# Software Architecture for Intelligent e-Learning Decision Support System

**M.C. Mihăescu[1], D.D. Burdescu[1], C.M. Ionașcu[2], and B. Logofătu[3]**
[1]Software Engineering Department, University of Craiova, Romania
[2]Statistics Department, University of Craiova, Romania
3CREDIS Department, University of Bucharest, Romania

**Abstract -** *The intelligent character of a decision support system is one of the most appreciated issues in current e-Learning trends. This paper presents a software architecture design for e-Learning domain. The design has as main goal usage of reusable components such that the application may be easily configured. The proposed software architecture ensures characteristics such as prediction, optimization and adaptation. These aims should be achievable by custom configuration of the application representing the container in which the components are deployed.*

**Keywords:** eLearning; decision support system; software architecture

## 1 Introduction

Domain specific software architectures (DSAA) [1,2,3] is a software engineering methodology that makes sure that a custom software architecture is obtained for a specific class of applications. In order to create a DSAA a reference architecture is needed. This architecture must describe the general data flow of the employed business logic. Within the reference architecture there are defined necessary components that will make up the system and is not dependent of the application domain. The defined components are domain specific and that is why they need a domain expert to define and interconnect them. The application configuration procedure is the step in which components are interconnected and the logical and functional dependencies are described. The container need to offer the possibility of setting all necessary dependencies.

Once the software architecture has been created the intelligent character needs to be offered. This means the software architecture needs to become a complex environment which works with goals set up by users. The system will manages performed actions by users and also user's priorities and deadlines. The main goal of the system is to emulate user's behavior in a predictable manner. In this way, suggestions may be offered to users, thus being obtained an intelligent decision support system.

The proposed architecture is described by a general reference specification where computational requirements for the intelligent decision support system are presented. The components that make up the system are presented in an abstract manner such that they may become reusable components in a domain specific library of implemented components. The components that were previously described according with a certain domain are instantiated and managed by an application configuration tool. This tool makes possible to be defined application-specific requirements, custom data sources, custom knowledge flows, custom business logic controllers or even custom data/results viewers.

There are five types of decision support systems: communication driven [4], data driven, document driven, knowledge driven [5], and model driven[6]. The presented approach from this paper is a hybrid approach between data and knowledge driven approaches. Data driven approach is due to the fact that time series data representing performed actions by learners are taken into consideration each time data model is being created or a decision is derived. The knowledge driven approach is employed because the data management is accomplished using data mining/machine learning algorithms. Since the software architecture design is abstract virtually any machine learning algorithms (e.g. clustering, classification, etc.) may be accommodated.

The main domains in which intelligent systems may be found are clinical decision support systems for medical diagnosis [7,8], business and management [9], agricultural production [10]. In e-Learning, building decision support systems has been associated with educational data mining. Educational data mining (EDM) is an emergent discipline concerned with developing methods for exploring the unique types of data that come from the educational context [11]. In short, EDM is the application of the data mining techniques in the area of education, with the aim of obtaining a better comprehension of the students' learning process and of how they participate in it, in order to improve the quality of the educational system.

## 2 Description of the architecture

The proposed reference architecture is a heterogeneous mixture of common architectural styles [12]. It is divided hierarchically into layers for different sets of computational tasks. In our approach there are defined two layers: a physical data layer and a knowledge representation layer. The knowledge representation level is a mirror of the physical data layer where knowledge regarding data has been extracted and custom business logic may be found.

The physical output of the decision builder module is represented by the advice. It may be represented by a chapter that needs more attention from the learner, a set of test questions that need to be answered by learner, a set o concepts that need more attention from the learner or a suggested learning path. A learning path may represent a timetable with activities (study of materials and/or test questions) that is advisable to be followed in order to meet constraints set by administrator, course manager and learner himself.

