corresponding connector.

The advantage of above mentioned Bipartite PDNs over the existing Bipartite PDN [2], [3], [4] is that it is simple , easy to form, easy to connect, and easy to understand the connections between various nodes.

The following algorithm “Mapping Nodes” shows the interconnection of the various nodes in the PDN.

```
Algorithm: Mapping Nodes
// A PDN has n= $\delta^2 + \delta$ +l nodes.
// ‘nd’ is memory allocated node.
// In PDN, there are 2*$\delta$ link of n nodes.
BEGIN
Step  1: Define variables/nodes.
Step 2: Assign memory allocation to the nodes &
 stored them in an array until condition $0 \leq i < n$ as:
for (i=0;i<n;i++)
{
   nd = malloc();
```

```
        ar1[i]=(int *) nd;
        printf("%d%u",i,ar1[i]);
        //Displayig address of seven nodes
} //end of for loop
Step 3: Map the ‘n’ nodes to n-1nodes while j < n.
// Using lemma 10/11.
Step 4: Display the results.
END
```

The expected output shows that how nodes are generated using their addresses and the appropriate node numbers. It also shows clearly that which node is connected to the four other nodes.

6 Generation of Perfect Difference Code

The Perfect Difference Code [6] is the code to represent the PDN architecture. This code can be generated by different ways. In this paper, we have made an attempt to develop the Difference Code on the basis of Perfect Difference Set. The Difference Code is the code representing the interconnection of the nodes of the PDN in the form of binary format i.e., 0’s and 1’s. This code can be used for addressing or routing any node in the PDN architecture. The only thing, to know for the said operation, is the node number ‘n’ and, shift ‘n’ bits towards right/left of the Difference Code. In other words, shifting of bits of the Difference Code is done in two ways when talking about clockwise routing or anti-clockwise routing. In clockwise routing, ‘n’ number of bits are shifted from right to left of the Difference Code in order to reach to the node number ‘n’ and vice-versa for anti-clockwise routing. This way we can rotate form source node to destination node and can form the cyclic ring. Hence shows that the code is cyclic in nature and can be called as Cyclic Code PDN.

Here is the simple algorithm to generate PDN Code based on the Perfect Difference Set.

```
Algorithm : PDN_Code_Generation()
// PDN has D+1 members and D is the prime or
power of prime number.
// PDN has (n-1)/2 nodes to the right side of
    node 0.
// PDN has (n-1)/2 nodes to the left side of the
    node 0.
BEGIN
Step 1: Define variables.
Step 2: Enter the perfect difference set and store
it in an array until condition $0 \leq i \leq D$.
Step 3: Create cyclic code for a single node by
storing ‘1’ if there is an edge from the  node else
store ‘0’ until condition $0 \leq j \leq D$.
Step 4: Map the N nodes to its 2*D links until
condition $0 \leq i < N$.
Step 5: Display the results until condition $0 \leq i<N$
END
```

7 Result of Performance Study and Simulation Output

Results of simulation are shown in Fig. 6 for link utilization of symmetric PDN. Considering each node is a source/destination of packets and hub simply forwards packets from source/hub to neighboring hubs/node of the PDN.

We have derived some important result using perfect difference graph to show the connectivity and complexity of the network. We used OPNET Modeler-14 to show the result for average link utilization for transferring formatted packets of fixed size of 500 bytes between hub to hub and hub to terminal in term of packet/sec.

Fig. 6 - shows rich connectivity of the network. In beging of the simulation all sources sends packets at different rate exponentialy to the nearby nodes after 15ms a constant rate of transsmition is gained. Inspite of different bandwidth of various links the transsmission takes place in the network at nearly constant rate. It shows that the communication time of different nodes in the network is constant.After link failure we found that average link utilization for node-0 from adjacent node 1, 4 &6, will be in the network and give us maximum bandwidth utilization in exponential time. It also shows the tolerance of the hub to hub average link utilization after the two links failure.

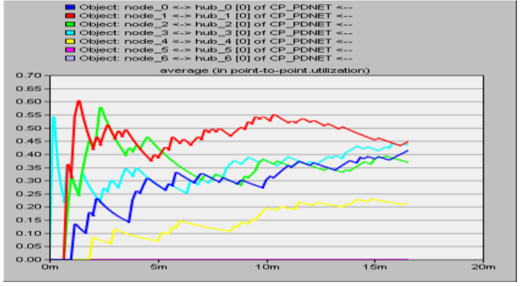


Fig. 6: Node to hub average link utilization for incoming packets/sec

8 Conclusion

In this paper, we have analyzed the link utilization of Chordal ring PDN topology for multicomputer wired network. Constant packet generation rate is adopted by packet sources/nodes. In the simulation, packets of fixed size of 200 bytes has been used. Static routing strategy was followed in each hub. We have considered network protocol UDP for transmission of the packets. We have carried out the high level simulation on Chordal Ring PDN topology using OPNET-14.

It may be concluded that the communication overhead emerging from the application of complex policies with the aim of ensuring up-to-date information, may eventually lead to adverse effects. A simple static algorithm has been proved more effective than a dynamic one. Therefore, there is a compelling need to balance the cost of the acquisition and the accuracy of the information on which the decisions of an algorithm are based. So, further enhancement an algorithm should consider not only the state and the features of the processors, but also their relative position in the network.

Although other interconnection networks with topological and performance parameters similar to PDNs exist, we still view PDNs are still worthy additions to the repertoire of computer system designers. Alternative network topologies offer additional design points that can be exploited to accommodate the needs of new and emerging technologies. The rich connectivity of PDNs makes it possible to circumvent faulty nodes and links with little or no increase in routing distance and with negligible performance impact. So we find better uses of PDNs.

We have studied topological properties of the perfect difference network and developed some very useful lemma based on our observations to the PDN. We hope that these lemmas will provide usefull guidelines for further study in the field of computer science. The simulation results of network interconnection for link utilization are very important to understand and decide the cost and profitability of installation of such network for parallel and serial computations. We present a balanced threaded n-tree structure of PDN for various operations such as neighbor discovery, flow analysis etc.

Acknowledgement

We are thankful to Prof. S. S. Katare, Retaired Professor for helpful advice to convert thoughts in usefull writing.

References

[1] Ammon, J., Hypercube Conncetivity within cc NUMA architecture Silicon Graphics , 201LN. Shoreline Blvd. Ms 565, Mountain View, CA 91343.

[2] Behrooz Parhami, and Mikhail Rakov "Perfect Difference Network and Related Interconnection Structures for Parallel and Distributed Systems", IEEE transaction on Parallel and Distributed Systems, Vol. 16, No 8, August 2005.

[3] Behrooz Parhami, and Mikhail Rakov, "Application of Perfect Difference Sets to the Design of Efficient and Robust Interconnection Networks".

[4] Behrooz Parhami, and Mikhail Rakov, "Performance, Algorithmic, and Robustness Attributes of Perfect Difference Networks", IEEE transactions on parallel and distributed systems, vol. 16, no. 8, pp 725-736.2, August 2005.

[5] Bose, R. C. and Ray-Chaudhuri, D. K., On a class of Error correcting Binary Group Codes, Inform. Control, Vol. 3, pp. 68-79, 1960.

[6] Chi-Shun Weng & Jinshown Wu, "Perfect Difference Codes for Synchronous Fiber-Optic CDMA communication Systems", Journal of Lightwave Technology, Vol. 19, No. 02, February 2001.

.

[7] Jó Ágila Bitsch Link, Christoph Wollgarten, Stefan Schupp, Klaus Wehrle Communication and Distributed Systems (Informatik 4), RWTH Aachen University Aachen, Germany {jo.bitsch, christoph.wollgarten, stefan.schupp, klaus.wehrle} @rwth-aachen.de

[8] Katare R.K., Chaudhari N.S., "Study Of Topological Property Of Interconnection Networks And Its Mapping To Sparse Matrix Model" International Journal of Computer Science and Applications, Technomathematics Research Foundation Vol. 6, No. 1, pp. 26 – 39, October 2009.

[9] Katare, R.K., Chaudhari, N.S., "A comparative study of Hypercube and perfect difference network for Parallel and Distributed system and its application to sparse linear system." Varahmihir Journal Of Computer And Information Sciences Sandipani Academy, Ujjain,(M P) India, Vol. 2, pp. 13-30, 2007.

[10] Michael W Dixon and Terry W Koziniec and Murdoch University, Perth, Australia "Using OPNET to Enhance Student Learning in a Data Communications Course", InSITE - "Where Parallels Intersect" June 2002.

[11] Palis Michel & Rajasekaran Sanguthevar, "Emulation of a PRAM on Leveled Networks"Department of Computer and Information Science, Uniersity of Pennsylvania, Philadelphia, PA 19104-6389 and Wei David S. L., Department of computer Science, Radford University, Radford, VA 24142

[12] Quinn M.J., "Parallel Computing", McGraw-Hill INC, 1994.

[13] Rakov A Mikhail "Hyperstar and Hyperhub Optical Networks Interconnection Methods and Structures," US Patent Application No. 09/634 129, filed August 2000.

[14] Rakov, M, "Method of interconnection Nodes and a Hyperstar Interconnection Structures," US Patent 5 734 580 issued on March 1998.

[15] Ramsden, P. Context and strategy: Situational influences on learning. *In Schmeck, R.R. (ed) Learning Strategies and Learning Styles*. Plenum Press, NY, 1988.

[16] Saad, Y. and Schultz, M.H., "Topological prosperities of Hypercubes" IEEE transactions on computers, Vol. 37 No. 7 pp. 867-872, July 1988.

[17] Singer J., "A Theorem in Finite Projective Geometry and Some Applications to Number Theory," Trans. American Mathematical Society, Vol. 43, pp. 377-385, 1938.

[18] Tamiko, Teil, "The design of the connection machine" Design Issues, Vol. 10, Number, 1994.

[19] Weldon, E. J., Jr., Complexity of Peterson-Chien Decoders, unpublished memorandum, 1965

Modeling of Gravity Anomalies of Folds with Depth-Dependent Density Contrast

An undergraduate research experience

A. Abokhodair, A. Aldughaither, A. Al-Zaharni, and A. Al-Mulla
Department of Earthsciences
King Fahd University of Petroleum & Minerals
Dhahran, 31261, KSA

Abstract - *This paper documents a pilot undergraduate research project undertaken by a group of senior students mentored by a faculty of the Earth Sciences Department at KFUPM in Saudi Arabia. The objective of the project was to develop a mathematical gravity model of deep-seated structural folds (anticlines and synclines) with depth-dependent density. These structures are important in hydrocarbon exploration. At completion of the project, the participating students have gained first-hand experience with computational topics and tools that are not normally introduced in the undergraduate curriculum including: the use of computer algebra, semiautomatic differentiation tools, basic theory of nonlinear least squares, error propagation and quantification of uncertainty in parameters estimation.*

Keywords: Undergraduate research, gravity modeling, data inversion.

1 Introduction

Undergraduate research (UR) is an opportunity for students to enhance their learning experience, gain new skills, develop critical thinking, hone problem-solving abilities, and form relationships with mentors, professors, and other students. Modern computing and communication technologies available to students today empower them to carry out and produce high-quality research and communicate its results to the wider academic and professional communities. Many leading universities around the world now foster undergraduate research and integrate it into their regular undergraduate curricula (e.g. http://www.cur.org/).

The present paper documents a pilot UR undertaken by a group of three senior students supervised by a faculty at the Earth Sciences Department of King Fahd University of Petroleum and Minerals (KFUPM) in Saudi Arabia. At completion of the project, the participating students have gained first-hand experience and developed facility with the following computational topics and tools: (i) use of computer algebra (Matlab) to solve problem; (ii) use of semiautomatic differentiation tools based on complex variables (Abokhodair, 2007, 2009); (iii) Basic theory of nonlinear least squares; (vi) error propagation and quantification of uncertainty in parameters estimation problems. It is worth noting that these topics are not introduced in undergraduate courses and the graduate courses where the topics are presented are elective course.

The problem posed to the students consisted in developing a computational model of gravity anomalies of geological fold structures with depth-dependant density contrast for use in inversion and interpretation of gravity data. Anticlinal and synclinal folds are very important structures particularly in hydrocarbon exploration as they are known to host many of the world’s largest oil reservoirs such as Al-Ghawar and Abqaiq of Saudi Arabia (Afifi, 2005). Field studies show that the density of sedimentary rocks varies with depth (Maxant 1980; Peirce and Lipkov 1988; Litinsky, 1989). This variation is attributed to several factors including compaction, stratigraphic layering, facies change, cementation, diagenesis and tectonic history (Cordell, 1973). Therefore, a viable model for the gravity effect of such structures must take into account this dependence of density on depth. Of the various density-depth models reported in the literature, the students, based on analysis of density log data, selected a hyperbolic model (HDC).

2 Mathematical Details

An actual anticlinal fold is shown in figure 1 (top) which we approximate by an upright 2D isosceles triangle as shown in figure 1 (bottom). A thin-slab (cyan) of thickness dz', half-width x', at depth z' has a gravity effect given by:

$$\delta g_z = 2G\left[\tan^{-1}\left(\frac{x+x'}{z'}\right) - \tan^{-1}\left(\frac{x-x'}{z'}\right)\right]\Delta\rho\left(z'\right)\delta z'$$

where Δρ(z') is the depth-dependant density contrast given by:

$$\Delta\rho(z') = \frac{\beta^2 \Delta\rho_o}{\left(z'+\beta\right)^2}$$

Making the change of variables:

$$u = \frac{x+x'}{z'}; \;\; v = \frac{x-x'}{z'}$$

the gravity anomaly of the anticline is obtained from the integrals:

$$\Delta \mathrm{g}(\mathbf{p},\mathrm{x}) = \mathrm{p}_1 \int_{u(z_b)}^{u(z_t)} \frac{a\tan(u)}{\left(c_1+\beta u\right)^2}\,du - \mathrm{p}_2 \int_{v(z_b)}^{v(z_t)} \frac{a\tan(v)}{\left(c_2+\beta v\right)^2}\,dv$$

After integration and rearrangement of terms the anomaly may be written as (see Appendix for details):

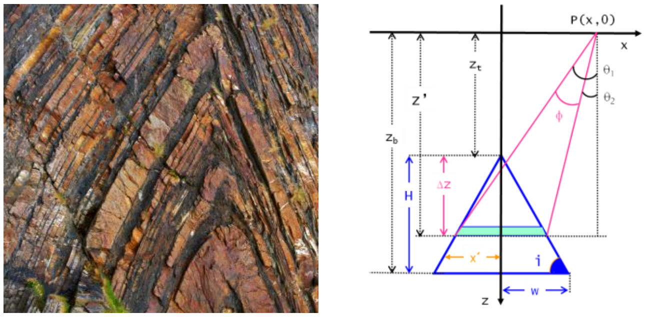


Figure 1: An actual anticlinal fold structure (left) and its approximating model (right).

$$\Delta \mathrm{g}(\mathbf{p},\mathrm{x}) = \mathrm{K}\left(T_1+T_2\right)$$

$$T_1 = Q_1 \ln\left(r_3/r_1\right) - Q_2 \ln\left(r_2/r_1\right) + \Delta Q \ln\left(s_1/s_2\right)$$

$$T_2 = R_3\Phi_3 - R_2\Phi_2 - R_1\Phi_1$$

3 Forward Modeling

It is instructive, as a validation test, to compare the response of the derived HDC model with that of the classical CDC model (e.g. Rao, et al.,1978) This is done in Figure 2, where the constant density contrast is taken as the mean of the HDC over the fold cross section. Clearly the two models diverge across the entire length of the profile. As expected, the HCD model response is consistently lower than the response of the CDC model. In the context of data inversion and parameters estimation, this systematic deviation contributes significant systematic errors in the estimated parameters from the CDC model. To explain, let the exact (HDC) fold model be $\mathbf{g}_e(\mathbf{p})$, the approximate (CDC) model be $\mathbf{g}_a(\mathbf{p})$ and the observed Bouguer data to be fitted be $\mathbf{d}_o$. Under the least-squares criterion, we attempt to minimize the norm of the residual: $\mathbf{r} = \left[\mathbf{d}_o - \mathbf{g}_a(\mathbf{p})\right]$. However, the residual may also be written as:

$$\mathbf{r} = \left[\mathbf{d}_o - \mathbf{g}_e(\mathbf{p})\right] + \left[\mathbf{g}_e(\mathbf{p}) - \mathbf{g}_a(\mathbf{p})\right]$$
$$= \mathbf{e}_d + \mathbf{e}_m$$

where $\mathbf{e}_d$ and $\mathbf{e}_m$ are, the random experimental error and the systematic modeling error respectively. Minimize the L_2 norm of the residual, therefore, may reduce the random component $\mathbf{e}_d$ but will have little or no effect on the systematic component $\mathbf{e}_m$. Thus $\mathbf{e}_m$ will ultimately propagate into and bias the estimated parameters of the fold.

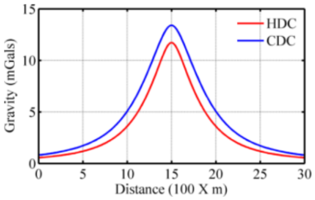


Figure 2: Gravity réponses of the HDC anticlinal fold model compared to the response of the classical CDC model.

4 Inverse Modeling

The HDC fold model was further validated in an inversion problem with simulated data. A set of gravity data was generated along a principal profile of an anticlinal fold using the HDC model with the parameters listed in Table 1. For the HDC function, we used the values $\Delta\rho_o$=1.0 gm/cc and β=2.5. The synthetic data was corrupted by additive errors N(0, σ) with σ = 15 per cent of the maximum anomaly peak. The inversion algorithm we used is based on Gauss-Newton method with Levenberg-Marquardt type damping (Neilson, 2010). The iterative step in this algorithm is determined by solution of the equation $\left(\nabla^2 f(\mathbf{p}) + \mu I\right) = -\nabla f(\mathbf{p})$, where $\nabla^2 f(\mathbf{p})$ and $\nabla f(\mathbf{p})$ are respectively the hessian $(\mathbf{H})$ and gradient ($\mathbf{g}$) of the objective function $f(\mathbf{p})$.

To compute the required derivative structures and , two approaches are available - numerical approximation and manual differentiation. Although the latter method is manageable using computer algebra tools, especially for small-scale problems such as the present one, manual differentiation is often avoided in actual practice for being time consuming, computationally expensive and error prone in programming. Therefore, we opted for numerical approximation methods and we used two techniques: ordinary finite difference (FD) and semiautomatic differentiation (SD) based on complex variables (Abokhodair, 2007, 2009). The results of the two inversion exercises are compared in Figure 3 and in Table 1.

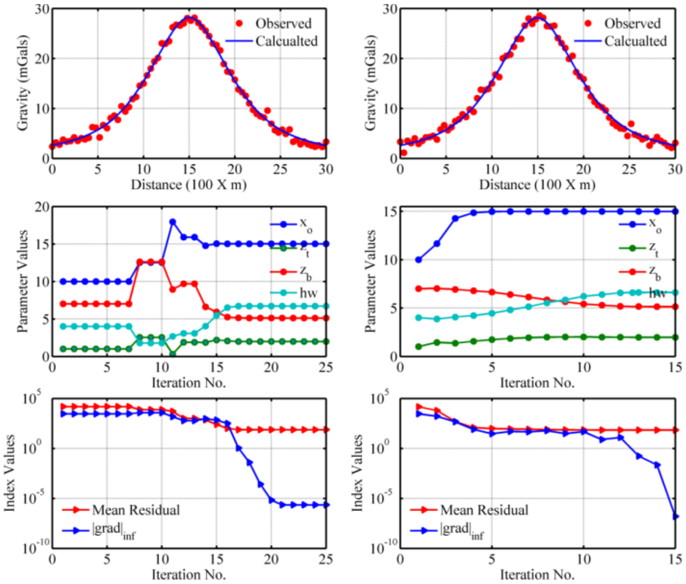


Figure 3: Inversion results using analytical Jacobian (left panel) and semiautomatic numerical Jacobian (right panel). Shown in the figure are the inverted and computed anomalies (top); the iteration paths of the model parameters (middle) and the convergence paths of the mean residual and the -

norm of the objective gradient.

The effect of the differentiation method on the progress of the inversion algorithm is clearly seen in Figure 3. As is well known, the accuracy of the FD method is highly sensitive to the differencing step-size, with a single optimum step size where it yields maximum accuracy of the order of 10^{-8}. In contrast, the SD method is independent of step size and has accuracy near machine epsilon (10^{-16}) (Abokhodair, 2009). Compared to the FD method, the superior performance of the SD method is reflected in the results of the SD-based inversion experiment by a small number of iterations and smoother convergence paths of all four parameters.

5 Error Propagation Analysis

The input data to the inversion algorithm is contaminated with a random error N(0,σ) which will propagate into the estimated parameters. The amount of errors propagated into the final solution is best measured by the covariance matrix of the solution. However, because the mapping from data space to parameters space is nonlinear, it is difficult to estimate this covariance matrix exactly. The next best alternative, therefore, is to approximate the covariance matrix from a linearized form of the residual function as follows. At the kth iteration step, the residual function in the neighborhood of $\mathbf{p}_k$ may be written as:

$$\begin{aligned}\mathbf{r}(\mathbf{p}) &= \mathbf{r}(\mathbf{p}_k + \delta\mathbf{p}) = \mathbf{d}_o - \mathbf{f}(\mathbf{p}) \\ &\simeq \left[\mathbf{d}_o - \mathbf{f}(\mathbf{p}_k)\right] - \mathbf{J}(\mathbf{p}_k)\,\delta\mathbf{p} \\ &= \delta\mathbf{d} - \mathbf{J}(\mathbf{p}_k)\,\delta\mathbf{p}\end{aligned}$$

where $\mathbf{J}(\mathbf{p}_k)$ is the Jacobian of $\mathbf{r}(\mathbf{p}_k)$. Least squares minimization of this linear approximation of the residual leads to the normal equations:

$$\delta\hat{\mathbf{p}} = \left(\mathbf{J}^T\mathbf{J}\right)^{-1}\mathbf{J}^T\delta\mathbf{d}$$

where we dropped the subscript and replaced it by the caret to indicate the final solution of the parameters estimates. Letting $\mathbf{Q} = \left(\mathbf{J}^T\mathbf{J}\right)^{-1}\mathbf{J}^T$ and taking the expectation of both sides of the above equation leads to:

$$\begin{gathered}E\left(\delta\hat{\mathbf{p}}\,\delta\hat{\mathbf{p}}^T\right) = E\left(\mathbf{Q}\,\delta\mathbf{d}\delta\mathbf{d}^T\,\mathbf{Q}^T\right) = \mathbf{Q}\,E\left(\delta\mathbf{d}\delta\mathbf{d}^T\right)\,\mathbf{Q}^T \\ \mathbf{C}_{\hat{p}} = \mathbf{Q}\,\mathbf{C}_{d_o}\mathbf{Q}^T\end{gathered}$$

If $\mathbf{C}_{d_o} = \sigma^2\mathbf{I}$ as in the present case, then the covariance matrix of the estimated parameters is given by:

$$\mathbf{C}_{\hat{p}} = \sigma^2\left(\mathbf{J}^T\mathbf{J}\right)^{-1} = \sigma^2\mathbf{H}^{-1}(\hat{\mathbf{p}})$$

The standard deviations of the estimated parameters are listed in table 1.

6 Discussion and Conclusions

Upper level undergraduate students, who have spent several years enrolling in courses heavily dependent on textbook learning, find undergraduate research with its hands-on, self-paced learning format a welcome break from the ordinary. This is the feedback I got from the students who took part in this pilot undergraduate research experiment. It is my conclusion that if challenged with relevant problems that arouse their curiosity, students can and do ask scientifically interesting questions. Moreover, with the right tools and advice, students can and do approach solutions to posed problems like scientists and consequently produce good quality research.

The work reported here is an illustrative example of what undergraduate students can accomplish given the opportunity and means. The problem posed to the participants has relevance in exploration geophysics, and during the three-month duration of the activity, the students accomplished their mission to the fullest. They derived a gravity anomaly model of geological fold structures with depth-dependant density contrast, thoroughly tested the model in forward and inverse modeling exercises, compared the model response to the classical constant-density model, evaluated the model performance in inversion under different differentiation schemes and finally carried out a sophisticated error propagation analysis of the estimated parameters. That is a full fledged applied research.

7 Acknowledgements

This research work was conducted by the undergraduate research group at the Department of Earth Science, King Fahd University of Petroleum and Minerals, (KFUPM), Dhahran, Saudi Arabia. We thank our mentor Dr. A. Abokhodair for his diligence while supervising the group and for his valuable suggestions while reviewing the manuscript.

8 References

Afifi, A. M., 2004: Ghawar- The Anatomy of the World's Largest Oil Field. www.searchanddiscovery.com/documents/2004/afifi01/index.htm, (2004).

Abokhodair, A. A.: Numerical tools for geoscience computations- Semiautomatic differentiation (SD). Computational Geoscience 11, 283-296 (2007).

Abokhodair, A. A.: Complex differentiation tools for geophysical inversion. Geophysics 74(2), H1-H11 (2009.).

Maxant J.: Variation of density with rock type, depth, and formation in the Western Canada basin from density logs. Geophysics 45, 1061–1076 (1980).

Peirce J.W. and Lipkov L.: Structural interpretation of the Rukwa Rift, Tanzania. Geophysics 53, 824–836 (1988)

Rao, B. S. R., and Murthy, I. V. R.: Gravity and magnetic methods of prospecting: Arnold-Heinemann Publishers Pvt. Ltd (1978).

Litinsky, V. A.: Concept of effective density: Key to gravity depth determinations for sedimentary basins. Geophysics, 54, 1474–1482 (1989).

Cordell L.: Gravity anomalies using an exponential density-depth function -San Jacinto Graben, California. Geophysics 38, 684–690 (1973)

Rao, B. S. R., and Murthy, I. V. R.: Gravity and magnetic methods of prospecting: Arnold-Heinemann Publishers Pvt. Ltd (1978).

9 Appendix

$$k = 2G\beta^2 \Delta p_0$$

$$\frac{1}{k}\Delta g_z(p, x) = T_1 + T_2$$

$$T_1 = Q_1 \ln(r_3 / r_1) - Q_2 \ln(r_2 / r_1) + \Delta Q \ln(s_t / s_b)$$

$$T_2 = R_3\varphi_3 - R_2\varphi_2 - R_1\varphi_1$$

$$Q_1 = \frac{P_1}{C_1^2 + \beta^2}\,; \quad P_1 = x - \alpha z_t\,;$$

$$C_1 = x - \alpha s_t \;; \quad s_t = \beta + z_t$$

$$Q_2 = \frac{P_2}{C_2^2 + \beta^2} \;; \; P_2 = x + \alpha z_t;$$

$$C_2 = x + \alpha s_t \;; \quad s_t = \beta + z_t$$

$$\Delta Q = Q_1 - Q_2$$

$$r_1 = \sqrt{x^2 + z_t^2} \;;$$

$$r_2 = \sqrt{(x - w)^2 + z_b^2}$$

$$r_3 = \sqrt{(x + w)^2 + z_b^2};$$

$$\varphi_1 = \frac{\pi}{2} + tan^{-1}\left(\frac{x}{z_t}\right)$$

$$\varphi_2 = \frac{\pi}{2} + \tan^{-1}\left(\frac{x - w}{z_b}\right)$$

$$\varphi_3 = \frac{\pi}{2} + \tan^{-1}\left(\frac{x + w}{z_b}\right)$$

$$R_3 = Q_1 q_1 \;\; R_2 = Q_2 q_2$$

$$R_1 = Q_2 q_{02} + Q_1 q_{01}$$

$$q_{02} = \frac{\beta z_t - C_2 x}{C_2 z_t + \beta x} \qquad q_{01} = \frac{\beta z_t - C_1 x}{C_1 z_t + \beta x}$$

$$q_2 = \frac{\beta z_b - C_2 x^-}{C_2 z_b + \beta x^-} \qquad q_2 = \frac{\beta z_b - C_1 x^+}{C_1 z_b + \beta x^+}$$

$$x^- = x - w; \;\; x^+ = x + w;$$

Assessment Agent based ABET Criterions

Abbas M. Al-Bakry
Software Dept.,University of Babylon, Babylon, Hilla, Iraq

Abstract-Assessment is the important process for testing the ability of the institution program to meet the required criterions to get the accreditation. In this work we deals with ABET quality assurance agency dedicated with applied science, computing, engineering and technology education and we concerned with computer science program. A software agent, which provide difference facilities developed to achieve the assessment function. The developed agent(Assessment Agent) perceive the current institution program status from the system environment, then perform sequence control instructions included two cycles of verifications to generate the Self-Assessment Report(SAR) and advices for getting accreditation.

Keywords: ABET criterion, assessment agent, quality assurance.

1 Introduction

Improving professional standards for a number of years has been developed by Quality Management Systems in Higher Education (QMSHE). In order being quality minded in higher education means caring about the expectations of students and other customers as well as all involved parties, and ensuring they are met. Students perceptions thus provide important information for lecturers if learners needs are to be fulfilled. An assessment of the quality of teaching programs comes at a time when the concern for quality in higher education is probably at an all time high. All processes in any organization (higher education institution) contribute directly or indirectly to quality as the customer (student) defines it. This will determine whether students' needs have been met. Quality systems in higher education have been important for decades. They help us improve professional standards by comparing them with international educational qualifications[10]. Assessment is something which is 'experienced', though not always positively, by almost all involved in a higher education institution: students, teachers, administrators, managers, policy makers, institutional leaders, library staff, the students union and those supporting students in professional contexts in which learning is assessed, e.g. the nursing supervisor, teaching mentor or manager of the placement student in business. Given this complexity surely it is important for us to try to understand the relationships between these groups in some systematic way[6]. The following authors deals with ABET organization criterion and assessment procedure: Charies R. Lang and Hakan Gurokan[Char08] presents the curricular structure and assessment methods of a new science program and describes how it has met ABET-CACs current criteria. James Collofello[7] begin to discussed the similarities of Capability Maturity Model CMM and ABET. Both technical and nontechnical similarities such as the impact of employee-faculty buy in and management- administrative support will be addressed. Eugene Essa et al. [5] focuses on ABET program. They require that programs show student achievement and certain course outcomes. Documentation of this requirement is particularly burdensome. Deborah A. Trytten [4] Fined that the Exhibit Collection and Analysis Tool (ECAT) was created to electronically store and organize display materials for our program in a manner that was useful for ABET program evaluators.. .

2 ABET Organization

ABET is a United State association ,it was founded in 1932 as the Engineers' Council for Professional Development (ECPD), an engineering professional body dedicated to the education, accreditation, regulation, and professional development of the engineering professionals and students in the United States. It was headquartered at the Engineering Societies Building and then the United Engineering Center in New York City until it relocated to Baltimore in 1996.

2.1 ABET general criterion

The ABET agency focused on eight general criterion starts with students and finished by the institutional support, table(1) present the general ABET criterions which are Students, Program Educational Objectives, Student Outcomes, Continues Improvement, Curriculum, Faculty, and Facilities [1][3].

TABLE 1: The General ABET citerions

Creterion Name	Describtion
Students	The student performance must be evaluated. Student progress must be monitored to foster success in attaining student outcomes, thereby enabling graduates to attain program educational objectives. Students must be advised regarding curriculum and career matters.
Program Educational Objectives	The program must have published program educational objectives that are consistent with the mission of the institution, the needs of the program's various constituencies, and these criteria. There must be a documented and effective process, involving program constituencies, for the periodic review and revision of these program educational objectives.
Student Outcomes	The program must have documented student outcomes that prepare graduates to attain the program educational objectives. There must be a documented and effective process for the periodic review and revision of these student outcomes. The program must enable students to attain, by the time of graduation: (a) An ability to apply knowledge of computing and mathematics appropriate to the discipline (b) An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution (c) An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs (d) An ability to function effectively on teams to accomplish a common goal (e) An understanding of professional, ethical, legal, security and social issues and responsibilities (f) An ability to communicate effectively with a range of audiences (g) An ability to analyze the local and global impact of computing on individuals, organizations, and society (h) Recognition of the need for and an ability to engage in continuing professional development (i) An ability to use current techniques, skills, and tools necessary for computing practice.
Continuous Improvement	The program must regularly use appropriate, documented processes for assessing and evaluating the extent to which both the program educational objectives and the student outcomes are being attained. The results of these evaluations must be systematically utilized as input for the continuous improvement of the program. Other available information may also be used to assist in the continuous improvement of the program.
Curriculum	The program's requirements must be consistent with its program educational objectives and designed in such a way that each of the student outcomes can be attained. The curriculum must combine technical and professional requirements with general education requirements and electives to prepare students for a professional career and further study in the computing discipline associated with the program, and for functioning in modern society.
Faculty	Each faculty member teaching in the program must have expertise and educational background consistent with the contributions to the program expected from the faculty member.
Facilities	Classrooms, offices, laboratories, and associated equipment must be adequate to support attainment of the student outcomes and to provide an atmosphere conducive to learning.

2.2 ABET criterion for computing programs

These program criteria apply to computing programs using computer science or similar terms in their titles and Table(2) present the specific criterions for the Computing program included Program Outcomes, Curriculum, and Faculty Qualifications[1].

TABLE 2: Computing program citerions

Creterion Name	Description
Program Outcomes	The program enables students to achieve, by the time of graduation: (j) An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices. (k) An ability to apply design and development principles in the construction of software systems of varying complexity.
Curriculum	Students have the following amounts of course work or equivalent educational experience: a. Computer science: One and one-third years that includes: 1. coverage of the fundamentals of algorithms, data structures, software design, concepts of programming languages and computer organization and architecture. 2. an exposure to a variety of programming languages and systems. 3. proficiency in at least one higher-level language. 4. advanced course work that builds on the fundamental course work to provide depth. b. One year of science and mathematics: 1. Mathematics: At least one half year that must include discrete mathematics. The additional mathematics might consist of courses in areas such as calculus, linear algebra, numerical methods, probability, statistics, number theory, geometry, or symbolic logic. 2. Science: A science component that develops an understanding of the scientific method and provides students with an opportunity to experience this mode of inquiry in courses for science or engineering majors that provide some exposure to laboratory work.
Faculty Qualifications	Some full time faculty members have a Ph.D. in computer science.

3 Software agent and its applications

The software agent define as " an autonomous system that receive information from its environment, processes it, and perform actions on that environment". And according to According to J. Ferber , an agent is a real or abstract entity, capable of acting on itself and on the environment. It can have a partial representation of this environment. It can communicate with other agent. And finally, its behavior is the result of its observations, its knowledge and its interactions.

As a result the definitions globally cover about the same scope. Thus, concerning software agents, it may be both more practical and more useful to define it by a set of characteristics. The principle structure of the software agent presented as in fig. 1. The agent can be activated from the environment through the sensors, perform sequence of instructions and return the results to the environment through the actuators [2].

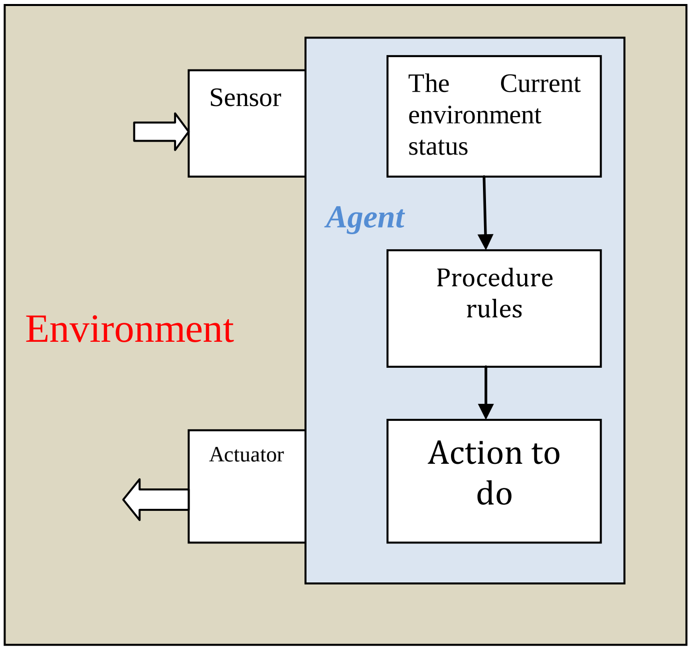


Fig. 1 **The** principle structure of the software agent

Putsadee Pornphol**[9]** presents an agent-based quality assurance assessment system for educational institution. Agents are used to check essential requirements that educational institution clients have to meet and prepare reports for assessors. The system proves useful and helps reduce assessment time from education expert assessors. Toshiyuki Sueyoshi and Gopalakrishna Reddy Tadiparthi[11] create an artificial wholesale market, where many different traders are equipped with learning capabilities. They validate the agent based model with the help of a data set from PJM electricity market. Han Yu, Zhiqi Shen and Chunyan Miao[Intelligent Software] proposes the Goal Net Designer which is an Integrated Development Environment (IDE) for modeling agent behavior based on Goal Net model, a goal orient methodology. Jianhua Liu[8] designed an agent-oriented software design method. The Assessment Agent System is composed of five types of software agents: instructor agent, student agent, management agent, assessment agent, and reporting agent.

4 The Agent Based Assessment Agent

The proposed system designed for doing the self assessment procedure for the computer science institutions and institution have a similar term in this title. The designed system based on a software agent that perform the automatic auditing by perceiving the current institution status the doing a sequence of control instructions to achieve the self assessment report(SAR). The SAR is the main action for the assessment agent beside the advice massages for accreditation level.

The system consists of three phases as presented in fig. 2. In the first phase agent verify the institution data base against the ABET general criterion data base(see ABET general criterion) if the verification to achieve the student outcome(for example ***a* to *i***). The next phase verify the data resulted from previous phase to achieve the remaining conditions (***j* to *k***). In the third phase a new assessment institution data base generated. The system result is the SAR and report to us an advice of the institution accreditation level.

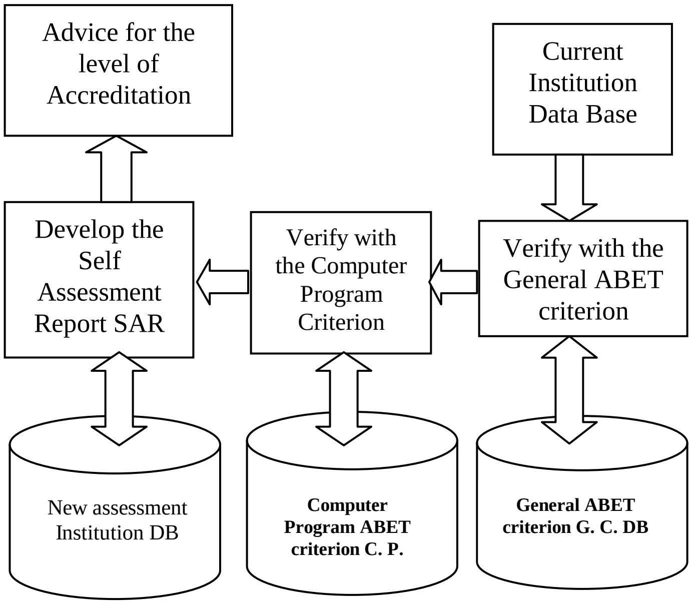

Advice for the level of Accreditation
Current Institution Data Base
Develop the Self Assessment Report SAR
Verify with the Computer Program Criterion
Verify with the General ABET criterion
New assessment Institution DB
Computer Program ABET criterion C. P.
General ABET criterion G. C. DB

Fig. 2 The Structure of the proposed system

5 The Assessment Agent Structure

The structure of the agent system based computing criterion presented in the figure(2). The assessor agent activated by receiving a signal to read the data base of the current institution to be tested, the fetched data verified against the ABET general criterion. The decision is advice to refine if the institution data not meet the requirement of ABET general conditions. From the other hand if the institution data meet the requirement of ABET general conditions then verified with the specific data (i.e. Computer Program) and again if the verification success to meet the specific program requirements then generate the Self Assessment Report SAR included in details all the institution states, the assessment data base for the current institution also developed and report advices for the institution to achieve the advancement.

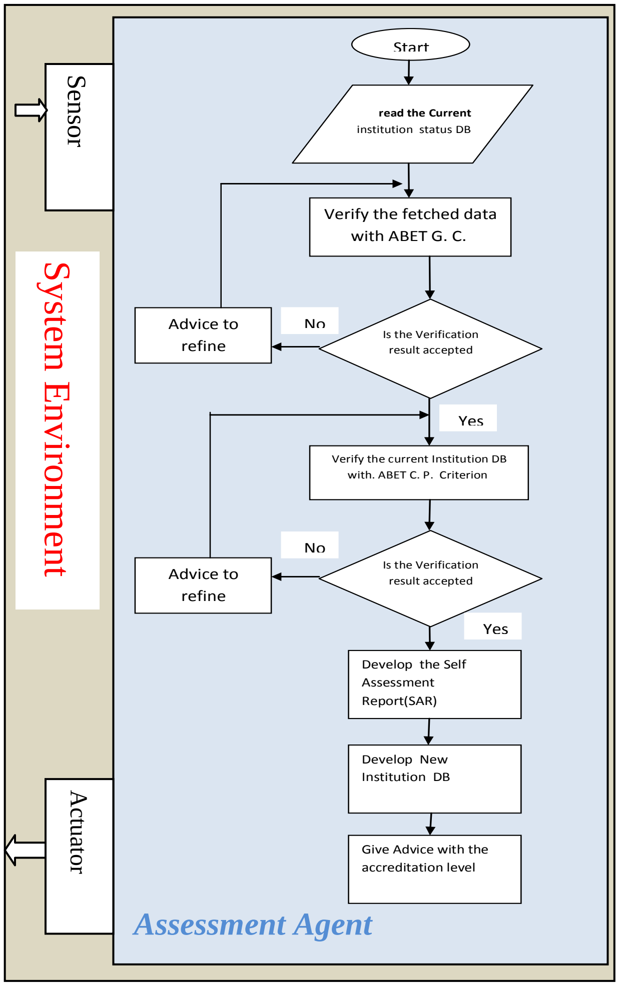


Fig. 3 Structure of the Assessment Agent

6 The Assessment agent algorithm

The assessment agent algorithm presented in figure(4) as an abstract instruction included *state*, *procedure and action*. The state represent a description for the current institution state and the procedure described by set of control instructions used to developed new assessment data base of the current institution to assess. The action which is the represent the results returned to the environment, the action is to develop SAR, and multi advices for upgrading and achieve enhancement for the institution.

Function Assessment-Agent(percept) return an action

__State__: Description of the current institution Data Base

__Procedure__: set of control instructions to develop new institution Data Base

__Action__: Developing the SAR and Advice for the level of the accreditation

State __is__ MODIFY current state (state, action, percept).
Procedure __is__ VERIFICATION and UPDATE(state, procedure).
Action __is__ VERIFICATION result(procedure)
Return action

Fig. 4 Abstract Instructions of the Assessment Agent

7 Conclusions

The system achieve an active solution for the main challenge in the quality assurance program in higher edication institutions which assessment problem. The assessment procedure done before by team work od assessors and they achieve their work in a multitages.