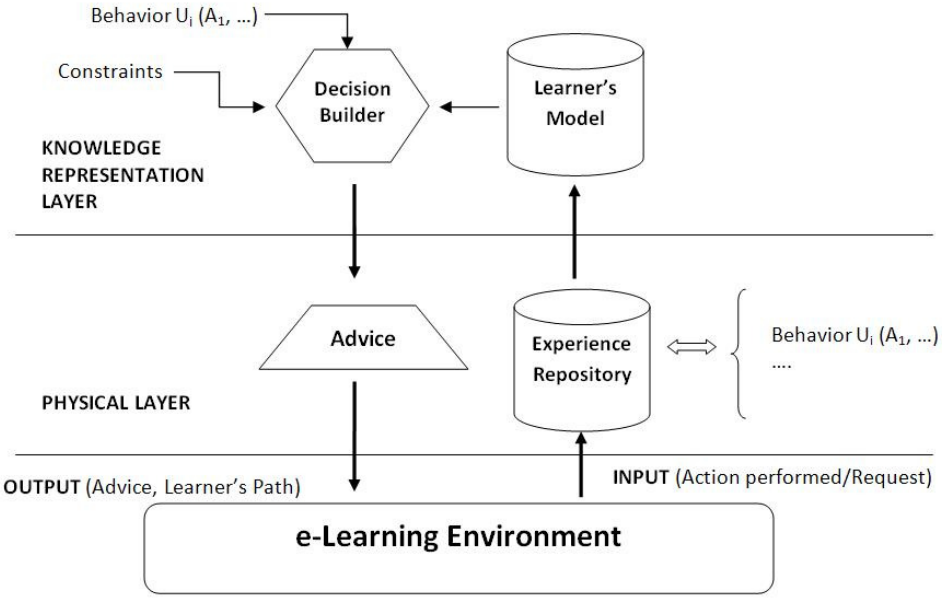


Figure 1. Reference Architecture

Each layer is organized into components that are custom designed for fulfilling certain functions. The physical layer mainly consists of the experience repository offered by the e-Learning environment. It is made of behaviors performed by users presented as a series of performed actions.

The Experience Repository from the Physical Layer takes shape of a learner's model in the Knowledge Representation Layer. The Learner's Model is highly dependent on the employed data mining algorithm used for data representation.

The Decision Builder represents the main business logic component from the knowledge representation layer. It takes as input the current Learner's Model, the current state of the learner (his behavior), a set of constraints from all involved parties and the learner's request. The current state of the learner is represented by all the actions performed by the learner up to the moment when he requests advice. The constraints are of different types. There are constraints set by the manager of the e-Learning environment, by the course manager and by the student.

The core of the reference architecture is represented by the learner's model. The learner's model may be represented by an embedded machine learning algorithm. The process of embedding a standard and abstract machine learning algorithm into an e-Learning environment represents the main mechanism for building intelligent decision support systems.

The main outcomes that may be obtained by machine learning integration are:

***Classification*** - The goal of classification is to build a set of models that can correctly predict the class of the different objects. The input to these methods is a set of objects (i.e., training data), the classes which these objects belong to (i.e., dependent variables), and a set of variables describing different characteristics of the objects (i.e., independent variables). Once such a predictive model is built, it can be used to predict the class of the objects for which class information is not known a priori. The key advantage of supervised learning methods over unsupervised methods (for example, clustering) is that by having an explicit knowledge of the classes the different objects belong to, these algorithms can perform an effective feature selection if that leads to

better prediction accuracy. Classification arranges the data into predefined groups. For example, an e-learning environment might attempt to classify an e-Learning resource (e.g. document, exercise, etc..) as required or not. Common algorithms include decision tree learning, nearest neighbor, naive Bayesian classification and neural networks.

***Clustering*** - Is like classification but the groups are not predefined, so the algorithm will try to group similar items together. Clustering is considered the most important unsupervised learning problem; so, as every other problem of this kind, it deals with finding a structure in a collection of unlabeled data. A loose definition of clustering could be "the process of organizing objects into groups whose members are similar in some way". A cluster is therefore a collection of objects which are "similar" between them and are "dissimilar" to the objects belonging to other clusters.

***Regression*** – Regression attempts to find a function which models the data with the least error. Regression is the oldest and most well-known statistical technique that the data mining community utilizes. Basically, regression takes a numerical dataset and develops a mathematical formula that fits the data. When you're ready to use the results to predict future behavior, you simply take your new data, plug it into the developed formula and you've got a prediction! The major limitation of this technique is that it only works well with continuous quantitative data (like weight, speed or age). If you're working with categorical data where order is not significant (like color, name or gender) you're better off choosing another technique.