The assessment function in the developed system achieved by the software agent(Assessment Agen), the autonomus feature of the agent facilitate the self auditing and enhancemet. The assessment agent precieve the current computing program status data base, perform two cycles of verifications then generate an efficient assessment data base for the current institution in the way to generate accreditation advices(enhancement recomondations) and develp the SAR.

8 References

[1] Accreditation Policy and Procedure Manual, Effective for Evaluations During the
2011-2012 Accreditation Cycle, 2012.

[2] Charles R. Lang and Hakan Gurocak, Assessment Methods for the Upcoming ABET Accreditation Criteria for Computer Science, 38th ASEE/IEEE Frontiers in Education Conference, 2008.

[3] Christophe Pincemaille, Intelligent agent technology, Cork Institute of Technology, 2008.

[4] Deborah A. Trytten, ECAT: An Electronic Data Analysis Tool for ABET Display Materials, 40th ASEE/IEEE Frontiers in Education Conference, 2010.

[5] Eugene Essa et al., ACAT: A Web-based Software Tool to Facilitate Course Assessment for ABET Accreditation, Seventh International Conference on Information Technology, 2010.

[6] Gibbs, G. Improving university teaching and learning through institution-wide strategies. Paper presented at Teaching and Learning in Higher Education: New Trends and Innovations. University of Alverno, 13-17 April, 2003.

[7] James Collofello, Applying Lessons Learned from Software Process
Assessments to ABET Accreditation, 34th ASEE/IEEE Frontiers in Education Conference, 2004.

[8] Jianhua Liu, The Assessment Agent System: Assessing Comprehensive Understanding Based on Concept Maps, Ph.D. thesis, Virginia Polytechnic Institute and State University, 2010.

[9] Putsadee Pornphol, An Agent-based Quality Assurance Assessment System, 5th WSEAS International Conference on E-ACTIVITIES, Venice, Italy, November 20-22, 2006.

[10] Quality Management Systems in Higher Education,, University of Primorska, Armand Faganel and Slavko Dolinšek, Slovenia, 2003.

.

[11] Toshiyuki Sueyoshi and Gopalakrishna Reddy Tadiparthi, Intelligent Agent Technology: An Application to US Wholesale Power Trading, IEEE/WIC/ACM International Conference on Intelligent Agent Technology, 2007.

SESSION

LATE BREAKING PAPER: PARALLEL COMPUTING, CLOUD COMPUTING, SOFTWARE ENGINEERING AND PROGRAMMING + GROUP LEARNING

Chair(s)

Prof. Hamid Arabnia
University of Georgia

Haskell as an Introduction to Parallel Computing for Undergraduates

Kyle Barton, Samuel Dodson, Danielle Fenske,
Benjamin Whitehead, Jens Mache
{kjbarton, sdodson, dfenske, benw, jmache}@lclark.edu
Department of Mathematical Sciences,
Lewis & Clark College, Portland, OR, USA

Abstract—*As computing manufacturers increasingly rely on the use of multiple cores instead of improved clock speeds, programming in parallel has become an essential skill for computer scientists. Insufficient effort has been made to introduce the concept of parallel computing to the undergraduate curricula. We explored the pedagogical potential of a functional programming language, Haskell, in the context of parallel computing education. We extended previous work by providing both sequential and parallel Haskell implementations of a program that uses Riemann integration to calculate* π.

Our assessment is that the parallel tools of Haskell are both concise and scalable. However, they can be difficult to learn, which brings us to our one reservation about using Haskell as a parallel programming language at an undergraduate level: learning functional programming may distract students from the original focus of learning general parallel programming concepts. As such, we expect that a module on parallel Haskell programming would be more effective if it was introduced in a programming languages course, which would presumably have already introduced students to the basic concepts of functional programming.

Keywords: parallel computing, Haskell, computer science education

1. Introduction

Over the course of the last decade, advancement in computer processor speed has noticeably slowed. As circuitry speed begins to run up against the speed of light, it becomes increasingly clear that Moore's Law cannot be maintained with a single-CPU computing paradigm. Engineers have mitigated the slowing rate of increasing processor speed by building computing systems that include more than one core processor. Today, modern personal computers have anywhere from two to eight CPUs, and industrial-grade servers include processing blades with dozens. However, multiple core systems cannot deliver adequate improvements in processing speed without software support. Programmers must alter traditional, singular coding methods to properly take advantage of multiple cores. This is done by splitting process workloads into independent tasks that can be accomplished in parallel. However, while good practices in parallel programming are vital to improving processing power, very little attention is paid to the discipline at the undergraduate level [1], [5]. Given the fact that parallel computing is a relatively new practice, no one language has emerged as the preferred parallel environment. Therefore this summer, we explored various parallel programming languages with respect to their pedagogical value for undergraduate computer science students[1]. Instead of focusing on one language that may become obscure in the coming years, it is more advantageous for students to learn a variety of languages, with the goal of understanding the benefits and downsides of each, while adopting a “parallel mindset" [6]. We searched for languages that were powerful, accessible, and informative about the nature of parallel programming in today's world of software engineering.

1.1 Haskell

Haskell was conceived as early as the 1980s, and a stable version was released in 2003 [2]. It is a purely functional language with a static, strong type system that incorporates automatic type inference. Haskell uses “lazy" evaluation in addition to optimized tail-call recursion in order to make recursive calls less resource-consuming. It also has built-in, convenient parallel interfaces which lead us to ask the question: can a functional programming paradigm be an effective environment in which to teach undergraduate students parallel computing?

When considering this question, educators must weigh the functionality and scalability of the parallel tools in Haskell against the overhead and possible distraction of teaching young programmers a new type of programming language structure. We decided to explore Haskell in parallel by implementing some simple, embarrassingly parallel problems demonstrated in the ACM Parallel Computing TechPack [7], namely, using Riemann integration of the unit circle in order to estimate π. We implemented this program with an eye for both scalability and teach-ability. Ideally, a parallel language strikes a balance such that its tools are powerful enough to run more complex systems in parallel, but are still simple enough to teach to undergraduate computer science students.

1.2 Parallel Computing

For more computationally expensive problems, a speedup can be achieved by dividing the solution into pieces that can be evaluated on multiple cores at once. While automatic parallelization is still not possible, Haskell is able to remove some of the annoying problems associated with traditional parallel computing techniques.

Haskell programs are deterministic, meaning that they always produce the same answer, so parallel programs can be run, tested, and debugged on a single core. Additionally, the programmer does not have to worry about classic problems like race conditions or deadlocks that are hard to detect and debug [3]. The low-level details of parallelism are abstracted in Haskell, so programs are more likely to run on a range of hardware. Additionally, Haskell uses advanced runtime systems, like parallel garbage collection[3].

The main challenge with programming in parallel is finding the optimal partition of the problem. In other words, the problem needs to be divided into pieces that can be computed in parallel. There should be enough pieces that each processor or core is always busy [3]. This process of optimization can be achieved through a trial and error process.

2. Quicksort

As an introduction to the Haskell programming language, below are sequential and parallel implementations of the Quicksort algorithm [8]. For the readers not familiar with this sorting algorithm, a description of it follows. The algorithm sorts a list of integers. An element from the list is chosen to be the pivot value. A sublist of all the elements less than the pivot is created, and recursively sorted. A sublist of all the elements greater than or equal to the pivot is also created, and recursively sorted. Then the two sorted sublists are appended.

2.1 Sequential Quicksort

The sequential version of Quicksort is seen as one of the best examples of the elegance of the Haskell programming language, because implementations of the Quicksort algorithm in other programming languages, such as C, C++ or Java, are usually much longer.

Starting at line 2, the code sequentially executes, going through each case until it reaches one that applies to the given situation. The first guard, on line 2, describes the case of a non-empty list. The last case, on line 5, checks for an empty list, in which case the empty list is returned, because it is already sorted. If the set is nonempty, our sorted array will be found by recursively calling sort on the smaller and larger lists.

The `where` keyword, on line 3, is placed directly after a statement that needs to use the definitions stated in the where block. A `where` block provides local definitions so that there is no danger of ambiguity or reusing names in other parts of the program. On line 3 the set of all `xs` in the set which are less than or equal to the pivot is sorted recursively until we reach the case of an empty set, and then we retrace our steps through the recursion and concatenate the smaller lists to the larger lists. We do the recursion and evaluation in this order because Haskell is a lazy language, and will not evaluate statements until it is absolutely necessary. Haskell will save a list of functions to evaluate and not until it reaches the end of the recursive loop will it finally go back and find the outcomes resulting from each pass.

```
1 sort :: (Ord a) => [a] -> [a]
2 sort (x:xs) = smaller ++ x:larger
3   where smaller =
        sort [y | y <- xs, y <  x]
4       larger =
        sort [y | y <- xs, y >= x]
5 sort _ = []
```

2.2 Parallel Quicksort

With the addition of three functions the sequential Quicksort can be written in parallel. The main difference is that the code takes advantage of the `par` and `pseq` functions, which can be found in the `Control.Parallel` modules. The `par` function takes the argument to its left and begins working on it in parallel, while moving on to the right argument. The `pseq` function is almost the opposite of `par`. When we call `pseq`, the function requires that its left argument finish before it allows the right argument to execute. The force function simply prevents lazy evaluation, so that the left argument is evaluated eagerly.

```
import Control.Parallel
parSort :: (Ord a) => [a] -> [a]
parSort (x:xs) =
     force larger `par`
     (force smaller `pseq`
     (smaller ++ x:larger))
  where smaller =
      parSort [y | y <- xs, y <  x]
        larger =
      parSort [y | y <- xs, y >= x]
parSort _ = []
force :: [a] -> ()
force xs = go xs `pseq` ()
where go (_:xs) = go xs
      go [] = 1
```

3. Riemann Integral

For demonstrating some of the features of Haskell, an instructor could show a simple program that will estimate π by approximating the area under half of the unit circle and multiplying it by two. This has been a popular introduction to parallelism and has been used in the ACM Parallel

Computing TechPack [7]. Our code could be a nice addition to this previous educational work, which has focused on more traditional parallel programming languages.

These programs are good class examples because they utilize the list features of Haskell, they are easy to relate to, and the transition to parallel code is fairly straightforward.

3.1 Sequential Riemann

The program will take an argument for the number of partitions and return an estimation of π. It will do this by the method of right-handed Riemann rectangle summation. To implement this sum we do the following. First we create a list that has an appropriate dx based on the number of partitions the user inputs. We then multiply dx by twice the height of the right hand point to get the area of our rectangle. We then add up all of the area of the rectangles to get our approximation.

```
-- Calculate upper hemisphere
   of the unit circle
circle :: Double -> Double
circle x = sqrt (abs(1 - x^2))
-- Calculate the area of
   right Riemann rectangle
area :: Double -> Double -> Double
area x1 x2 = (x2 - x1) * circle x2
-- Recursively add the areas
   of the Riemann rectangles
estimate (x:[]) = 0
estimate (x:y:xs) =
    (area x y) + estimate (y:xs)
```

3.2 Parallel Riemann

The parallel version is almost identical code, using a similar recursive function to add the areas of the Riemann rectangles. The primary difference comes from the insertion of the `par` and `pseq` functions. In our parallel estimation of π, `par` is calculating smaller, and `pseq` is calculating larger at the same time. Larger makes a recursive call to `parEstimate`, giving us another smaller section to begin executing in parallel. This essentially gives us a cascading sum of parallel computations of the areas of the Riemann rectangles. Once larger –with the recursive smallers– is finally complete, larger and smaller are added together, Resulting in π.

```
import Control.Parallel
-- Calculate upper hemisphere
   of the unit circle
circle :: Double -> Double
circle x = sqrt (abs(1 - x^2))
-- Calculate the area of
   right Riemann rectangle
area :: Double -> Double -> Double
area x1 x2 = (x2 - x1) * circle x2
-- Recursively add the areas
   of the Riemann rectangles
parEstimate :: [Double] -> Double
parEstimate (x:[]) = 0
parEstimate (x:y:[]) = area x y
parEstimate (x:y:xs) =
smaller `par`
  (larger `pseq` smaller + larger)
  where smaller = area x y
        larger = parEstimate (y:xs)
```

3.3 Long-term Assignments

The Riemann integral is a nice introduction to Haskell and the power and simplicity of its parallel computing packages, but the ideas could be further explored in a long-term homework assignment [6]. There are a wide variety of embarrassingly parallel programs that could be applied using the Haskell tools described here. Instructors could assign as long term projects various forms of population simulations, such as the Game of Life. These would be excellent, challenging problems for young programmers that would introduce them to slightly more complex Haskell code. In the case of such an assignment, it might be beneficial to provide to students easy, intuitive graphics packages, so that students could focus more on the parallel, back-end aspect of their codes, rather than the challenge of creating good visualizations.

4. Discussion

In conclusion, we have found that Haskell is a programming language that can be used effectively in order to teach undergraduates about parallel computing. The parallel patterns in Haskell, like `par` and `pseq`, are straightforward and should therefore take little time to introduce to students. The fact that the number of cores to be used in the program can be specified when compiling means that parallel computing is scalable.

Our parallel π estimate, for example, should be able to take advantage of massively parallel threading systems, while also working on any personal computer. These properties make the parallel options in Haskell both easy to learn and powerful to use. There is, however, some question as to whether the language should be included in the curriculum of a course dedicated to parallel computing. While functional programming is compelling and powerful, it is also very different from any of the imperative languages that most undergraduates are familiar with. As such, the learning curve with Haskell can be steep, and may distract students from the focus of learning good practices in parallel computing. One solution could be to include a module on parallel computing with Haskell in a programming language structures course, rather than in a course dedicated solely to parallel computing. Therefore if students already understand functional programming, they can be focused on learning the parallel part of the code, deriving the maximum educational benefit from the work shown here.

5. Future Work

We would like to explore more challenging parallel problems. One interest is in Boids simulations which are used to simulate the flocking of birds [4]. This is an embarrassingly parallel problem. It may be a good introduction to parallel computing, because the visualization of the algorithm is attractive.

The simulation makes parallelization accessible to students because the idea that each thread is an object, or boid, on the screen will ultimately lead to better understanding of the theory as a whole. This simulation would introduce students to graphics in Haskell, while still keeping the fundamentals of the project relatively simple. This style of project has been shown to be particularly useful in educating students about parallel programming, because they will not get hung up on details of the code, but instead be able to grasp the overall ideas and advantages of parallelism.

We would also like to implement this tutorial in a programming languages and parallel computing class at Lewis & Clark College in the fall of 2013 and spring of 2014, respectively.

6. Acknowledgments

Partial support for this work was provided by the National Science Foundation's "Transforming Undergraduate Education in Science, Technology, Engineering and Mathematics (TUES)" program under Award No. 1044932, by the John S. Rogers Science Research Program and by the James F. and Marion L. Miller Foundation. We would also like to thank David Bunde and Jingke Li.

References

[1] Joel Adams, Richard Brown, and Elizabeth Shoop. Patterns and exemplars: Compelling strategies for teaching parallel and distributed computing to CS undergraduates. In *EduPar-13*, 2013.

[2] Miran Lipovača. *Learn you a Haskell for great good! A beginner's guide.* No Starch Press, San Francisco, CA, 2011.

[3] Simon Marlow. *Parallel and concurrent programming in Haskell.* O'Reilly & Associates Inc, Sebastopol, CA, 2013.

[4] Craig Reynolds. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, 1987.

[5] Wilson Rivera. How to introduce parallelism into programming languages courses. In *EduPar-13*, 2013.

[6] Suzanne Rivoire. A breadth-first course in multicore and manycore programming. In *Proceedings of the 41st ACM technical symposium on Computer science education*, 2010.

[7] Paul Steinberg and Matthew Wolf. ACM parallel computing TechPack. http://techpack.acm.org/parallel/. Accessed: 2013-05-30.

[8] Bryan Sullivan. *Real world Haskell.* O'Reilly & Associates Inc, Sebastopol, CA, 2009.

Computing Education on Cloud

Ying Xie, Ken Hoganson
Department of Computer Science, Kennesaw State University, Kennesaw, Georgia, USA

Abstract - *Guided by the vision of engaging diverse student populations, especially students from the underrepresented groups, in an increasingly computationally empowered workforce, we launched a "Computing Education on Cloud (CEC)". The CEC project aims to deliver critical computing education as cloud services, which can be accessed anywhere, anytime, by any registered user with minimum requirement of resources. In this paper, we identified three categories of critical computing education and describe the solutions towards some challenging issues in delivering open courseware in the identified categories as cloud services.*

Keywords: Cloud Computing, Computing Education, Open Courseware, Education as a Service

1 Introduction

The National Science Foundation (NSF) has launched a funding program that is called Computing Education for the 21st Century (CE21), which aims to "build a robust computing research community, a computationally competent 21st century workforce, and a computationally empowered citizenry" [1]. Being both educators and researchers in computing, we share the same vision of the NSF CE21 program on engage diverse student populations, especially students from the underrepresented groups, in an increasingly computationally empowered workforce.

As stated in the NSF CE21 Proposal Solicitation, innovation in computing technology has driven economic development and produced strong growth in computing jobs [1]. However, the increasing demand in computing job market has exacerbated the shortfall of workforce in computing, which in turn poses a big challenge for computing education. The divergence between the demand for computing professionals and productivity of computing education can be illustrated using cloud computing technology. A research from IDC predicts that the cloud will generate nearly 14 million new jobs worldwide by 2015 [2]. In the state of Georgia, Silver lining, a cloud computing company is in the process of adding 900 new jobs in Atlanta [3]. However, very few institutions of higher education provide courses on cloud technology. Taking the state of Georgia as an example, only a handful of all Georgia public institutions offer cloud computing related courses.

We believe that one of the major reasons for the difference between the demand for computing professionals and productivity of computing education is the lack of resources, including, but not limited to, faculty expertise, teaching materials, computing equipments, lab space, and best practice. In order to solve this dilemma, we propose this "Computing Education on Cloud (CEC)" project that aims to deliver critical computing education as cloud services, which can be accessed anywhere, anytime, by any registered user with minimum requirement of resources. We start the project by identifying the following categories of computing education that are critical but often missing from a typical computing curriculum due to resource constraints: 1) emerging computing technologies (such as cloud computing, big data computing, and mobile computing); 2) algorithmic thinking (problem solving skills that are essential to a computationally competent workforce and computationally empowered citizenry); and 3) computing security (critical to addressing a fundamental problem that a large population of computing technology creators lack deep understanding of computing security challenges and technologies). We further set the primary goal of the CEC project to be the producing of open courseware, which can be delivered as could service, for each of the three identified categories. Each courseware contains a group of units that cover different aspects of the corresponding category. A courseware unit is a collection of the following items: unit description, learning objectives, lecture notes, recorded lectures, simulation tools, lab manual, cloud lab environment, assignments, and assessment questions. All the items of a courseware unit will be hosted on our cloud platform and accessible at user's PCs, Laptops, or mobile devices anywhere and anytime. The second goal of the CEC project is to evaluate the effectiveness of the open courseware from different perspectives, including user population, courseware adoptions, learning objectives, student success, and reviews by peer institutions.

2 Computing Educational Perspective of Cloud Computing

Cloud Computing is a computing model that enables rapid on-demand provisioning of computational resources that are hosted remotely through the Internet as services. Typical types of computing resources that can be delivered as cloud service include computing infrastructures, computing platforms, and computing software. Correspondingly, the three typical cloud services are Infrastructure as a Service

(IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).

IaaS can be further divided into Server as a Service, Storage as a Service, and Network as a Service. Computing educators can leverage IaaS to set up required computing environment for various educational needs. For instance, Linux server with proper setting can be obtained from cloud as service in order to satisfy the needs of variety of computing courses, such as Operating Systems and System Programming. More complicatedly, a virtual Beowulf cluster can be set up from multiple virtual servers with the Network as a Service, which is a very cost effective way to build a lab environment that promotes parallel computing. In the next section, we will describe how IaaS is used to deliver necessary lab environment for computing security courseware in the CES project.

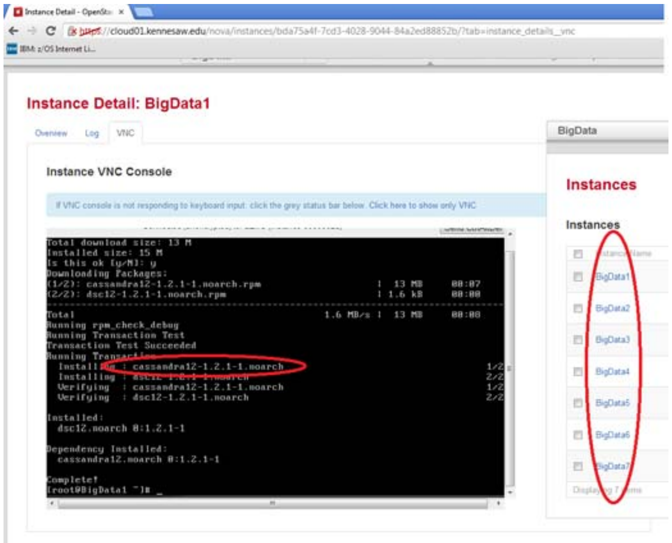
PaaS delivers highly scalable run-time platform as cloud services. Prominent examples of PaaS include Google App Engine [4] and Windows Azure [5]. Teaching the development of web applications or web services is an important component in a computing curriculum. Instead of setting up traditional LAMP environment as the run-time platform for the corresponding courses, computing educators can instruct students to deploy their web applications to Google App Engine if using Java technology or Windows Azure if using C#. By using PaaS in teaching, a computing curriculum is able to put more emphasis on building scalable software, which is an essential skill that any computing graduate should have nowadays. In the next section, we will focus on how the CES project leverages Big Data computing platforms as cloud services to deliver critical computing education on emerging technologies.

SaaS have been evolved as a popular model for delivering software, especially business applications, such as business intelligence, CRM, and HR applications. In the CES project, we will use SaaS as the model to deliver the produced open courseware. Especially, we will design a suite of software services to help students to develop algorithmic thinking skills, as described in the next section.

3 The CEC Project

The major activities of the CEC project are the production of a set of open courseware units that cover the critical computing education in the following identified categories: emerging computing technologies (such as cloud computing, big data computing, and mobile computing); algorithmic thinking (problem solving skills that are essential to a computationally competent workforce and computationally empowered citizenry); and computing security. Each courseware contains a group of units that cover different aspects of the corresponding category. A courseware unit is a collection of the following items: unit description, learning objectives, lecture notes, recorded lectures, simulation tools, lab manual, cloud lab environment, assignments, and assessment questions. All the items of a courseware unit will be hosted on our cloud platform and accessible at user's PCs, Laptops, or mobile devices anywhere and anytime.

Producing courseware as cloud service for each identified category has its own challenges to overcome. For instance, the courseware unit for teaching Big Data Computing in the category of emerging computing technology requires provisioning big data computing platform as cloud service, so that the end user without local access to high throughput server clusters can simply access big data computing platform hosted on our cloud from any browser or mobile application. In order to address this challenge, we set up an experimental cloud platform using KVM [6] and Rackspace [7]. This cloud platform, which is called cloud@cs.Kennesaw.edu, integrates multiple servers and workstations in the Cloud Lab at the Department of Computer Science. Furthermore, as a proof-of-concept, we created a big data computing platform and provisioned it as a cloud service. The big data cloud platform includes a Hadoop cluster and a Cassandra database ring. Figure 1 demonstrates that our big data computing platform can be simply accessed from an Internet browser. We will use the same approach to provide cloud services for platforms that are needed for cloud computing, mobile computing, and computing security labs.

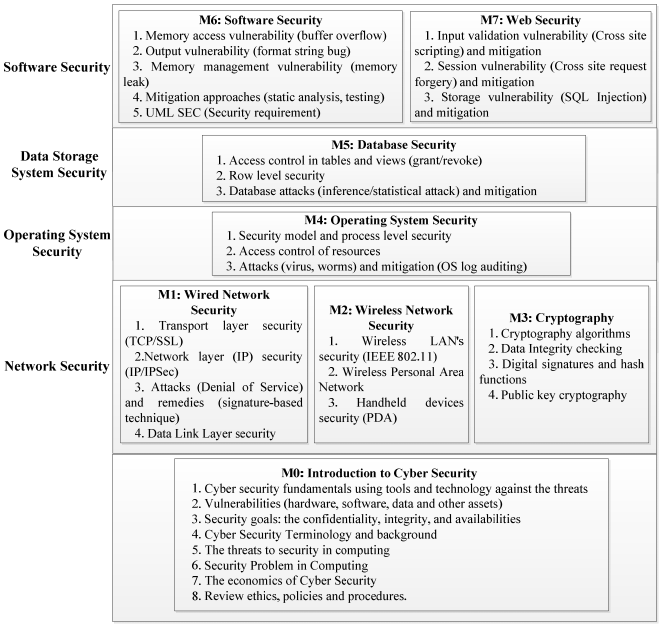


If VNC console is not responding to keyboard input: click the grey status bar below. Click here to show only VNC

```
Total download size: 13 M
Installed size: 15 M
Is this ok [y/N]: y
Downloading Packages:
(1/2): cassandra12-1.2.1-1.noarch.rpm                     |  13 MB     00:07
(2/2): dsc12-1.2.1-1.noarch.rpm                           | 1.6 kB     00:00
--------------------------------------------------------------------------------
Total                                            1.6 MB/s |  13 MB     00:08
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : cassandra12-1.2.1-1.noarch                                    1/2
  Installing : dsc12-1.2.1-1.noarch                                          2/2
  Verifying  : cassandra12-1.2.1-1.noarch                                    1/2
  Verifying  : dsc12-1.2.1-1.noarch                                          2/2

Installed:
  dsc12.noarch 0:1.2.1-1

Dependency Installed:
  cassandra12.noarch 0:1.2.1-1

Complete!
[root@BigData1 ~]# _
```

Figure 1: Access our Big Data Computing Platform at an Internet browser

Algorithmic thinking is a fundamental method for problem solving in this era of computing. It concerns how to take a vague problem and turn it into a set of interpretable steps. Cathy Davidson, a Ruth F. DeVarney Professor at Duke University, argues that our education needs to start emphasizing the 4th R (Reading, wRiting, aRithmetic, algoRithms) in kindergarten, even preschool [8]. We believe that equipping K-16 students with solid algorithm thinking skills is a necessary and effective way to broadening the participation in computing. However, due to the intrinsic nature of solving complex problems, it is common that students feel subjects involving algorithmic thinking are challenging to learn. In order to address this bottleneck, we plan to produce cloud services that provide step-by-step animation for a wide range of typical algorithms. Users can access our cloud services from Internet browsers or mobile applications to interact with an animated process, which can help them to understand every details of the algorithm in a visible and intuitive way.



M6: Software Security
1. Memory access vulnerability (buffer overflow)
2. Output vulnerability (format string bug)
3. Memory management vulnerability (memory leak)
4. Mitigation approaches (static analysis, testing)
5. UML SEC (Security requirement)

M7: Web Security
1. Input validation vulnerability (Cross site scripting) and mitigation
2. Session vulnerability (Cross site request forgery) and mitigation
3. Storage vulnerability (SQL Injection) and mitigation

M5: Database Security
1. Access control in tables and views (grant/revoke)
2. Row level security
3. Database attacks (inference/statistical attack) and mitigation

M4: Operating System Security
1. Security model and process level security
2. Access control of resources
3. Attacks (virus, worms) and mitigation (OS log auditing)

M1: Wired Network Security
1. Transport layer security (TCP/SSL)
2.Network layer (IP) security (IP/IPSec)
3. Attacks (Denial of Service) and remedies (signature-based technique)
4. Data Link Layer security

M2: Wireless Network Security
1. Wireless LAN's security (IEEE 802.11)
2. Wireless Personal Area Network
3. Handheld devices security (PDA)

M3: Cryptography
1. Cryptography algorithms
2. Data Integrity checking
3. Digital signatures and hash functions
4. Public key cryptography

M0: Introduction to Cyber Security
1. Cyber security fundamentals using tools and technology against the threats
2. Vulnerabilities (hardware, software, data and other assets)
3. Security goals: the confidentiality, integrity, and availabilities
4. Cyber Security Terminology and background
5. The threats to security in computing
6. Security Problem in Computing
7. The economics of Cyber Security
8. Review ethics, policies and procedures.

Figure 2. Eight Knowledge Area of Cybersecurity

Although the need for equipping computer science majors with comprehensive knowledge in cybersecurity has been long recognized in literature [9, 10, 11], it is far from a norm that an institution of higher education expands its CS curriculum in cybersecurity. The major challenge of preparing computing majors with comprehensive knowledge in computing security lies in the fact that security issues are widely spread in different aspects of computing, such as networking, databases, operating systems, web, software applications, and cryptography. Therefore, teaching computing security is not a matter of adding one or two new courses. Our approach to address this issue is that we first form a comprehensive view of cyber security by identifying eight knowledge areas (M0-M7) in a layered model of cyber security as shown in Figure 2. Each knowledge area contains a list of security topics, each of which consists of corresponding security issues and solutions. Then, for each cyber security topic identified in figure 3, we design and implement a group of courseware units as cloud services. For example, in the knowledge area M2-Wireless Network Security, the following three courseware units will be produced: M2.1-Wireless LAN's security, M2.2-Wireless Personal Area Network security, and M2.3-Handheld device Security; in the knowledge area M7-Web Security, the following three courseware units will be produced: M7.1-Input validation vulnerability and mitigation, M7.2-session vulnerability and mitigation, and M7.3-storage vulnerability and mitigation. Finally, we view a typical CS curriculum as a "motherboard" and plug in all cybersecurity course units in the "motherboard" to form an expanded version of CS curriculum with added value on cyber security as shown in figure 3. In this way, an institute of higher education can easily plug these units in their existing computing curriculum without requirements on extra resources.

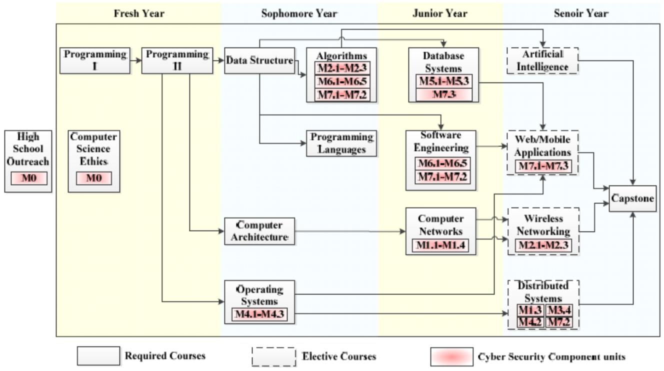


Figure 3. Illustration on how computing security courseware units fit a typical computing

Most of the cybersecurity courseware units require a server environment with proper software installations as the lab settings. We will again leverage the cloud computing technology to deliver the required lab environments as services. More specifically, for each relevant cybersecurity courseware unit, we will configure a server with proper hardware setting and necessary software installation and save the server image. The users of cybersecurity courseware will then be able to provision as many identical servers as needed from the image on the cloud.

4 Final Remarks

Cloud computing is reshaping the information technology landscape. As computing researchers and educators, we need to not only consider how to incorporate cloud computing technology itself in the computing curriculum, but also rethink how critical education could be delivered to diverse populations in a more effective way by leveraging the promises the new technology brings. The described Computing Education on Cloud (CEC) project is such an effort that aims to deliver critical computing education as cloud services, which can be accessed anywhere, anytime, by any registered user with minimum requirement of resources.

We identify the following three subject areas are critical but often missing from a typical computing curriculum due to resource constraints: emerging computing technologies, algorithmic thinking, and computing security. The major activities of the CEC project are then the production of open courseware covering these areas that can be delivered as cloud services. In this paper, we describe the solutions that address some challenging issues in delivering the proposed courseware via cloud.

As an important component of the CEC project, the evaluation of the produced courseware services will be conducted from multiple angels, including user population, courseware adoptions, learning objectives, student success, and reviews by peer institutions. Feedback provided by multi-factor evaluations will helps us to continuously refine our courseware services. Therefore, this paper is also serving as a call for collaboration and participation in the CEC project.

Acknowledgement: The authors of this paper would like to thank Sean Pryor and Joe Brown for their technique support in construction of the IaaS portion of the experimental cloud platform. The authors would also thank Dr. Hossain Shahriar, Dr. Jing He, and Dr. Sarah North for their valuable discussion on open courseware for cybersecurity.

5 References

[1] Computing Education for the 21st Century (CE21), NSF. http://www.nsf.gov/pubs/2012/nsf12527/nsf12527.htm

[2] Cloud Computing to create 14 million new jobs by 2015, IDC study. http://www.microsoft.com/en-us/news/features/2012/mar12/03-05CloudComputingJobs.aspx, March 05, 2012.

[3] U. Karkari. Silver Lining: Cloud computing company to add 900 new jobs, Atlanta Business Chronicle, Feb. 1, 2013.

[4] Google App Engine. https://developers.google.com/appengine/

[5] Windows Azure. http://www.microsoft.com/en-us/server-cloud/windows-azure.aspx

[6] KVM. http://www.linux-kvm.org

[7] Rackspace Cloud. http://www.rackspace.com/Cloud

[8] C. Davidson. “Why we need a 4th R: Reading, wRiting, aRithmetic, algorithms”, DMLcentral, January 25, 2012. http://dmlcentral.net/blog/cathy-davidson/why-we-need-4th-r-reading-writing-arithmetic-algorithms

[9] A. Yasinsac. “Information Security Curricula in Computer Science Departments: Theory and Practice”, the George Washington University Journal of Information Security, Vol. 1, No. 2, 2002.

[10] M. Locasto and S. Sinclair. “An Experience Report on Undergraduate Cyber-Security Education and Outreach”, Proceedings of the 2nd Annual Conference on Education in Information Security (ACEIS 2009), Ames, IA, USA, February 2009.

USING A MULTIDISICPLINARY RESEARCH PROJECT TO STRENGTHEN LEARNING IN SOFTWARE ENGINEERING

Wendy A. Lawrence-Fowler, Laura M. Grabowski,
Richard H. Fowler

Department of Computer Science
The University of Texas-Pan American
Edinburg, Texas

Abstract— It is our premise that a single experience, even in an in-depth project based course, is not adequate for students to transfer all required software engineering knowledge and skills to other situations. The project we described allows us to instill an engineering attitude in the curriculum and the academic computing environment. Through flexibility and responsiveness to change, this approach prepares students to adapt to rapid change in the field by teaching enduring principles in the context of current practice. Exposing computer science students to the convergence of multiple disciplines strengthens a student's ability to manipulate and automate different levels of abstraction simultaneously and develop new conceptual frameworks to address current and future challenges in hardware and software engineering.

Keywords—experiential learning; software engineering; visualization; Avida; project based learning

I. INTRODUCTION

The core body of knowledge, skills and abilities (KSAs) that a software engineer must know are defined in projects like Software Engineering Body of Knowledge (SWEBOK) [1] and SE2004 Software Engineering Education Knowledge (SEEK) [2]. Although SE2004 presents curriculum guidelines for the design and adaptation of educational programs for software engineering, there are still challenges for educators. As with most other disciplines, software engineering education cultivates a wide range of capabilities that extend from current engineering and professional practice to those skills and knowledge sets that will result in the next generation of software engineers with the abilities to lead and solve challenges well into the 21st century. This means that educational opportunities must encompass project management, professional ethics, relevant legal knowledge, international exchange, organizational and management skills, discipline and teamwork, initiative and creativity, as well as foster autodidactism. All of this must be done in a manner that allows students to apply enduring formalisms within current practice.

To capture the context of current practice in industry, a number of courses have incorporated projects that emulate real-world experiences, sometimes involving real clients [3-13]. The introduction of formal software engineering knowledge together with concurrent technique and skill development, can bring valuable experience to both students and faculty [5]. While it is critical that software engineering students leave the university with the technical and soft skills expected by employers [14], a single semester or two of course work may not prepare them to move into the workplace as a software engineer.

Our solution to this challenge has been to design a capstone experience that brings students into an ongoing multi-disciplinary research project as software engineers. The role the students play depends on the state of the project, their previous coursework, and their individual strengths. We believe that this experience reinforces and extends the knowledge and skills that are learned through the software engineering curriculum. By placing them in an organizational structure with the strategic objective to develop a library of software tools for the visualization of data generated by the Avida platform, students learn about individual accountability, learn about measureable deliverables, and learn to create transparency through communication. We contend that working within a multidisciplinary team, particularly in a domain outside of the students' expertise, not only fosters autodidactic behavior, but also expands the students' ability to think abstractly.

II. PROJECT DOMAIN

A. Bioinformatics and Phylogeniies

The amount of data generated in bioinformatics continues to grow exponentially, data format changes often and there is a continual increase in the complexity of results. This means that researchers need tools that allow them to effectively link, match, cleanse, transform, analyze and understand the huge volume of data (Big Data). Tools that are currently available to work with Big Data are often complex and hard to use or require resources beyond those readily available [15]. In the area of phylogenetics, this has lead to a proliferation of idiosyncratic, experiment based software tools, data formats, analytical techniques and web services. In turn, this creates a challenge when integrating phylogenetic and other related biological data and a need for reusable software that can read, manipulate and transform this data into the various forms required to build computational pipelines[16].

B. Visualization in Bioinformatics

The goal of visualization is to provide insights from masses of data through visual representations [17]. Visualizations are powerful and often elegant abstractions that provide mechanisms for the discovery of potentially valuable relationships in data. Techniques for visualizing commonalities between hierarchically structured data sets have a long history in visualization, and new graph visualization techniques have been introduced to display multi-category graphs, i.e., graphs that show both connectivity and category [18,19,20,21].

The use of visualization in bioinformatics is not new, and a number of successful systems have been developed. However, challenges remain in developing readily available and easy to use visualization tools that can be used with large heterogeneous datasets. Many of the current techniques for bioinformatics visualization of phylogenetic data employ node-link graphs to represent data as trees. Although some software packages can handle the massive amount of data generated in bioinformatics work, reconstructions of phylogenies have typically been limited, primarily by running time, to sets of some dozens or hundreds of organisms [22].

Even though there are numerous software systems already available, active research drives unforeseen change in requirements which creates a continuing need for tools and modular software kits that can be integrated into bioinformatics analysis workflows independent of data storage formats [16], volume, and complexity. The immediacy of the need for results often leads to the development of analysis and visualization programs by scientists whose principle education is in fields other than Computer Science or Software Engineering [22,23]. The increasing body of nontrivial software being developed by end users can be a very costly practice [22] and often a distractor from the original research initiative.

C. *The Avida Platform*

Avida is a software platform for digital evolution, used for studying evolutionary processes relevant to both natural and artificial systems [24]. Avida is free, opensource software written in C++, and can be easily extended to add new features and system capabilities. Avida is the most widely used digital evolution software in the world, and its users encompass a wide range of backgrounds and expertise, including biology, computer science, engineering, and philosophy. Avida places a population of self-replicating computer programs, called *digital organisms* or *Avidians*, in a user-defined computational environment. Avida is not a simulation of natural evolution, but a separate instance of evolution in its own right: the digital organisms compete, replicate, and mutate, thus satisfying the fundamental requirements of evolution [25].

The Avida platform is highly extensible. This and the flexibility of system configuration means that Avida experiments produce vast amounts of data that need to be processed for analysis. The platform contains some limited analysis tools, but in general, individual researchers must write most of their own analysis tools. Computer scientists and engineers who use Avida do not generally find this particularly onerous, except that writing such custom tools is time-consuming. However this situation is a problem for researchers who are not trained or experienced programmers. Learning the tools and programming skills required to write software for basic analysis is time consuming, and presents a significant barrier for non-programmer users of Avida.

III. Existing Software Engineering Curriculum

The University of Texas-Pan American (UTPA) offers ABET accredited degrees in Computer Science and in Computer Engineering. In addition to providing a core body of knowledge in the domain, the educational objectives for both degree programs include providing graduates with an understanding of social, professional and ethical considerations related to their respective disciplines

There is a required course in Software Engineering in each degree. This course may be taken after students have completed courses in advanced algorithms and data structures and in systems programming. Students in the Computer Engineering software track are required to take a second course in Software Engineering. A third course in software verification, validation and quality assurance is available as an elective. Each degree requires a capstone senior project. The typical pathway through the either degree results in very few students graduating with two or more semesters of Software Engineering.

Software Engineering 1 provides a formal approach to the state-of-the-art techniques in software design and development with focused discussion on project planning, requirements, specification, system design, testing and implementation, integrating the knowledge that the students have learned in their other classes. Team projects are used to provide some experience in real world problem solving. The nature of the projects requires that students learn to not only apply software engineering principles, but, more importantly, develop collaborative and project management skills and expand their written and oral communication proficiency. The teams work on identified problems or challenges, most often involving concepts outside of CS and CE students' domains of expertise. This forces students to expand their existing knowledge base beyond the principles of their discipline and software engineering. Addressing a real problem adds value and motivation to the learning experience: the project becomes a challenge with multiple options for a solution, not just an exercise. Students take ownership, and the process to a solution results in identifiable learning outcomes [3,4,13].

The end products of Software Engineering 1 class projects are used as a starting point for the Software Engineering II course, which focuses on the analysis of requirements and software architecture with an emphasis on object design, and adds implementation, testing and validation, maintenance and software re-engineering. Again, students work on the project(s) in teams. A third course, Software Verification, Validation and Quality Assurance, covers methods for evaluating software for correctness and reliability, including code inspections and their role in software verification, program proofs and testing methodologies, formal and informal proofs of correctness, unit and system testing techniques, testing tools and limitations of testing, statistical testing, and reliability models.

The curriculum includes a capstone senior project in which students construct a software product, following it through the stages from initial specification to the final completed project. This paper discusses the use of the capstone course for experiential learning that is designed to deepen students' comprehension of the application of the fundamental principles of software engineering while strengthening their skill in communication and collaboration through the incorporation of students into an ongoing interdisciplinary software development project focused on a research initiative [14].

The capstone course we discuss supports learning in the context of an ongoing software development project, Avida Toolkit. The project involves the visualization of phylogenies generated from Avida experiment data. Avida provides a complex computational environment in which the evolution of digital organisms are tracked and analyzed to help find answers to multiple research questions. The transparency of the Avida platform means that any experiment may result in a very large data set. A set of tools exists to work with Avida data, but the tool library is continually under revision as new research questions arise. The approach taken in the capstone course is to use a software development process model to develop a tool library that can serve as the basis for students' innovation, differentiation and growth in understanding while supplying tools for answering questions that researchers ask, so that knowledge that can be gained in phylogenies and, more broadly, bioinformatics.

IV. The Avida Toolkit Project

The Avida Toolkit Project originally started as a way to provide data analysis and visualization tools for researchers working on the Avida platform, thus allowing them to focus their attention on the science, and so discover complex relationships in the data that would provide new insights in their domain, rather than programming and tool development. Like many software projects, new technologies, applications, and participants resulted in project evolution. Changes in technology and environmental circumstances allowed the addition of a second objective for the project. The project would provide an experiential learning opportunity in which students apply their knowledge of software engineering, computer science, and computer engineering to an ongoing software engineering project addressing data analysis needs of researchers using the Avida platform.

The Avida Toolkit project originated two and a half years ago with one faculty member, whose research involves analysis of Avida data. She began working with a undergraduate student to determine the feasibility of working with the Avida data files in the existing environment using resources available at UTPA. Given acceptable results, she assigned the Avida toolkit as the project for a graduate level course in software engineering, providing both a real-world experience for graduate students and research support of Avida platform users. The results was three distinct project plans, requirements, requirements analysis, design, and design analysis documents.