# 3 Application Configuration

Application configuration is concerned with custom setting up of the defined components. The experience repository has the physical shape of an XML file that is regularly updated with actions received from the e-Learning environment. Although the data source is always an e-Learning environment there has been defined a general mechanism for specifying the way action data is structured. Firstly, the parameter set that defines the activity of a learner is described. For each parameter there is defined a type (usually integer of float) such that each instance (learner) is described by a set of values. Secondly, there is defined the effective data source which may be a txt file, an xml file or even a database.

The configuration of the Learner's Model regards the employed machine learning algorithms. A set of machine learning algorithms (e.g. clustering, decision trees, classification rules) may be used for extracting knowledge from the experience repository. Besides setting up the used algorithm there are also specified what parameters and in what way they are used. Some of the constraints already set by involved parties may be used in building the learner's model. The constraints are related with the quality of the obtained learner's model. The quality is of critical importance since the decision builder component relies on it.

The Decision Builder represents the most important component within the architecture. It represents the main controller of the business logic. It has input from three sources: the learner's model, the learner's behavior for which advice is obtained and the constraints from system, course manager or learner himself. The output of this module has the a representation in physical layer and is abstractly called advice. The advice may be represented by a list of chapters that need more study, a list of concepts that need more study or even quiz questions that learner is invited to answer. A more elaborate advice may be obtained in the form of a recommended learner's path. This is represented by a set of activities that need to be performed is a certain order and with a certain amount of time assigned for each specific activity.

The application configuration phase regards building a knowledge flow represented by all the necessary steps until results are obtained. The main steps are:

***Define Data Sources and Data Sinks***. Data sources are used to indicate where data is coming from. It may be represented by various file types and sources and may be configurable for a custom file name of data source or even dataset.

***Define and Setup Filters***. Filters are used to preprocess data prior to classification or learning. There may be set up both supervised and unsupervised filters and they may be configurable depending on filter type.

**Define Modeling Technique**. A classification, clustering or association algorithm may be used. The parameters are configurable depending on classification algorithm or clustering algorithm.

***Use an Evaluation Methodology***. This procedure is used to configure both inputs to and outputs from algorithms. There may be used various algorithm performance evaluators. This step is of great importance because is measures the confidence in the obtained recommendations.

**Use a Visualization Procedure**. This is the final step. It is used to visually display outputs. It supports performance and summaries of the results.

# 4 Sample Setup

The setup of such a system requires certain specific setup phases. The first setup phase regards the client e-Learning platform and the way context and activity data are represented. These two components are regarded as Context Representation and Activity Data Representation. The context representation regards all data that resides within the client e-Learning environment.

Table 1. Sample context representation

| Context variable | Description |
|---|---|
| nModules | Number of modules managed by the e-Learning platform |
| disciplines | The disciplines studied at each module |
| chapters | The chapters associated to each discipline |
| concepts | The concepts associated to each chapter |
| quizzes | The quizzes associated with each concept |

All this data is gathered into an XML file and is loaded by the decision support system. Table 1 presents the main fields in the context representation.

The second phase regards the activity data representation. The activity data regards data that may be obtained from the actions that were performed by learners.

Table 2. Sample activity data representation

| Context variable | Description |
|---|---|
| positiveCount | The number of correctly answered questions. |
| correctPercent | The percentage of correctly answered questions from the total number of questions. |
| totalTries | The total number of tries (answered questions) |
| avgTries | Medium number of tries per question |
| avgQuestionTime | On average, how long (in minutes) it takes for a student to answer a question |
| totalTime | Total time spent on testing |

All this data is gathered into an XML file and is generated by the decision support system from the available data. Table 2 presents the main fields in the activity data representation. When a learner requests advice the decision support system computes the values for each field from table 2. All the fields are regarded as features (or parameters) for the learner that needs advice. The values are computed at runtime when the request for advice command is issued by the learner. These values represent a digest of the activity performed by the learner.

Another setup phase regards the constraints representation. There are three setup protocols, one for each type of user involved in the platform: administrator, course manager and learner. From the learner's point of view, one constraint may regard the time that is needed for study. The student may want to minimize the time in which a certain level of knowledge is reached. This is accomplished by specifying a desired grade or even a certain percent of knowledge coverage for a concept, chapter or even discipline. A more restrictive constraint may be set by a learner regarding the certainty with which a certain goal is to be achieved. The learner my need one hundred percent certainty that he passes the exam or he may need a lower certainty level. On the other hand, the learner may specify a certain degree of certainty for the goal of being the best. Other custom situations may occur when learners have specific goals.