Each set of documents contained valuable insights into the development of a toolkit, however, as might be expected, there were competing plans for the project implementation. However, it was possible to merge the documents to create a single requirements document and a set of design documents that aligned with the requirements. There were also three software prototypes of tools.

Subsequently, the project expanded to include additional faculty with knowledge and expertise in biology, evolution, artificial intelligence, information management, and visualization. As a management team formed and collaboration solidified, the project requirements began to expand. Just as we see in our project based classes in software engineering, each participant brought his or her own perspective and understanding of the problem, here, including visualizing phylogenies, visualizing big data sets, analyzing big data sets, evolution, and biology. It swiftly became apparent that to be successful in creating an Avida toolkit, we would have to be deliberate in our actions or competing strategies and priorities would quickly mire our execution in quicksand. So, we decided to practice what we teach.

Gaining insight from the experience of Schaetter, Koeglmayr, Blankenbach, and Nippa [12], we use a slightly modified incremental development process based on Boehm's spiral model [26]. Using the model, we combine the advantages of top-down and bottom-up concepts, integrating prototypes in the design process. Incremental refinement as we move through the spiral allows us to better manage the risk and keep the software development process under control. Because this is an ongoing research project, students in CS and CE can join the project as their capstone course. Students' specific goals, tasks and products for their capstone course are driven by the current state of the software development process for the Avida Toolkit project.

The Spiral Model is an ideal model for this project. We have the flexibility to incorporate a variety of requirements, design techniques, and implementation styles, e.g. on one cycle we may use Agile programming or XP and on another cycle we may use a version of commercial off the shelf (COTS) products to incorporate new or repurposed source code. Each iteration of the cycle allows us to refine or extend the Avida Toolkit.

This approach allows us to be aware of and protect the project against chasms between plans and actions and between actions and outcomes. Although actual implementation may be punctuated at any point in time by changes in availability faculty and student participation time, actual execution is bolstered by implementing corporate strategies for accountability: progress against the plan is measured, individual assignments are made with clear and concise deliverables and firm due dates; results and actions are communicated transparently.

This approach means students and faculty work together closely. Students experience different roles and learn to communicate within a diverse project team, as well as with specific stakeholders. Teamwork and efficient and transparent group communication establishes a basis for a learning community that includes both students and faculty. For example, two seniors in computer science recently joined the team. Their work involved refining requirements for Avida data management to facilitate the use of open source software libraries to derive and display trees. Each student was given a specific assignment. While one student focused on a comparison of existing software libraries used in visualization of large and pervasive document sets, the other student focused on Biopython and the development of a prototype tool to display and manipulate phylogenetic trees in 2 and 3-D. The separate assignments provided for individual accountability and transparent communication made sure that the team was aware of the overall progress of the project.

Based on the student generated deliverables, requirements definition will begin for a new preprocessing module allowing syntactic analysis of Avida data. Using the existing code graph layout we will develop prototypes that generate neontological time-scaled views, phylogenetic depth views, and pseudo-paleontological views of the Avida data. We will maintain the current ability to move through, refocus, zoom in, zoom out and carve out portions of the graphs. Other revisions under review include enhanced manipulation of actual node data. We will take advantage of our previous research in document set visualization and software developed at UTPA, InfoViz [27, 28,29]. The current functionality in InfoViz uses 2 and 3-D visualization tools to identify groups of useful documents and gain access to relevant clusters and individual documents. Given the obvious analogy to the size and complexity of data generated during Avida experiments, we are evaluating the potential redesign and repurposing of the InfoVIz software and expand our perspective on the visualization of phylogenies.

This project is a real-world software development environment in an academic environment. New team members are on-boarded and off-boarded as if they were working in industry. They have to familiarize themselves with the Avida platform, statistical analysis and visualization concepts used in the project, as well as the terminology and basic ideas involved in deriving phylogenies and they must understand the current state of the Avida Toolkit project.

V. DISCUSSION

An interdisciplinary approach and a learning method that uses a culture of accountability and the state of the project to drive the definition of the individual learning experience is not common in Software Engineering education. Brugge and Gluhow [3] discuss the importance of finding an innovative way to facilitate the knowledge transfer process, a way to ensure that knowledge outlasts the end of the semester. They suggest that enhancing written documentation is not enough, and that involvement beyond the project course is important.

Given our own experience and that of others, as well as a project based course in software engineering that emphasizes learning and application of principles of software engineering together with a senior capstone project where students demonstrate their ability to complete a software project, we were able to create a framework where students extend their skills in software engineering, expand their understanding of complex computational environments and exhibit actual transfer of knowledge in computing sciences and software engineering to the solution of problems in evolutionary biology. Using a real world, faculty-driven research project, students work in a multidisciplinary team including faculty with specializations in visualization, information and knowledge management, artificial intelligence, and evolutionary biology. The project uses visualization as a primary element to bind the concepts of the disciplines through the management and display of the datasets of biology in the computing environments employed in the project. To promote deeper understanding and experience in knowledge transfer, we have designed a project plan that includes individual accountability as well as deliberate, effortful abstraction together with time to search for connections [12,30].

This approach gives students more control over their learning and more importantly, students acquire new knowledge only as a necessary step in solving authentic and cross-disciplinary problems representative of professional practice. Students link prior knowledge with new knowledge and gain a more balanced understanding of complex situations. Furthermore, affective learning serves as a critical link between cognitive and behavioral learning that motivates students and enhances educational outcomes, learning that results in behavioral change and not just conceptual mastery [12].

The purpose of creating a real-world software development project using the spiral model is to improve the students' learning experience, while reinforcing the soft skills required in software engineering and computer science courses. It provides students with the experience of understanding and solving real-world challenges and the opportunity to understand the software development process. Students achieve defined learning outcomes and soft skills that will be important to them in their software engineering careers. In addition, students gain experience in a real, on-going research project.

VI. CONCLUSIONS

It is our premise that a single experience, even in an in-depth project based course, is not adequate for students to transfer all required software engineering knowledge and skills to other situations. The project we described allows us to instill an engineering attitude in the curriculum and the academic computing environment. Through flexibility and responsiveness to change, this approach prepares students to adapt to rapid change in the field by teaching enduring principles in the context of current practice. Exposing computer science students to the convergence of multiple disciplines strengthens a student's ability to manipulate and automate different levels of abstraction simultaneously and develop new conceptual frameworks to address current and future challenges in hardware and software engineering.

Finally, we suggest that this hones soft skills and reflects the reality of large development projects as seen in the workplace; team members come and go throughout the life of the project. Fortunately, this project has the advantage of a stable core of faculty whose research interests continue to drive the project forward. Future plans include using biologically inspired techniques, such as flock based clustering, to computationally based automation and computationally inspired techniques from 2 and 3-D visualization that have been shown successful in other domains, to biological problems. Using an analogy of the corporate world and their approach to visualization and data analytics may also provide additional insight to manage, understand, and take advantage of the bioinformatics data's full value.

REFERENCES

[1] Bagert, Donald J., Dupuis, Robert, Freeman, Peter A., Saiedian, Hossein, Shaw, Mary M., and Thompson, J. Barrie. "Panel: Software Engineering Body of Knowledge (SWEBOK)." *Proceedings of the 23rd International Conference on Software Engineering (ICSE)*, 2001.

[2] "Software Engineering 2004 Curriculum Guidelines for Undergraduate Programs in Software Engineering: A Volume of the Computing Curricula Series." 2004. http://sites.computer.org/ccse/. Accessed 7 January, 2013.

[3] Brugge, B. and Gluchow, M. "Towards Production Ready Software in Project Courses with Real Clients." *First International Conference on Software Engineering Education based on Real-World Experiences (EduRex)*, 2012, pp. 5-9.

[4] Grega, W., Kornecki, A., Sveda, M., and Thiriet, J. M. "Developing Interdisciplinary and Multinational Software Engineering Curriculum." *Proceedings of the ICEE'07*, 2007, pp. 3-7.

[5] Sebern, Mark J. "The Software Development Laboratory: Incorporating Industrial Practice in an Academic Environment." *Proceedings of the 15th Conference on Software Engineering and Training*, 2002, pp. 118–127.

[6] Suri, Deepti and Durant, Eric. "Teaching Requirements through Interdisciplinary Projects." *Proceedings of the 2004 ASEE North Midwest Regional Conference*, 2004.

[7] Jazayeri, Mehdi. "The Education of a Software Engineer." *Proceedings of the 19th IEEE International Conference on Automated Software Engineering*, 2004, pp. 18–27.

[8] Herold, Michael, Ganci, Aaron, Ribeiro, Bruno. Ramnath, Rajiv and Stone, R. Brian. "Work in Progress - Computer Science Perspectives on Integration with Human-Centered Design." *ASEE/IEEE Frontiers in Education Conference (FIE)*, 2011, F4F.

[9] Dekhane, S. and Tsoi, Mai Yin. "Work In Progress - Inter-Disciplinary Collaboration For A Meaningful Experience in a Software Development Course." *ASEE/IEEE Frontiers in Education Conference (FIE)*, 2010, S1D-2.

[10] Jaccheri, L. and Sindre, G. "Software Engineering Students Meet Interdisciplinary Project Work and Art." *Proceedings of 11th International Conference on Information Visualization*, 2007, pp. 925-914.

[11] Albu, A. B., Malakuti, K., Tuokko, H., Lindstrom-Forneri, W., and Kowalski, K. "Interdisciplinary Project-Based Learning in Ergonomics for Software Engineers: A Case Study." *The Third International Conference on Software Engineering Advances*, 2008, pp.295-300.

[12] Schaetter, Alfred, Koeglmayr, Hans-Georg, Blankenbach, Karlheinz, and Nippa, Markus. 2009. "Interdisciplinary Approach to Software Engineering Education." *The Journal of Systematics, Cybernetics and Informatics*, vol. 7, no. 5, pp. 29-36.

[13] Richardson, I., Reid, L., Seidman, S. B., Pattinson, B., and Delaney, Y. "Educating Software Engineers of the Future: Software Quality Research Through Problem-Based Learning." *IEEE-CS Conference on Software Engineering Education and Training (CSEE&T)*, 2011 pp. 91-100.

[14] Bareiss, Ray and Katz, Edward. "An Exploration of Knowledge and Skills Transfer from a Formal Software Engineering Curriculum to a Capstone Practicum Project." *Proceeding of Conference on Software Engineering Education and Training (CSEE&T)*, 2011, pp.71-80.

[15] "Big Data, Bigger Opportunities: SAS Special Report on High Performance Analytics." 2012. http://www.sas.com/reg/gen/corp/hpa-insights. Accessed 8 January 2013.

[16] Talevich, Eric, Invergo, Brandon, Cook, Peter, and Chapman, Brad. "Bio.Phylo: A Unified Toolkit for Processing, Analyzing and Visualizing Phylogenetic Trees in Biopython." *BMC Bioinformatics*, 2012, pp. 219-227.

[17] Tufte, Edward R. 1983. *The Visual display of Quantitative Information*. Graphics Press.

[18] Wang, Yue, Teoh, Soon Tee, and Ma, Kwan-Liu. "Evaluating the Effectiveness of Tree Visualization Systems for Knowledge Discovery." *Proceedings of Eurographics Visualiztion Symposium*, 2006, pp.67-74.

[19] Zainon, W. M. N., Talib, A. Z., and Belaton, B. "Display-Based Approaches for Phylogenetic Tree Visualisation." *International Conference on Distributed Framework and Applications (DFmA)*, 2010, pp. 1-7.

[20] Jones, Chad, Armstrong, Ryan, and Ma, Kwan-Liu. "Visualizing the Commonalities Between Hierarchically Structured Data Queries." *Proceedings of the International Workshop on Visual Languages and Computing*, 2010, pp. 251-256.

[21] Hale, P., Solomonides, T., and Beeson, I. "Requirements and Software Engineering for Tree-Based Visualisation and Modelling: A User Driven Approach." *Postgraduate Conference for Computing: Applications and Theory (PCCAT 2011)*, 2011.

[22] Moret, B., Wang, M E, Li-San, and Warnow, T. 2002. "Toward New Software for Computational Phylogenetics." *Computer*, vol.35, no.7, pp. 55-64.

[23] Letondal, Catherine and Zdun, Uwe. "Anticipating Scientific Software Evolution as a Combined Technological and Design Approach." *Second International Workshop on Unanticipated Software Evolution*, 2003.

[24] Ofria, C. and Wilke, C. O. 2004. "Avida: A Software Platform for Research in Computational Evolutionary Biology." *Artificial Life*, vol. 10, pp. 191-229.

[25] Dennett, D. "The New Replicators." In M. Pagel (ed.). 2002. *Encyclopedia of Evolution*. New York, NY: Oxford University Press, pp. E83-E92.

[26] Boehm B. 1986. "A Spiral Model of Software Development and Enhancement." *ACM SIGSOFT Software Engineering Notes*, vol. 11, no. 4, pp. 14-24.

[27] Fowler, R. H., Fowler, W. A. L., and Wilson, B. A. "Integrating Query, Thesaurus, and Documents through a Common Visual Representation." *Proceedings of the 14th Annual International ACM SIGIR Conference*, 1991, pp. 142-151.

[28] Fowler, R. H. and Fowler, W. A. L. "Information Visualization and Retrieval Using Stereoscopic Display of Document and Term Networks." *Stereoscopic Displays and Applications VIII. Society of Photo-Optical Instrumentation Engineers*, 1998, pp. 148-155.

[29] Fowler, R. H. "Document Visualization Overview and Detail by Selectively Varying Density." *Proceedings of the International Conference on Modeling, Simulation, and Visualization Methods*, 2010, pp. 34-40.

[30] Perkins D. N. and Salomon, G. "Transfer of Learning", 1992. In *International Encyclopedia of Education (2nd ed.)*. Oxford, UK: Pergamom Press.

Evaluating a tool to Support Programming Learning

Carlos J. Costa[1], **and Manuela Aparicio**[2,3]
[1]School of Technology and Architecture, Insituto Universitario de Lisboa (ISCTE-IUL), Lisboa, Portugal
[2] ISEGI, Universidade Nova de Lisboa, Lisboa, Portugal
[3] IADE, UNIDCOM, Lisboa, Portugal

Abstract - *The problem of teaching and learning computer programming is far from being solved. In this context, the purpose of this paper is identifying the success factors related to systems used to support the learning process of computer programming. In this paper, it is presented a model supported in the literature review (especially Delone & McLean success model). The empirical work consists of evaluating the model. .*

Keywords: Computer programming learning, robotics, virtual robots, learning solutions

1 Introduction

Learning is a process of acquiring knowledge. In the area of computer programming, also known as computer coding, this learning process is not an easy task. In order to improve learning, Richard E. Pattis [26] developed an alternative method for introducing students to computer programming.

By initially limiting a student's language repertoire to imperative commands whose actions can be visually displayed, the Karel [18] approach quickly introduces students to concepts like procedures and key control structures.

Over time, several environments have been developed to introduce younger students to the concepts of programming through robotics (whether real or virtual). These include Papert's Turtle Graphics, Pattis' Karel [25][26] and Lego Mindstorms.

The use of robots is inherently attractive to many young students (even more than 2D graphics). However, the constraints imposed here by the real (or virtual) world have to do with the available kits that give these environments an essentially compositional focus, and restrict the opportunities for exploration and extension by students.

There have, of course, been many attempts over time to develop initial learning environments specifically for beginning programmers that focus on supporting learners in the general task of programming, either through direct manipulation of objects or through data and structure visualization.

In this context, we identify main problems and propose a model to evaluate the success of system usage. We also intent to identify what are the main dimensions related to these systems.

2 Programming Learning Problem

Educators face problems when teaching students programming [11] because programming is dynamic and abstract. Several authors have identified these difficulties in learning programming [22] [2] [36] [37] [21]. When faced with questionnaires, students perceived fewer difficulties than deduced by their teachers. Programming students often fail to recognize their own deficiency, due to their lack of general vision of the problems. In fact, students tend to focus on specific aspects rather than general ones. These facts are verified by several studies [21] [22]. To start, student and teacher perceptions on programming's knowledge base are rather different as teachers understand better the students' limitations than themselves. It is a fact that learning programming contains several activities. For example, learning a language's features, variables, program design and program comprehension. It is quite common that students start studying programming by features and variables rather than modeling. A study conducted by Rist [31] demonstrates that syntax understanding is not the main difficulty. Students may know the syntax and semantics of individual statements, but they do not know how to combine those elements in order to produce valid programs [42]. The cognitive domain plays an crucial role in the process of learning programming [30]: data recall (forget to declare initialize variables), understanding of the meaning of the problem, difficulty in translating that into a logic program, difficulty in algorithm design and applying a concept in a new situation, analysis of the program structure, and synthesis difficulty in integrating different modules. Furthermore, there are a myriad other reasons programming is difficult to learn [8] like originality, many skills, program design and comprehension, choice of language, timing and course structure, and varied individual student abilities. Table 1 summarizes learning difficulties discovered by different studies.

Table 1: Programming Learning Difficulties

	Less Experience	Novelty	Many Skills	Program Design	Program Comprehension	Choice of Language Program	Timing Course Structure	Different Abilities	Attitude towards learning	Cognitive factors
Jenkins, 2002 [15]		√	√	√	√	√			√	√
Robins *et al*, 2003 [33][34]	√		√	√	√		√	√	√	√
Lahtinen *et al*, 2005 [21]			√	√	√		√	√		√
Ala-Mutka *et al*, 2005 [2]			√	√	√	√		√		√
Sajaniem & Hu, 2006 [36][37]			√	√	√		√	√		√
Radosevic *et al*, 2009 [29]			√	√	√		√	√	√	√
Renumol *et al*, 2009 [30]			√	√	√		√	√	√	√
Cummmings, 2010 [8]		√	√	√	√	√	√	√		
Kordaki 2010 [20]	√		√	√	√	√	√	√		

According to the authors cited in Table 1, most difficulties were due to a lack of skills, and difficulties in integrating knowledge (program design and program comprehension). Timing and course structure also played a decisive role that prevented students in acquiring adequate knowledge for programming. A conclusion of some of the studies was this: if students start to learn program design first, then language structure, student comprehension was improved. A student's general attitude towards learning programming was also underlined in these studies: students often found programming extremely difficult, and many withdrew from courses soon after starting.

3 Information System Success

In this study, we used the DeLone and McLean Model of Information Systems Success [10]. This model measures the effectiveness level that an information system, the satisfaction level and the service level of a systems usage.

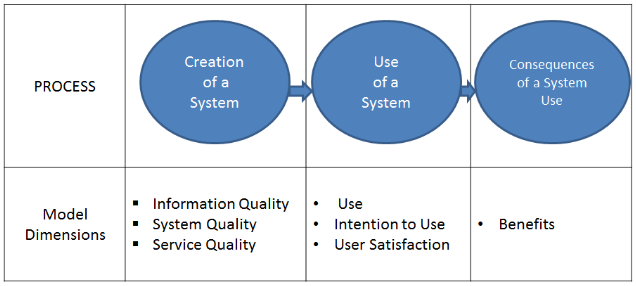
One of the main theories in which this model is based is Theory of Planed Behavior (TPB) [1]. TPB theory modules "how attitudes and behavior influence intention". Usually, TPB model explains the basis of human behavior, in which individuals lead their actual behavior by perceived behavioral normative conducts and individual intentions.

On other words, Delone and McLean Model [9] can be applied since the systems´ creation, from the system usage or usage intention to understand well the consequences' of a system use. The approach was used to understand the usage of a virtual environment in an effective way to teach programming learning.

The process of the DeLone and McLean Model has three phases:

1. Creation of a System
2. Use of a System
3. Consequences of a System

Table 2: Information Systems Success (Based on the DeLone & Mc Lean Process



PROCESS	Creation of a System	Use of a System	Consequences of a System Use
Model Dimensions	▪ Information Quality ▪ System Quality ▪ Service Quality	• Use • Intention to Use • User Satisfaction	• Benefits

Delone & Mclean [9] presented an Information Systems Success Model that consists of six interconnected dimensions of information system success. It is a framework and model for measuring the complex-dependent variable in Information Systems research. They organized information system success measures into six dimensions: System quality, Information, quality, use, user satisfaction, service quality, individual impact and organizational impact.

System quality measures the information system processing and the technical success. System quality is measured by functionality, ease-of-use, reliability, flexibility, data quality, portability, integration and importance.

Information quality is measured by completeness, timeliness, accuracy, relevance and consistency of information system output.

Use measures the usage of the output of an information system.

User satisfaction, that measures the response to the use of the output of an information system individual impact and organizational impact analysis the effect of information on the behavior of the user and the effect of information on organizational performance.

Service Quality with this dimension we can evaluate the service provided by the system, organization, department or an outsourcing company. Service quality can be measured by tangible, reliability, responsiveness, assurance and empathy. As in any other service, the quality of it is related with the support given to the client. The relationship between the two actors and the way that they play their roles is very important to achieve success.

According to presented an Information Systems Success Model that consists of six interconnected dimensions of information. A system can be evaluated in terms of information quality and system quality. These characteristics affect the subsequent use and user satisfaction. As a result of using the system, certain individual benefits will be achieved.

The individual outcomes effect (positively or negatively) organizational outcomes.

The updated model [10] is composed of the following dimensions: information quality, system quality, service quality, use and intention to use, user satisfaction, and net benefits.

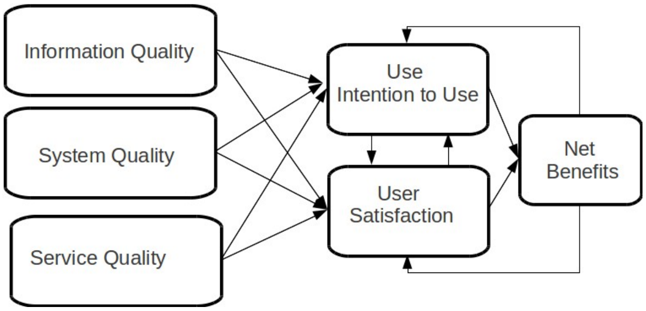


Figure 2: Delone & McLean Model 2003[10]

Both initial model and the updated model were tested, validated, adapted and extended by many researchers.

In this kind of systems we found difficult to separate between system quality and information quality. In fact, what is considered as information quality may be identified as system quality. On the other hand, service quality is also a misleading dimension. In fact, service may include not only the the service provided by the consultant, service provider, information system department, but also a set of support systems (including people or not). In those systems we may consider manual (printed or digital), wiki systems or forum systems.

4 Proposed Evaluation Model

The proposed model is composed by the following dimensions: support quality, system quality, use or Intention to Use, user satisfaction and net benefits.

The Support Quality is a dimension that measures the quality of the information generated and provided by the system. It's important that the users trust this information, which must be relevant, complete and consistent to preform programing tasks. In this dimension it considered whether students find useful the existence of extra support by providing information. Here it is used in this dimension two types of support: reference manual support; examples and guidance support to the learners.

System Quality is related to the measure of the technical success. This dimension evaluates the characteristics of the application in terms of providing a good support to write code languages and the way it displays the results of students programming lines.

Use or Intention to Use is a dimension that measures the expected behavior of a user, an attitude of intention of using a system. In this dimension we have the analysis of the students' usage intention in terms of future usage for illustrating programming, basic commands, structures, variables or functions.

User Satisfaction evaluates the perception that students have about the system and their experience by using it. If the user is happy with the system, he will use it again. This dimension measures the system success in terms of real usage, the effective use experience. Therefore it is considered here the actual usage experience by the students, whether they liked to use system to learn programming, variables and functions.

Net Benefits is a dimension that intends to measure the effective success of the system by the analysis of the impacts that it has in an individual and in organization levels. We can say that the individual and organizational impacts are part of this dimension. In this context students' learning objectives are assessed, in terms of their own success in learning programming, increasing performance, time saving in learning and university usefulness in teaching programing.

So the relations between the dimensions are specified in the Figure 3. These relations were tested by the authors of the model by others and should be used by the companies that implement this kind of system to evaluate their performance.

Supported in the same rational that is behind figure 2, this model has the following prepositions:

- System Quality positively influences Use

- Support Quality positively influences Use

- User Satisfaction positively influences Use

- Net benefits influences Use

- Support Quality positively influences User Satisfaction

- System Quality positively influences User Satisfaction

- Use positively influences User Satisfaction

- Net benefits influences User Satisfaction

- Use positively influences Net Benefit

- User Satisfaction positively influences Net Benefit

Summarizing, the model proposed is essentially the model of figure 2. The main difference is that there is no service quality or information quality. The concept presented here is the concept of support.

5 Empirical Work

To assist student learning, various visualization approaches have been proposed, used and tested. Visualizations can leverage understanding and learning of abstract models and their applications in a program. [13] [5]; [43] Visualization provides a different interface between learned abilities, and can improve attitude towards programming. There are multiple visualization tools available, such as graphical interfaces, games, robots and virtual robots.

There are several platforms that were developed with the objective of facilitating student learning. We analyzed a set of those applications, which are used by instructors to teach the fundamentals of programming. One of the most popular, (referred to by many authors and used by several studies), is Karel. Dated from 1981[18], and developed by Pattis at Stanford University, there have also been other variations of Karel,[26] and other systems developed in other universities from 1995 to 2011.

We purpose a solution where students interact with an editor, having as a result, an output.

We developed a reference manual, a topic manual, and samples as supporting documentation.

The reference manual is a simplified manual consisting mainly of a list of procedures.

The topic manual is text explaining each topic (basic procedures, conditional constructs, looks, and variables), with simplified examples.

The samples are some examples, including a small case and a solution.

The system was developed using the user interface developed by Jonoski Pavle and Jasmakovska Fani, with some additional functionalities added. This system has a dashboard. Displays on the right side of the screen a grid filled with squares. At the bottom left of the screen stands a dragon that waits for instructions to move along that same grid. The dragon can go to any direction according to the instructions given on the left side of the screen. If a student wants the dragon to move to the right, he/she must push the green right arrow. As shown in the Figure, students have a small number of available instructions, corresponding to a program control. Each time a student selects a button a code line appears on the code editor. On the left side of the screen there is a list that summarizes the support contents available to the students. It presents the dragon and the game objective. There are available a list of guides and examples. This feature allows the student to become familiarized with code lines. The system also has different patterns that correspond to various levels of difficulty, within the exercises. This system has a reference manual, exercises and a manual.

An empirical study was conducted to understand if a virtual robot could be used as a system to enhance and facilitate programming learning. The perused empirical work objectives were the following:

1. Information systems success emergent dimensions’

2. Verify whether the encountered dimensions correspond to the DeLone & McLean Model

3. Verify if dimensions have effects on other dimensions

6 Data Presentation & Results Discussion

In the empirical work, it was used the proposed system shown in figure 2. The system was presented to a group of first-year computer science students. After the system presentation, students were asked to do a set of programming exercises using the mentioned system. The programming exercises were performed during two, 90 minutes, face to face classes. For last, students were asked to complete a questionnaire; the number of valid answers was 78. Each answer used a Likert scale of 1 to 7 (1- profoundly disagree …7- completely agree).

The questionnaire was structured according to the DeLone and McLean [10] dimensions: information quality, system quality, use/intention to use, user satisfaction and net benefits. In the questionnaire, the service quality dimension was not measured. Therefore, it was considered that students could feel some pressure in answering favorably to possible questions about the lectured classes and their teacher.

Concerning types of support, the reference manual was considered to be the least preferred type of support. Examples and the manual were most preferred by the students. This indicates that the students were not familiar with the subject.

6.1 Factor analysis

In order to identify the main dimension we use factor analysis and Promax rotation. We adopted Promax rotation as long as it allows the factors to be correlated with one another. We obtained 6 factors.

Table 3: Factor Analysis

	Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
P01	0.233	0.4		0.197		-0.113
P02	0.151	0.611		0.239		
P03		0.643		0.19		
P04		0.885				
P05		0.789				0.133
P06		0.757				
P07		0.929		-0.129		
P08	-0.154	0.903				
P09		0.872		-0.146		

	Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
P10		0.765	0.147	-0.142	0.171	
P11					0.8	
P12					0.9	
P13	-0.124	0.231		0.128	0.685	
P14		0.183		-0.162	0.893	
P15		0.135		0.117	0.754	
P16	0.797	-0.219				
P17	0.787	-0.141	0.215	-0.288	0.115	
P18	0.884					
P19	0.924			-0.138		
P20	0.837	-0.15	-0.272			0.348
P21	0.905					-0.148
P22	0.722	-0.149	0.169		0.109	
P23	0.613	0.157		0.179	-0.133	
P24	0.702	0.322				
P25	0.532	0.358				
P26	0.629	0.262		0.108	-0.103	
P27	0.638	0.194				
P30			0.927	0.147		-0.136
P31	-0.13		0.932	0.139		
P32			0.989			
P33			0.955	-0.114		
P34	0.128		0.651	-0.111		0.361
P35	0.119		0.796	-0.102		
P41	-0.131	0.126	0.111	0.141		0.876
P42		0.18	0.13	0.228		0.521
P43				0.176		0.738
P44	0.147		-0.158	0.723	0.112	
P45		0.121		0.678	0.103	
P46			0.194	0.859	-0.158	
P47		-0.117		0.997		0.12
P48			0.108	0.601	0.118	
P49		-0.325		0.696	0.125	0.117

The following table present the summary of factorial analysis results.

	Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
SS loadings	7.148	6.744	5.005	4.176	3.494	2.025
Proportion Var	0.17	0.161	0.119	0.099	0.083	0.048
Cumulative Var	0.17	0.331	0.45	0.549	0.633	0.681

Test of the hypothesis that 6 factors are sufficient. The chi square statistic is 1019.42 on 624 degrees of freedom. The p-value is 1.55e-21

The interpretation of the factors identified is coherent with the literature and previous experiences. The name attributed to each factor may be: (F1) System quality (including either system/performance quality and information/output quality); (F2) Support quality (examples) (F3) Use and Intention to use; (F4) Net Benefits; (F5) Support quality (reference manual); (F6) Satisfaction.

Results are related to our proposal. Nevertheless, two kinds of support were identified (examples and reference manuals).

6.2 Impact on Net Benefits

We performed a regression to analyze the impact of F3 (Use and Intention to use) and F6 (Satisfaction) on F4 (Net Benefits).

Coefficients:

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	8.009e-17	9.036e-02	0	1
F3	4.898e-01	1.052e-01	4.655	1.39e-05 ***
F6	2.600e-01	1.057e-01	2.46	0.0162 *

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error is 0.7929 on 74 degrees of freedom. Multiple R-squared is 0.4219, Adjusted R-squared is 0.4062 and F-statistic: 27 on 2 and 74 DF, p-value: 1.568e-09.

Results from support the following statements: (1) Use positively influences Net Benefit and (2) User Satisfaction positively influences Net Benefit

6.3 Impact on Use/Intention to Use

Coefficients:

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	2.744e-16	8.223e-02	0	1
F1	1.188e-01	1.133e-01	1.049	0.29784
F2	5.361e-02	1.134e-01	0.473	0.63783
F4	2.165e-01	1.178e-01	1.838	0.07029 .
F5	2.812e-01	1.079e-01	2.607	0.01114 *
F6	2.625e-01	9.914e-02	2.648	0.00998 **

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

The residual standard error is 0.7215 on 71 degrees of freedom. Multiple R-squared obtained is 0.5231 and the adjusted R squared: 0.4895. The F-statistic is 15.58 on 5 and 71 DF, p-value: 2.524e-10

We performed a regression to analyze the impact of F1 (System quality) F2 (Support quality – examples), F4 (Net Benefits), F5 (Support quality - reference manual), F6 (Satisfaction) on F3 (Use and Intention to use) .

Results from support the following statements: (1) Support Quality (manual) positively influences Use and (2) User Satisfaction positively influences Use

6.4 Impact Satisfaction

We performed a regression to analyze the impact of F1 (System quality) F2 (Support quality – examples), F4 (Net Benefits), F5 (Support quality - reference manual) and F3 (Use and Intention to use) on F6 (Satisfaction)

Coefficients:

	Estimate	Std. Error	T value	Pr(>|t|)
(Intercept)	-4.112e-16	9.391e-02	0	1
F1	2.443e-01	1.271e-01	1.922	0.05860 .
F2	-1.466e-01	1.285e-01	-1.14	0.25792
F3	3.423e-01	1.293e-01	2.648	0.00998 **
F4	3.022e-01	1.329e-01	2.273	0.02604 *
F5	-1.377e-01	1.279e-01	-1.077	0.28528

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

The residual standard error is 0.824 on 71 degrees of freedom. The multiple R-squared: 0.3721, and the Adjusted R-squared is 0.3279. F-statistic is 8.415 on 5 and 71 DF, p-value: 2.725e-06.

Results from support the following statements: (1) Use positively influences User Satisfaction and (2) Net benefits influences User Satisfaction.

6.5 Model Discussion

The following statements are statistically supported as it is declared in the last section: (1) Use positively influences User Satisfaction; (2) Use positively influences Individual Impact; (3) System Quality positively influences Use; (4) Support Quality positively influences Use; (5) User Satisfaction positively influences Use; (6) System Quality positively influences User Satisfaction; (7) Support Quality positively influences User Satisfaction; (8) User Satisfaction positively influences Individual Impact; (9) Individual Impact positively influences Organizational Impact

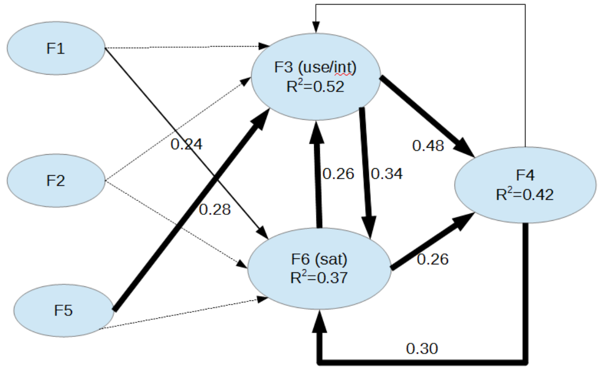


Fig. 3 – Final Model

7 Conclusions

The purpose of this paper is presenting the success factors concerning the usage of a system related to education context. In this paper, it is presented a model supported in the literature review (specially Delone & McLean success model 1 [10]. The empirical work consists of evaluating the model. According to the literature review, there are several challenges related to the learning of computer programming. Some of them may be partially answered by utilizing virtual robots. We also analyzed the impact of the use of this framework, and identified that some features are more effective than others. The usage of the reference manual (factor 5) has an impact on the system usage or on the intention to use the system. The use/intention to use (factor 3) is more relevant than user satisfaction (factor 6) to the system success. It was verified that success (factor 4) is important to the user satisfaction (factor 6), and the system feedback plays an important role in the user satisfaction (factor 6).

8 Acknowledgements

Partially financed by FCT (Fundação Ciência e Tecnologia).

9 References

[1] Ajzen, I. (1991). The theory of planned behavior. Organizational Behavior and Human Decision Process, 50, 179–211.

[2] Ala-Myka, K. (2005) “Problems in Learning and Teaching Programming- a literature study for developing visualizations in the Codewitz-Minervapoject” Codewitz Needs Analisys, lIteratur Study

[3] Alice (2000) available at: http://www.alice.org/index.php?page=what_is_alice/what_is_alice

[4] Brusilovsky, P., Calabrese, E., Hvorecky, J., Kouchnirenko, A., and Miller, P. (1997) Mini-languages: A Way to Learn Programming Principles. Education and Information Technologies 2 (1), pp. 65-83.

[5] Cardellini, V., . Colajanni, M, and Yu, P.S. (2000) “Dynamic Load Balancing on Web-Server Systems”, IEEE Internet Computing , May – June 2000. pp. 28-39

[6] Costa, C. J., Aparicio, M., & Cordeiro, C. (2012). Web-Based Graphic Environment to Support Programming in the Beginning Learning Process. Em M. Herrlich, R. Malaka, & M. Masuch (Eds), Entertainment Computing - ICEC 2012 (pp 413–416). Springer Berlin Heidelberg.

[7] Costa, C. J., Aparicio, M., & Cordeiro, C. (2012). A solution to support student learning of programming. Em Proceedings of the Workshop on Open Source and Design of Communication (pp 25–29). New York, NY, USA: ACM.

[8] Cummings, S. (2010) “Feedback 2.0: An Investigation into using sharable Feedback tags as Programming Feedback” Ph.D. Thesis, Durham University

[9] DeLone, W. H., & McLean, E. R. (1992). Information systems success: The quest for the dependent variable. Information Systems Research, 3(1), 60-95.

[10] DeLone, W. H. e E. R. McLean. (2003). “The DeLone and McLean Model of Information System Success: A Ten-Year Update”. Journal of Managemet Information Systems 19, No. 4, 9–30.

[11] Donmez, O. & Inceoglu, 2008 “A Web Based Tool for Novice Programmers: interaction in use”

[12] Dorn, B. & Sanders, D. (2003) “Using Jeroo to Introduce object-oriented programming” in 33rd ASEE/IEEE Frontiers in Education Conference, November 5-8

[13] Fowler, M., & Beck, K. (2012). Refactoring: improving the design of existing code. Boston: Addison-Wesley.

[14] GlowScript (2010) available at: http://www.glowscript.org

[15] Jenkins, T. (2002) “On the difficulty of Learning to Program” ;university of Leeds, available at: http://www.psy.gla.ac.uk/~steve/localed/jenkins.html

[16] Jeroo: B. Dorn and D. Sanders. “Using Jeroo to introduce object-oriented programming”. In FIE '03: Proceedings of the 33rd annual Frontiers in Education Conference, volume 1, pages T4C 22-27, 2003.

[17] Jerro (2003) available at: http://home.cc.gatech.edu/dorn/jeroo

[18] Karel (1981) Available at: http://www.stanford.edu/class/cs106a/

[19] Knobby available at: http://www.cse.lehigh.edu/~glennb/um/GetKnob.htm

[20] Kordaki, M. (2010). A drawing and multi-representational computer environment for beginners’ learning of programming using C: Design and pilot formative evaluation. Computers & Education, 54(1), 69–87.

[21] Lahtinen, E., Ala-Mtka, K., Jarvinen, H. (2005) “A Study of the Difficulties of Novice Programmers” in ITCSE 2005 proceedings ACM, Portugal

[22] Milne, I., & Rowe, G. (2002). Difficulties in Learning and Teaching Programming&mdash; Views of Students and Tutors. *Education and Information Technologies*, *7*(1), 55–66.

[23] Monty Karel (2008) Available at: http://csis.pace.edu/~bergin/MontyKarel/index.html

[24] Negroponte, N. (1996). *Being Digital* (1st ed.). Vintage.

[25] Pattis, Richard E., (1981) Karel the Robot: A Gentle Introduction to the Art of Programming, John Wiley and Sons: New York

[26] Bergin,J. Stehlik, M. Roberts, J & Pattis, R.(1996) Karel++: A Gentle Introduction to the Art of Object-Oriented Programming. John Wiley & Sons.

[27] Pears, A., Seidman, S., Malmi, L., Mannila, L., Adams, E., Bennedsen, J., Devlin, M., et al. (2007). A survey of literature on the teaching of introductory programming. *SIGCSE Bull.*, *39*(4), 204–223.

[28] Piech, C. & Roberts, E. (2011) “Informtatics Education Using Nothing but a browser” in ICTand Informatics in a Globalised World of Education, Kenya

[29] Radosevic, D.; Orehovacki, T. & Lovrencic, A. (2009) “Verificator: Educational Tool for Learning Programming,” in Informatics in Education, 2009, vol. 8, No 6, 261-280

[30] Renumol, V.; Jayaprakash, S. & Janakiram, D. (2009) “Classification of cognitive difficulties of students to learn computer programming” Indian Institute of Technology, India

[31] Rist, R. S. (1996) “Teaching Eiffel as a Firts Language”, Journal of Object-Oriented Programming, 9,30-41.

[32] Roberts, E. (2005) “Karel the Robot” Department of Computer Science, Stanford University

[33] Robins, A. ; Rountree, J. & Rountree, N. (2003) “Learning and teaching Programming: a review and discussion”, Computer Science Education, 2003,vol 13, n2, pp137-172

[34] Robins, A., Rountree, J. & Rountree, N (2003) “Learning and Teaching Programming: A Review and Discussion”, in Computer Science Education, 2003, Vol.13, nº2, pp. 137-172

[35] RoBOTL (1995) available at; http://users.rcn.com/n1hbr/robotl/robotl.html

[36] Sajaniemi, J. & Hu, C. (2006) “Teaching Programming: Going beyond “Objects First” in 18th Workshop of the Psychology of Programming Interest Group, University of Sussex, September 2006 , pp-255-265

[37] Sajaniemi, J., Hu, C., (2006) “Teaching Programming: Going beyond, objects first” in P. Romero, J. Good, E. Acosta Chaparro & Bryant Eds Proceedings PPIG18, University of Sussex

[38] Sanders, D. & Dorn., B. (2003) “Classroom experience With Jeroo”. Journal of Computing Sciences in Colleges, 18(4) pp. 308-316

[39] Sanders, D. & Dorn., B. (2003) “Jeroo: A tool for teaching object-oriented programming”. In SIGCSE '03: Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education, pp 201-204

[40] Stamouli, I., Begum, M., Mancy, R. & Reid, N. (2005) “ExploreCSEd: Exploring Skills and Difficulties in Programming Education”, *Final Report of the Findings for Higher Education Academy*, October 2005

[41] Stanford (2012) “Why should we use Karel to teach programming?”, available at:

http:// bermuda.stanford.edu/curricula/FAQ/Karel.html

[42] Soloway, E., & Spohrer, J. C. (1988). Studying the Novice Programmer. Lawrence Erlbaum.

[43] Thomas, L., Ratcliffe, M., Woodbury, J. and Jarman, E. (2002). Learning Styles and Performance in the Introductory Programming Sequence. Proceedings of the 33rd SIGCS E Symposium, pp. 33-42.

[44] VPython (2000) available at: http://vpython.org/

Effects of Group Learning in Programming Class

Youngmi Kwon[1]
[1]Department of Information Communications Engineering, Chungnam National University,
Daejeon, South Korea

Abstract – *In this paper, the effects of group learning were investigated when group learning and self-study are applied to programming language classes in computer science area. C programming language course and Data Structures course were selected for experiments as a beginner's level class and intermediate level class. Havruta method consisting of 2 or 3 members and group activity method consisting of 8 to 12 members were applied for a group self-learning. This paper focused on the adequate number of group activities, the extent of students' preference to group activities, and the difficulties that students feel as obstacles in group activity. Mediocre class compared to the beginner class preferred group activity in a great extent. But students show higher preferences for some topics rather than they show the even interest in all the topics. So instructors need to develop the adequate topics for the group activity according to the level of students.*

Keywords: Data Structures, C Programming, Group Learning, Group Activity, Havruta

1 Introduction

Programming language classes generally consists of instructor's teaching and students' experiments. This paper investigated the effects of group self-learning in the programming language class. C language class as a beginners' class for freshmen and Data Structure class for sophomore were chosen as sample classes. C programming language class teaches language itself and the Data Structure class uses programming language as a tool to implement the algorithms. The understanding of programming language is completed with PC-experiments based on the understanding of grammar and principles. For these two classes, this paper applied two types of group activities: the one is Havruta – discussion style self learning process and the other is group activity which makes team video to explain some concepts using their body, action, and some resources. For these group activities, the level of satisfaction and the factors of difficulties were evaluated analyzed. In chapter 2, Havruta discussion-based self-learning activities and group activity in Data Structure class are introduced. In chapter 3, the results of group activities were presented and the results are analyzed in chapter 4. Chapter 5 is the conclusion of this paper.

2 Group Learning in Programming Class

2.1 Class Selection

C programming class [1] is open for the freshmen. So it is the first programming language course for the very beginners. Data Structures class [2, 3] is to understand the various data structures and algorithms applied on those structures. It uses C/C++ or Java as an already-acquired-tool to implement the concept. To evaluate the effects of the group learning in programming language classes, these two classes are chosen, and havruta and group activity were applied. Havruta is to understand concepts based on the discussion with the partner. The group activity is to understand not by legacy programming but by body expressions.

2.2 Havruta

Havruta is a tool which has been used in teaching Jewish Torah. It lets two persons discuss and help understand the Torah [4, 8]. Havruta means fellowship. Two partners help each other to check which part they don't know and which part they do. In this paper, this Havruta method was applied only in case the students don't understand well even after they listened to the instructor's explanation for some concept. Because it takes too long time to proceed the class with Havruta for every single concept.

2.3 Group Activity

Havruta method can be applied to all the programming subjects. Additional method is a group activity consisting of more than 5 members to express some concept with their own way. In this paper, two sorting algorithms were assigned to the team of 8~12 students. They took an idea brain storming meeting to complete the project. The well made video was requested as a final result of team project. This activity was triggered by [5-7]. The existing public results were made by specialized dancers and video experts. But all the students are neither dancers nor experts in handling video. So they try to find out their own way for it very actively. Many students enjoy this process and keep the memory as very valuable and interesting activity for long time.

3 Experiment & Results

3.1 Havruta Preferences

In the beginners' class, the Havruta combined method was not much preferred. It was 46.4%. But the beginners preferred the instructor's purely repeated and detailed explanation which was 53.6% as shown in Table 1. But in the intermediate level class, 100% students preferred the Havruta combined method of teaching as shown in Table 2.

Table 1. Preferences to Havruta for Beginner

Favorite method in class (1st yr)	# of selections	Percentag e
a. Instructor's explanation only	0	0%
b. Instructor's explanation + experiment	15	**53.6%**
c. Havruta + experiment	0	0%
d. Havruta + Instructor's explanation + experiment	13	**46.4%**
e. Alone reading	0	0%

Table 2. Preferences to Havruta for Mediocre

Favorite method in class	# of selections	Percentag e
a. Instructor's explanation only	0	0%
b. Havruta + Instructor's explanation	42	**100.0%**

3.2 Adequate Number of Members in Havruta

In the intermediate level programming class which all the students preferred Havruta, 71.4% students answered that two members are most adequate for a discussion. But 28.5% students wanted more than 3 persons to discuss. It is not the insignificant ratio, too. Table 3 shows it.

Table 3. Adequate Number of Members in Havruta

Options	# of selections	Percentag e
1 person	0	0.0%
2 persons	30	**71.4%**
3 persons	8	**19.0%**
more than 4	4	9.5%

3.3 Difficulty Factors in Havruta

The most difficult factor that students voted was the situation that both students in a discussion group are lack of understanding the new concept resulting in no progress. Table 4 shows that in beginners' class, 82.1% students answered for this. Table 5 shows that in mediocre level class, 78.6% students answered for this. The value itself is lower than that of the beginners' class, but they show the consensus for the most difficult factor. Besides this, the lack of discussion time was ranked the 2nd position. It was 10.7% in the beginners' class and 21.4% in the intermediate class. Rather than a simple programming language class, the mediocre class which demands more complex understanding for complex algorithms showed higher requirement of longer time.

Table 4. Difficulties in Havruta for Beginner

Difficulties in Havruta	# of selections	Percentage
a. Both students can't understand the concept	23	**82.1%**
b. Given time is too short for discussion	3	10.7%
c. Lack of freedom to choose my pair	2	7.1%
d. Unfamiliarity with this kind of self learning	0	0.0%
e. Fond of studying alone	0	0.0%

Table 5. Difficulties in Havruta for Mediocre

Difficulties in Havruta	# of selections	Percentage
a. Both students can't understand the concept	33	**78.6%**
b. Given time is too short for discussion	9	**21.4%**
c. Lack of freedom to choose my pair	2	4.8%
d. Discussing noise	3	7.1%
e. etc.	2	4.8%

3.4 Group Activity Preferences and the Quality of Results

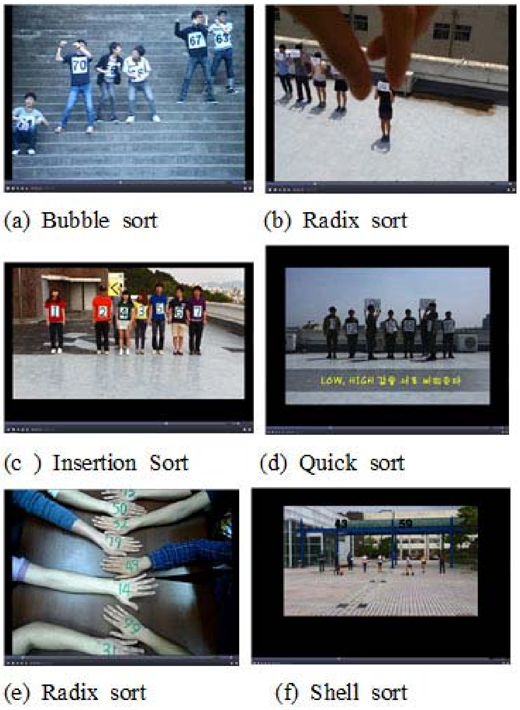
As shown in Table 6, there are small group of 7.1% who prefer the personal solution process through programming without group activity. They are introversive and/or excellent students compared to the 92.9% students who prefer the group activity.

All the 92.9% group activity supporters don't show the even preferences to the every kind of sort. They have preferred items (bubble, insertion, selection and merge) and non-preferred items (quick and shell). The resulting output samples are shown in Fig. 1.

But the preferences are not directly connected with the quality of the results. Radix sort was ranked as low preferences, but the quality of the output was generally very satisfactory.

Table 6. Preferences for Group Activity vs. Personal Programming Learning

Group Activity Preferences	Personal Programming Preferences	Percentage
39		**92.9%**
	3	7.1%


(a) Bubble sort (b) Radix sort

(c) Insertion Sort (d) Quick sort

(e) Radix sort (f) Shell sort

Figure 1. Samples of Favorable and Unfavorable Sort Expressions

3.5 Adequate Number of Members in Group Activity

In the experiment, minimum 8 students and average 10 students were clustered into one group to provide sufficient number of members to express some concept. After the group activity, 50% of students answered 8 is the most adequate size of a group as shown in Table 7. The 21.4% of students answered the 6 is the most adequate. And the students of the same percentage showed preferences for the number of students to be 10. When the group activity is led by a few active students, and most of the students are passive in activity, the passive students answered that the small number of group member is enough for the group activity.

Table 7. Adequate # of Members in Group Activity

Options	# of selections	Percentage
2 persons	1	2.4%
3 persons	2	4.8%
4 persons	5	11.9%
5 persons	5	11.9%
6 persons	9	**21.4%**
7 persons	6	14.3%
8 persons	21	**50.0%**
9 persons	8	19.0%
10 persons	9	**21.4%**

3.6 Difficulty Factors in Group Activity

As shown in Table 8, students showed the most difficulty in video handling (57.1%). If instructor does not require the video as the final output, this will be solved. The second difficult factor is that professor assigns the kind of sorts to the group and the students don’t have freedom in selecting the sort types for the group activity. But, if the freedom is given to the group, only some specific preferred sort methods would be selected as shown in Table 9. In that case, the group activities for the various concepts sharing would not be guaranteed. So, this difficulty should be handled carefully.

In the experiment, two weeks were given to complete the activity. But 47% students answered that it is too short. The actual problem is that the chapter for the group activity appears in the end of the semester. So it is not easy to give further longer days for group activity. To solve this difficulty, the topic change can be one option.

Table 8. Difficulties in Group Activity

Difficulties in Group Activity	# of selections	Percentage
a. Lack of members	0	0.0%
b. Lack of resources	7	16.7%
c. Lack of ability in video handling	24	**57.1%**
d. Lack of idea	7	16.7%
e. Lack of leader's leadership	0	0.0%

f. Lack of given days	20	**47.6%**
g. Lack of freedom to choose sorts	21	**50.0%**
h. Difficulty in use of location	9	21.4%
i. etc.	0	0.0%

Table 9. Preferred Sort Types

Sort	Good to Express	Percentage	Difficult to Express	Percentage
bubble sort	38	**90.5%**	2	4.8%
radix sort	3	7.1%	4	9.5%
merge sort	18	**42.9%**	6	14.3%
insertion sort	28	**66.7%**	0	0.0%
selection sort	19	**45.2%**	3	7.1%
quick sort	8	19.0%	13	**31.0%**
shell sort	7	16.7%	18	**42.9%**
heap sort	0	0.0%	2	4.8%

4 Discussion

There was small percentage of 7.1% students to prefer the personal programming solving process to the group activity. But most students prefer the Havruta and group activity. But for the Havruta, it was not shown that students prefer it at any circumstances. As we can see in Table 1, the very beginner group showed the higher priority on the instructor’s explanation than Havruta discussion method. But as in Table 2, the intermediate group students showed much higher preference for the Havruta method. The most difficult factors for the Havruta is the students’ overall low level of understanding. When both partners are not good at the concept which they are trying to understand, they felt it is not effective (82.1% in Table 4 and 78.6% in Table 5). The preferred size of the Havruta is two (71.4 % in Table 3), and the preferred size of the group activity is 8 (28.5 % in Table 3).

5 Conclusions

In the programming class, Havruta and group activities can be applied in many ways. But for the beginners with low knowledge, the Havruta discussion based method can be kept from progress. So instructors should decide which group activity could be effective in each subject and for each student group. Group activity themes can be developed more actively year by year by instructor. This paper stated the difficult factors for the group learning and some guidelines to solve them. This paper showed that the various cooperative method of learning can be effective even in programming class which requires individual computing works. In Data Structure class, linked list, stack, queue and tree concepts can be the resources for the group activity as well as sort.

6 References

[1] WS Kang and YH Shin, Perfect C, Infinity Books Press, 2010.

[2] E. Horowitz, et al, Fundamentals of Data Structures in C, 2nd Ed., Computer Science Press, 2008.

[3] In Kook Cheon, Data Structures in C, Saeng Reung Press, 2009.

[4] Orit Kent, A Theory of Havruta Learning, Journal of Jewish Education, vol. 76, no.3, p.215-245, 2010.

[5] Select sort, http://www.youtube.com/watch?v=Ns4TPTC8whw , 2011.

[6] Quick sort, http://www.youtube.com/watch?v=vxENKlcs2Tw, 2009.

[7] Shell sort, http://www.youtube.com/watch?v=CmPA7zE 8mx0, 2011.

[8] Rachel. G. Schultz, Havruta: Learning in Pairs, http://www.myjewishlearning.com/practices/Ritual/Torah_Study/How_to_Study_Torah/Havruta_Learning_in_Pairs_.shtml

A Process for Updating Computer Science Curriculum for Non-Majors

Melis Öner, David Kaczynski, and Roger Lee

Central Michigan University
Department of Computer Science
Mount Pleasant, MI 48859, USA
(oner1m, kaczy1da, lee1ry)@cmich.edu

***Abstract*—Computers permeate society deeper and more thoroughly as computer science and adjacent fields progress. Students who graduate from institutions of higher education are expected to adapt to and utilize computer technologies, regardless of major. Many universities already offer computer science courses for non-majors, but little research exists on methods to continuously update existing curriculum. We propose a method for updating computer science curriculum for non-majors by identifying course objectives that are directly tied to those recommended by the ACM and IEEE Joint Task Force for Computing Curricula. This process addresses how to identify and bridge the gaps between existing and new course objectives and how we measure the impact of the course using computer attitude surveys. Our research finds that universities may be able to utilize the methods outlined in this paper to update their computer science curriculum for non-majors with confidence.**

***Index Terms*—computer science education, non-majors, IT key qualifications, curriculum, computers and society**

I. Introduction

Information and software technologies are growing everyday. Today, not only computer engineers and computer scientists use software. From photographers to physicians, every profession utilizes from various numbers of softwares and information technology tools. This fact enhances the importance of being computer literate. Computer literacy is defined in [Gupta] [**?**] as individual ability to employ a computer system, have a basic understanding of the operating system, manage file and disk operations, use computer applications for individual or job-related tasks, and use the Internet for communication, information acquisition, or commerce purposes. Being computer literate is not only knowing how to use the tools; computer literate adults must be able to deploy these activities for academic work and job-related tasks.

Computer literacy is more than surfing on the Internet or playing computer games. A computer literate person uses his or her computer to work more efficiently. We cannot limit the benefits of computer literacy to job- or academic-related tasks, either. Computer literate people can use computers for various casual tasks, such as querying weather or traffic conditions, shopping, or communicating with friends and family. There are all examples of daily computer uses in social and vocational life.

Central Michigan University offers a computer science course for non-majors: CPS 100, or Computers and Society. The course has an enrollment of several hundred students each year, largely due to being a compulsory course in a variety of degree programs. The course covers a wide range of topics, including:

- history of computer hardware and software,
- societal impacts of computer technology,
- applications of computer technology,

- proficiency in navigating the Internet,
- computer security and risks, and
- proficiency in Microsoft Office and Microsoft Windows software.

CPS 100 is taught with equal parts of lecture and lab periods. In the case of the lab, the exercises have not been updated for several years. This has lead to many lab exercises being inoperable due to outdated instructions for user interfaces for software products, such as Google and Windows.

Computer technology is developing faster everyday, and computer science education must be updated depending on the new developments. ACM and IEEE Computer Science joint Task Force publishes Computer Science Curricula at regular intervals. This generally-accepted guide encouraged us to ask this question: Do our current lab exercises supply the outcomes defined in Computer Science Curricula 2013? We decided to use our outdated lab exercises as an opportunity to overhaul our curriculum with modern computer science learning objectives.

In order to effectively update the curriculum for CPS 100, we have three research objectives:

1) Identify guidelines for modern computer science curriculum
2) Define a process for bridging the gaps between current course content and the new curriculum
3) Measure and analyze the progress of students

For the purposes of this research, we define the methods used to meet our objectives and leave the results of implementation to future research.

II. Background

A. ACM and IEEE Computer Science Curricula Guidelines

ACM and IEEE revise recommended computer science curricula every 10 years [?]. The curricula guidelines are volunteered by institutions of information technology in both industry and academia from all around the world. The Joint Task Force define a set of underlying principles to guide the construction of the curricula. Several of these principles specifically influenced us to adopt their recommendations for our non-majors course: "Computer Science curricula should be designed to provide students with the flexibility to work across many disciplines. . . CS 2013 must provide realistic, adoptable recommendations that provide guidance and flexibility, allowing curricular designs that are innovative and track recent developments in the field. . . [and] The CS2013 guidelines must be relevant to a variety of institutions."

B. Designing a Computer Science Course for Non-Majors

Guzdial and Forte [?] define a process for designing a computing course for non-majors: set objectives, choose a context, set-up feedback process, define infrastructure, and finally, define the course content. The authors go one to describe how each of these steps apply to the create of a Media Computation course.

We may implement similar techniques as we take steps to revise and update our course material. However, the context for our course is already decided via the lecture materials. Also, we will be implementing additional methods for mapping our current class activities to the pending course curriculum guidelines in order to reduce unnecessary efforts in redesigning the course.

C. Incorporating ACM/IEEE Curricula Guidelines into Undergraduate Curriculum

Tewari and Friedman documented and published their results for updating the computer science curriculum at Temple University in the early 1990s [?], when software engineering was still young. They identify the flaws in their existing curriculum and the new objectives that they wish to incorporate. For software engineering, this included such topics as requirements development, incremental development, software architecture patterns, and more. These topics were elicited from 1991 version of *ACM/IEEE-CS Computing Curricula*. Tewari and Friedman go on to specify the changes to the course work and laboratories, instructional methods, and the early experiences of adopting the new curricula.

For our current research, we are discovering a method with which we will be updating our course's curriculum. The curriculum will not actually be updated until a future semester. We extract updated course objectives from the 2013 version of *ACM/IEEE-CS Computing Curricula*. At the time of this writing, the Irondraft v. 1.0 is the latest version of *ACM/IEEE-CS Computing Curricula*, and we hope that we may use the official release soon for our update process.

D. Applying Key Qualifications to IT Curriculum

In 2008, Dörge and Schulte published a paper [?] on applying the ten most common key qualifications to IT curriculum, producing a set of learning objectives that were independent of current technologies. These objectives can be summarized as self-reliance, self-training, using computers independently of operating systems, keeping up with the latest technological developments, adapting computer technology to fit one's own needs, recognizing dangers and risks for computer systems, using new trends and developments, and the ability to assess societal impacts of information technology (potentials and threats).

We find that several of our existing course objectives for our Computers and Society course highly resemble the aforementioned concepts derived by Dörge and Schulte, such as ability to communicate, flexibility, and capacity to solve problems. However, how key qualifications are extrapolated to specific assignments will differ from course to course. For example, we would like to extend operating system independence to office productivity software independence as well.

III. Goals of Updated Curriculum

We hope to see several benefits by updating our computer science curriculum in this manner. Because we are using international standards that represent the evolving interests of both industry and academia, we intend to create a foundation of course objectives that are supported by an international community of experts. Since the *ACM/IEEE-CS Computing Curricula* is constantly under review and being updated, we hope that we can revisit this relationship for future updates in course content as well.

Lab exercises with updated instructions and content may be target towards functioning productively in a computer-oriented work environment. With the updated curriculum and course content, we hope to alleviate social anxiety that students may have towards computer technology when beginning the class. We plan to establish a framework of surveying students at the beginning and end of each semester and measuring the students' attitudes towards the course and computer technology in general.

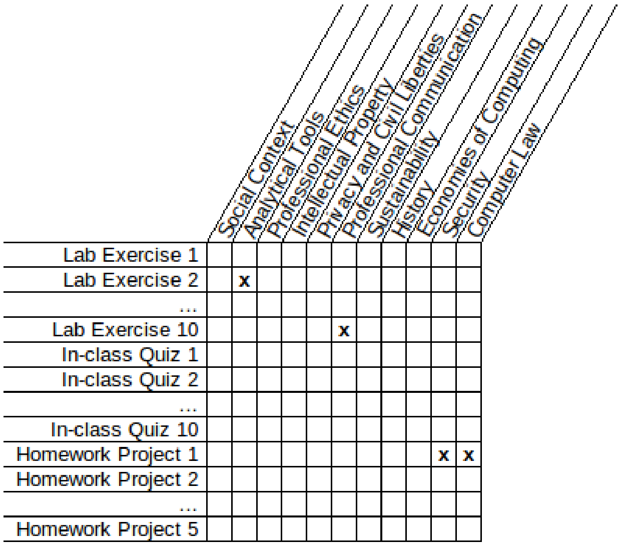
By explicitely defining our process for updating our curriculum and documenting our progress, future educators may avoid stagnant course content in CPS 100 by applying the same or a similar process. For example, future administrators may be able to consult the current version of *ACM/IEEE-CS Computing Curricula* for guidance in selecting course objectives, and our reseaerch may be consulted for the process of updating the current curriculum and content.

IV. Method

Here, we elaborate on the research objectives introduced in Section 1 and define our methods for meeting our objectives.

A. Identify Curriculum Guidelines

First, we must identify a set of topics to be addressed in the updated course curriculum. For this, we utilize technology-specific objectives as described by the 2013 ACM and IEEE Joint Task Force on Computing Curricula [?]. In the *ACM/IEEE-CS Computing Curricula*, The Social and Professional Practice subject is divided into sub-topics, and each sub-topic



	Social Context	Analytical Tools	Professional Ethics	Intellectual Property	Privacy and Civil Liberties	Professional Communication	Sustainability	History	Economies of Computing	Security	Computer Law
Lab Exercise 1											
Lab Exercise 2		x									
...											
Lab Exercise 10						x					
In-class Quiz 1											
In-class Quiz 2											
...											
In-class Quiz 10											
Homework Project 1										x	x
Homework Project 2											
...											
Homework Project 5											

Fig. 1. An example of a coverage matrix that helps identify gaps in existing curriculum versus the *ACM/IEEE-CS Computing Curricula 2013* [?]

is associated with a list of learning objectives. Elective and non-elective, all sub-topics can be seen in Figure 1. Each sub-topic has three parts: explanation of the sub-topic, topics to be covered, and learning outcomes. Each learning objective is labeled appropriately in Bloom's Taxonomy of Educational Objectives. We choose different outcomes to update our lab exercises depending on the Bloom's Taxonomy label and feasibility. CPS 100 is a non-major course, so we do not intend to apply all the learning outcomes in this course.

B. Bridge Gaps in Curriculum

Since there are existing course materials that may already satisfy the new curriculum guidelines that we are adopting, we have created a coverage matrix for identifying the gaps between current materials and the updated curricula recommendations.

We define a coverage matrix as a two-dimensional grid, as seen in Figure 1. On the x-axis, we list the subject areas that have been identified as defined in the previous step (Identify Curriculum Guidelines). The y-axis is a list of the current activities in the class, such as in-lab exercises, homework assignments, or other projects. For each activity, we indicate which areas (if any) of the new course objectives are satisfied. Note that if an activity is deemed inoperable due to outdated software interfaces or other technical issue, that activity is not considered to be covering any topic. Once all of the course activities have been added to the axis, we populate the matrix with Boolean check marks; either the activity covers the the topic, or it does not. After the matrix is appropriately populated, it may be determined which activities do not contribute to the course objectives and which course objectives still need to be satisfied by observing which rows or columns are respectively missing check marks.

Once the gaps in curriculum have been identified, we go on to bridge the gaps by following the relevant processes in course design defined by Guzdial and Forte [**?**], such as defining the infrastructure and building the activities.

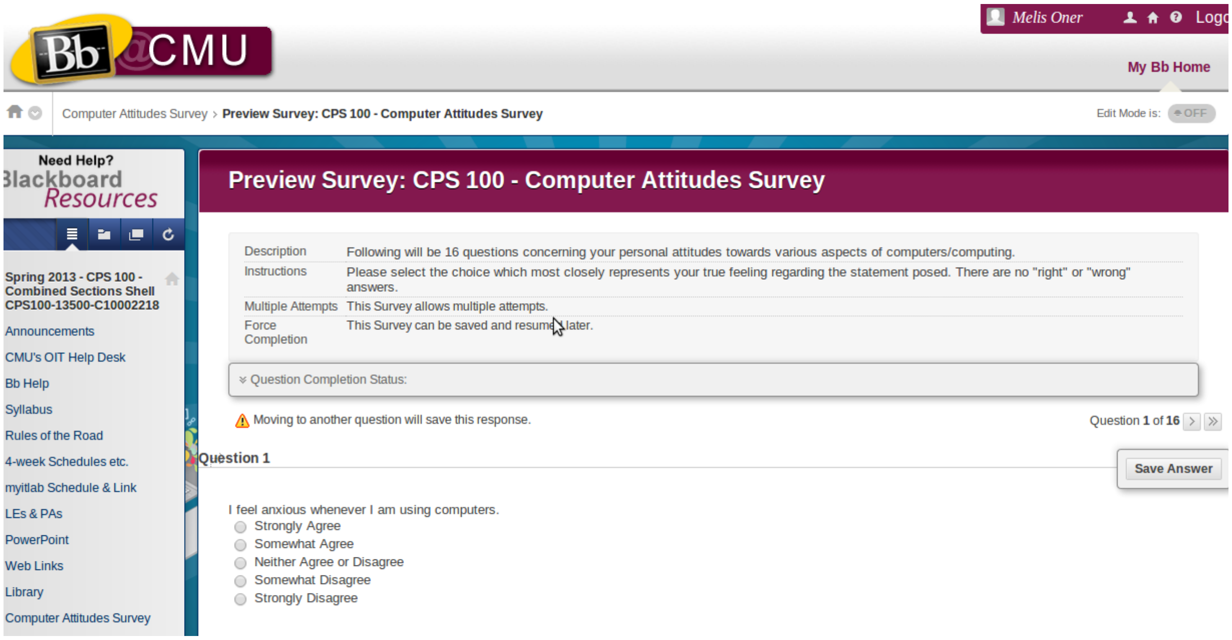
C. Measure and Analyze Progress of Students

To measure the progress of the students, we measure the attitudes of students towards computers via surveys. We currently survey students at the beginning of each semester for their attitudes towards computers. As it can be seen in Figure 2, Computer Attitudes Survey is on Blackboard, which is the main platform of CPS 100 course. This survey includes sixteen different statements about how students feel about working with computers and/or how they feel in the computer environment. Students are asked to answer questions on a five-level Likert scale. Students may choose one of the following levels:

- Strongly agree
- Somewhat agree
- Neither agree or disagree
- Somewhat disagree
- Strongly disagree

The following are questions from our student attitude survey:

1) I feel anxious whenever I am using computers.



Preview Survey: CPS 100 - Computer Attitudes Survey

Description	Following will be 16 questions concerning your personal attitudes towards various aspects of computers/computing.
Instructions	Please select the choice which most closely represents your true feeling regarding the statement posed. There are no "right" or "wrong" answers.
Multiple Attempts	This Survey allows multiple attempts.
Force Completion	This Survey can be saved and resumed later.

⚠ Moving to another question will save this response.

Question 1

I feel anxious whenever I am using computers.

○ Strongly Agree
○ Somewhat Agree
○ Neither Agree or Disagree
○ Somewhat Disagree
○ Strongly Disagree

Fig. 2. This survey is the Preview Survey of CPS 100 in Central Michigan University. All students have to take this survey as a part of Lab Exercise number one, first week. This survey has 16 questions and targets to measure attitudes of students towards computers.

2) I wish that I could be as calm as others appear to be when they are using computers.
3) I am confident in my ability to use computers.
4) I feel tense whenever I am working on a computer.
5) I worry about making mistakes on the computer.
6) I try to avoid using computers whenever possible.
7) I experience anxiety whenever I sit in front of a computer terminal.
8) I enjoy working computers.
9) I would like to continue working with computers in the future.
10) I feel relaxed when I am working on a computer.
11) I wish that computers were not as important as they are.
12) I am frightened by computers.
13) I feel content when I am working on a computer.
14) I feel overwhelmed whenever I am working on a computer.
15) I feel comfortable with computers.
16) I feel at ease with computers.

We plan to continue surveying students in this manner once the new curriculum has been updated and the changed implemented. Since this survey is on the Blackboard system, it can be applied at the end of the semester, too. By this way, we will compare the progress of students prior to the course changes with the progress of students after the course has been changed to identify areas where students attitudes lack improvement.

V. Conclusion

Information and software technologies are growing everyday. Today's technology developments require people from different professions to be computer literate. Central Michigan University offers a computer science course for non-majors and non-minors: CPS 100, or Computers and Society. The course covers a wide range of topics, including: history of computer hardware and software, societal impacts of computer technology, applications of computer technology, proficiency in navigating the Internet, computer security and risks, and proficiency in Microsoft Office and Microsoft Windows software. For the purposes of this research, we define the methods we will use to update our course activities and leave the results of implementation to future research.

We define our methods for meeting our objectives. We identify a set of objectives to be addressed in the updated course curriculum based on *ACM/IEEE-CS Computing Curricula*. In these guidelines, there is set of objectives for Social Issues and Professional Practices for computer science education. There are ten different sub-topics under this content. Each sub-topic has various learning outcomes, and they are labeled with Bloom's Taxonomy of Educational Objectives. Obtaining all of the learning outcomes of the Social Issues and Professional Practice would not be reasonable because a lot of sub-topics of this part are offered as electives, and they are considered as separate classes. Thus, we decided to choose the most applicable learning outcomes of our CPS 100 classes depending on their Bloom's Taxonomy labels and feasibility of a course for non-majors.

After defining our course objectives, we need to identify the gaps between our current curricula and the new course objectives. By this way we are going to see which course activities need to be changed and which course objectives still need to be addressed. A two dimensional coverage matrix is used for defining the gaps.

We plan to measure these changes' effect on the students by replicating the Computer Attitudes Survey on Blackboard at the end of the semester. Comparing students' feelings about working with computers and/or being in a computer environment at the beginning and at the end of the semester will give us an idea about the effectiveness of our implementation of course content that promotes computer literacy and aims to aleviate social anxiety towards computers.

VI. Future Research

For future research, we would like to document the processes defined in this paper as applied to the curriculum of CPS 100 at Central Michigan University as of the Spring 2013 semester. This includes: citing the official 2013 IEEE and ACM Guide to Computing Curricula as a source for our course curriculum, identifying specifically the current gaps in course curriculum, documenting how the gaps were covered by updating the curriculum, and comparing the reactions of students prior to the updated curriculum to the reactions of students after the curriculum is updated.

References

[1] G. K. Gupta, Computer literacy: essential in todays computer-centric world, SIGCSE Bull., vol. 38, no. 2, 2006, pp. 115119.

[2] ACM/IEEE-CS Joint Task Force for Computer Curricula. *ACM/IEEE Computing Curricula 2013: Ironman Draft (Version 1.0)*. http://ai.stanford.edu/users/sahami/CS2013//ironman-draft/cs2013-ironman-v1.0.pdf

[3] M. Guzdial and A. Forte. "Design Process for a Non-majors Computing Course," in *Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education*, St. Louis, MO, USA. Feb 2327, 2005, pp. 361-365.

[4] R. Tewari and F. Friedman. "A Framework for Incorporating Object-Oriented Software Engineering in the Undergraduate Curriculum," in *Computer Science Education*, vol. 4, no. 2, 1993, pp. 45-62.

[5] C. Dörge and C. Schulte. "What are information technologys key qualifications?," in *Proceedings of the Conference on Integrating Technology into Computer Science Education*, Madrid, Spain, Jun 30-Jul 2, 2008, pp. 296-300.

Should College Algebra be a Coerequisite for Computer Science 1? FECS: Position Paper

Christine F. Reilly and Emmett Tomai
Computer Science Department, University of Texas – Pan American
1201 W. University Dr., Edinburg, Texas, 78539, USA
Email: reillycf@utpa.edu, tomaie@utpa.edu

Abstract—*It has long been thought that mathematics background is a factor in a student's success in introductory computer science courses. At our university, College Algebra is a corequisite for Engineering Computer Science I. For some students, meeting this corequisite means that they have to delay taking Computer Science and Computer Engineering courses by one or more semesters until they have achieved the necessary mathematics preparation. We conducted an initial investigation into whether the College Algebra corequisite is actually beneficial, and found mixed results. This prompted us to embark on a larger study. In this paper we discuss the history behind mathematics prerequisites for introductory computer science and computer engineering courses, present the findings from our initial study, and describe our current larger study.*

Keywords: computer science, computer engineering, success, mathematics

1. Introduction

At our university, in order to enroll in Engineering Computer Science I (CS1), students must have concurrently enrolled in College Algebra or have placed into a higher level math course. The reasoning behind this corequisite is that College Algebra provides the problem solving abilities that are required for CS1. In this paper we present our ongoing investigation into whether the College Algebra corequisite is actually helpful.

We offer a Computer Science and a Computer Engineering major. Students from both majors must take the same sequence of introductory computer science courses. CS1 is actually the second course that our Computer Science and Computer Engineering majors take. The first course is Introduction to Computer Science or Introduction to Computer Engineering, courses that provide a survey of their field and introduce basic concepts. Following CS1, students from both majors continue the introductory computer science sequence with Computer Science II (CS2), then Algorithms and Data Structures.

The drawback to the College Algebra corequisite for CS1 is that many students enter our university with poor mathematics preparation from high school. Students who are placed at or below the College Algebra level in the mathematics sequence are not able to take CS1 and can fall behind in the major, requiring extra semesters for graduation. Additionally, students who entered the university intending to major in Computer Science or Computer Engineering may lose interest and switch majors if they are not able to take the major courses in their first couple of semesters at the university. Partially due to these reasons, we allow some students to take College Algebra and CS1 in the same semester.

The University of Texas – Pan American is located in the Rio Grande Valley of South Texas. The university as a whole is pursuing student retention initiatives in order to increase the percentage of students who graduate in six years, which now stands at 42%. Our examination of whether the College Algebra corequisite for CS1 fits into the university's initiative to increase the six-year graduation rate because we are trying to help students complete the major within the standard four-year undergraduate period.

In Section 2 we explore related work and further motivate our study. Section 3 describes our initial investigations into performance in CS1 based on a student's math background. We describe our current study in Section 4, including a description of the questionnaire we administered to the CS1 students in Spring 2013. Section 5 describes our ideas for expanding this study into a study of how students perform in later computer science and computer engineering courses based on their grade in CS1. Finally, we conclude the paper in Section 6.

2. Related Work

Since the early days of the Computer Science field, it has been thought that mathematics background has an impact on a student's success in introductory computer science courses. A number of studies were conducted in the early 1980's that examined the success of students based on mathematics preparation and other background factors. Many of these studies aimed to discover factors that predict success in a computer science major, and planned to use these factors as part of admissions decisions.

Konvalina et. al. conducted a study that compared students who withdrew from beginning computer science courses with those who did not [1]. They examined factors such as educational background, previous computer experience, mathematical ability, and potential for academic success in computer science. Their study was conducted with 382 students in an Introduction to Computer Science course that had one year of high school algebra as a prerequisite. They found that the students who did not withdraw from the course were older, had better performance in high school, had more previous computer science experience, and had more mathematics background that those students who withdrew from the course. Following this study, the researchers used the findings for advising and placing students. Their initial placement results were that the withdraw rate decreased from 40 percent to 23 percent.

Another study examined first–semester freshman who enrolled in an introductory computer science course, and followed these students through their first year at the university [2]. The authors state that successfully completing the first year of a computer science major is a good indicator of success in the major. At the beginning of their sophomore year, these students were classified into two groups: computer science and related majors, and other majors. The students in the computer science and related majors group were found to have higher SAT math and verbal scores, higher high school rank, and stronger high–school mathematics and science backgrounds.

Butcher and Muth [3] also examined a number of factors in order to find predictors of success in computer science courses. Overall, they found that high school GPA and ACT math scores were the best predictive factor. They also found that the higher level of high school math a student had completed, the more likely that student was to succeed in computer science.

A number of factors that might contribute to success in CS1 were examined in [4]. The study used a questionnaire to collect information from 130 students in a CS1 course. This information was categorized into previous programming experience, previous non-programming computer experience, attribution for success/failure, self–efficacy, comfort level, encouragement from others, work style preference, math background, and gender. These factors were compared with the midterm course grade in order to determine which factors may contribute to success in CS1. The authors found that comfort level in the computer science class was the best predictor of success in the course. A student's math background was the second most important factor. The authors conclude that CS1 professors should create a welcoming atmosphere in the class where students are encouraged to ask questions and seek help. They also recommend that advisors consider math background when advising students whether or not to take CS1.

Another university that faces the same challenge as we do of students entering the university with a weak math background found that these students leave the Computer Science major at a high rate [5]. They define a weak math background as placement in elementary or intermediate algebra when a student enters the university. This definition of weak math background is similar to our College Algebra corequisite. Their approach for increasing retention rates in the Computer Science major was to create a CS0 course that introduced programming concepts using a media–rich programming language called Scratch. The results of their study was that the at–risk students who took CS0 in the first semester of their freshman year had a higher retention rate than the not at–risk first semester freshmen who took CS1 in the same semester.

Much of the prior work examined high school transcripts and standardized test scores in order to gather information about a student's academic background. We are taking a simpler approach of looking at whether a student is enrolled in College Algebra, or has placed into or completed a higher math course. Using a prerequisite course as a indicator of whether a student is likely to succeed in CS1 may be less detailed than looking at a number of other factors. However, with course registration being computerized, we need a simple metric that can be enforced by the registration system when students are signing up for CS1.

Table 1
CS1 Pass Rate of Students Based on Concurrent Math

Concurrent Math	Num. of Students	Num. Passing CS1	%Passing CS1
College Algebra	14	9	64%
Higher Math	41	25	61%
No Math	54	22	41%
All Students	109	56	51%

3. Initial Investigation

In order to get some idea of how effective our College Algebra corequisite is, we performed an initial examination of how many students passed CS1 based on the math course they took in the same semester as CS1. The reason we focused on the concurrent math course is simply because it was relatively easy to get this data. It would have been more difficult to gather information about the highest math class each student in the class had passed. This means that we have no information about the math preparedness of students who were not concurrently taking a math course. As discussed below, more detailed studies comparing math courses taken and CS1 success are planned for the future.

The grades of 109 CS1 students from the Fall 2012 semester were reviewed. We broke these students into three groups: those who took College Algebra concurrently with CS1, those who took a math class higher than College Algebra concurrently with CS1, and those who were not taking any math class in Fall 2012. We calculated the percentage of the students in each group that passed CS1, along with the overall pass rate. A passing grade in CS1 is considered to be a C or better. These results are shown in Table 1.

We found that students who took College Algebra concurrently with CS1 had a slightly higher pass rate (64%) than those taking a higher math course (61%). The pass rate of students who were not taking a concurrent math course was 41%, and the overall pass rate was 51%. Presumably, the students not taking a concurrent math course had already passed or placed out of College Algebra.

This result raises the question of why students who concurrently take any math class are performing better in CS1 that those not taking a math class. Possible reasons include college preparedness, expectations, knowledge about the appropriate classes to register for each semester, avoidance of math, and bad advising. At this point, we are focusing on determining if the College Algebra corequisite is actually helpful, but we will keep these other factors in mind for future studies.

We plan to expand this study in the future by working with the university to gather a larger and more detailed data set. This will enable us to look at the pass rate of CS1 students based on the highest math course they have completed (or the class they have placed into). We will also expand this study across multiple semesters.

This initial study lead us to explore options for obtaining additional data about student's preparedness for CS1. We decided to gather data by asking students to complete a questionnaire, as described in the next section.

4. Current Study

We are currently engaged in a study involving the students who took CS1 in Spring 2013. We designed a questionnaire to gather information about CS1 students' math and programming background. This questionnaire was administered to the approximately 60 students in two sections of CS1 at the beginning of the Spring 2013 semester. Table 2 shows the list of questions on the questionnaire.

This questionnaire evolved from one that a number of our instructors were informally using to gather general information about the students in their courses. For our purposes, we added the questions about math and programming background. We also added Question 5 (how much do you enjoy math) because one of our instructors feels that enjoyment of math may be a predictor of success in CS1.

Questions 1 through 3 are asking for basic demographic information. We gather information about math background and enjoyment of math in Questions 4 and 5. We then get information about the student's computer science experience. The Introduction to Computer Science and Introduction to Computer Engineering courses provide a survey of the field and a short introduction to programming using the Lego Mindstorms platform. Ideally, these introductory courses will be completed prior to a student taking CS1. However, because of scheduling issues or a student's prior computer science and mathematics experience, some students are allowed to take Introduction to Computer Science/Engineering and CS1 concurrently. Because our CS1 course has a high fail/drop rate, as do many CS1 courses around the world, Question 7 asks whether they have previously taken CS1. We then inquire about previous programming experience in Question 8. The last set of questions were on the original questionnaire that had been used by a number of instructors for many semesters. We ask about their computer, email, internet, and learning software use in Questions 9, 10, 11, and 12. Then Questions 13, 14 and 15 gather information about how busy a student is outside of this class and how much time she expects to dedicate to this class. Questions 16 and 17 are open response questions where a student can express what they hope to gain from CS1 and what difficulties they expect they may face.

We are currently in the process of comparing the questionnaire responses with each student's grade and determining if there is any correlation between math or programming background and the grade received in CS1.

Table 2
CS1 Questionnaire

1. Identifying information (name, ID number)
2. Major and minor
3. Classification (freshman, sophomore, junior, senior, other)
4. Current math course, or highest math course taken
5. I enjoy math... (select on scale of 1 to 5)
6. Have you taken Introduction to Computer Science/Engineering? If so, what grade did you receive?
7. Have you taken CS1 before? If so, what grade did you receive?
8. Do you have any programming experience? If so, what language(s)?
9. Do you have a computer at home? If so, is it a laptop or desktop?
10. How frequently do you check your university email?
11. How frequently do you use the internet?
12. How frequently do you check Blackboard (the university's online learning system)?
13. How many hours are you enrolled in this semester?
14. How many hours per week are you working?
15. How many hours do you expect to dedicate to this class outside of class time?
16. What do you hope to gain from taking this class?
17. Are there any difficulties you anticipate that will prevent you from successfully completing this course?

5. Future Directions

In addition to continuing the studies described in Sections 3 and 4, we plan to look into additional success factors in what we call the introductory pipeline in Computer Science. This pipeline includes the CS1, CS2, and Algorithms and Data Structures courses. We would like to improve the pass rate in all of these courses.

Along with other instructors who teach these three courses, we have anecdotally observed that earning a grade of C in one of these courses may predict that a student is likely to struggle in the next pipeline course. This may indicate that students who earn a C in a particular pipeline course are not gaining the knowledge needed to succeed in the next course. The grade of C may also be due to poor study habits or other factors that persist as the student progresses through the pipeline.

6. Conclusions

From the early days of the Computer Science field, there has been interest in discovering what factors indicate that a student is likely to succeed in a Computer Science major. Some educators are interested in these factors in order to assist in admissions decisions. We are more interested in discovering how we can better prepare our students and increase the success rate of students in our Computer Science major.

Mathematics background has been identified by many studies to be an important factor that predicts success in Computer Science. Our CS1 course requires that students are currently taking College Algebra, have completed College Algebra, or have placed into a higher level math course. Because a number of our students place into a lower math course than College Algebra upon entering the university, these students are held back from taking Computer Science or Computer Engineering courses until they have completed one or more semesters of math. We are currently studying whether the College Algebra corequisite actually helps students succeed in CS1. This study will allow us to set the CS1 prerequisites/corequisites in a way that includes as many students as possible in the course while ensuring that those students are adequately prepared for the course.

References

[1] J. Konvalina, S. A. Wileman, and L. J. Stephens, “Math proficiency: A key to success for computer science students,” *Communications of the ACM*, vol. 26, no. 5, pp. 377–382, May 1983.

[2] P. F. Campbell and G. P. McCabe, “Predicting the success of freshmen in a computer science major,” *Communications of the ACM*, vol. 27, no. 11, pp. 1108–1113, Nov. 1984.

[3] D. F. Butcher and W. A. Muth, “Predicting performance in an introductory computer science course,” *Communications of the ACM*, vol. 28, no. 3, pp. 263–268, Mar. 1985.

[4] B. C. Wilson and S. Shrock, “Contributing to success in an introductory computer science course: A study of twelve factors,” in *SIGCSE 2001*, Charlotte, North Carolina, USA, 2001.

[5] M. Rizvi and T. Humphries, “A scratch–based cs0 course for at–risk computer science majors,” in *42nd ASEE/IEEE Frontiers in Education Conference*, Seattle, Washington, USA, 2012.