From the course manager point of view, the constraints are specified in a different way. These constraints regard the policy of the course manager regarding the desired amount of knowledge learners need to have. One possible setup of constraints is to enforce a certain accuracy level for learner's classification. The course manager is offered the environment and the needed tools and mechanisms with which he can scale up or down the metrics associated the knowledge level measurements.

Let us suppose a course manager decides that the general results are too good and thus future learners need to have harder tests. This may be accomplished by modifying (increasing) the value of the parameter that represents the knowledge weight of a test or exam. Thus, harder questions will be part of the test or exam. During study period the recommender system will provide tests that have higher difficulty level and thus the advice regarding the time spent and the resources (concepts, chapters, quizzes) that need more study will be dynamically and automatically adapted.

Another setup phase regards the way learner's model is built. The outcome of this step may be regarded as a custom built knowledge flow that integrates different machine learning algorithms. A first step may be represented by a k-means clustering which may be used for outlier detection. The instances that are far way from the obtained centroids are pruned and thus more representative data is obtained.

A second step in the knowledge flow mat be represented by a classification tier. Different classifications may be obtained using Decision Tree inductions mechanism or a Bayesian classifier. Each classifier may be used of parts of the dataset obtained after clustering or on whole dataset after pruning the outliers.

The final step of the setup regards the Decision Builder. At this level there are specified the parameters that configure the knowledge flow regarding the constraints that were already setup and the current behavior of the learner that requested advice. The setup is performed by customizing certain values in xml type properties files. The properties files are the ones that generate the custom behavior of the decision support system.

The manager of the system needs to be a data analyst with good knowledge of e-Learning, data analysis and information retrieval techniques. The configuration of the knowledge flow and of all parameters that trigger the functionality of the Decision Builder module will have direct

influence over the quality of the obtained advice and thus of the entire system.

# 5 Conclusions and Future Works

This paper presents a domain specific software architecture that can be used for building an intelligent decision support system.

The architecture is built on two levels. One is a physical level where all necessary data representation is accomplished. This regards e-Learning platform's context data and activity data representation as input. The physical representation of the output is represented by the obtained advice which may take different shapes.

The knowledge representation level is composed of two main building blocks: a learner's model and a decision builder. These two modules describe a knowledge flow whose functionality is managed by constraints.

The software architecture is scalable in the way that machine learning algorithms may be easily added within the knowledge flow. The data sources may be easily integrated by specifying the location and the structure of the input data. The knowledge flow is designed to accept parameters (constraints) such that the different learning goals may be achieved.

As future works, there is a need to define a procedure for measuring the quality of the obtained advice. Further experiments need to be accomplished with different context representations and different constraints. Other machine-learning/data mining algorithms need to be integrated along with their corresponding constraints. The experience repository needs to have implemented more elaborate mechanisms for data management since it store huge amounts of data.

# 6 Acknowledgement

This work was supported by the strategic grant POSDRU/89/1.5/S/61968, Project ID61968 (2009), co-financed by the European Social Fund within the Sectorial Operational Program Human Resources Development 2007 – 2013.

# 7 References

[1] E. Mettala. "Domain specific software architectures," in ISTO Software Technol. Community Meet., June 1990.

[2] W. Tracz, L. Coglianese, and P. Young, "A domain-specific software architecture engineering process outline," in SIGSOFT Software Eng. Notes, vol. 18, pp. 4049, 1993.

[3] I. D. Tommelein, B. Hayes-Roth, and R. E. Levitt, "Altering the SightPlan knowledge-based systems," J. Artificial Intelligence in Eng., Automation, and Manufacturing, vol. 6, pp. 19-37, 1992.

[4] Stanhope, P., "Get in the Groove: building tools and peer-to-peer solutions with the Groove platform", New York, Hungry MindsK, 2002.

[5] Power, D. J., "Decision support systems: concepts and resources for managers", Westport, Conn., Quorum Books, 2002.

[6] Gachet, A., "Building Model-Driven Decision Support Systems with Dicodess", Zurich, VDF, 2004.

[7] Barnett GO, Cimino JJ, Hupp JA, Hoffer EP. DXplain. An evolving diagnostic decision-support system. JAMA. 1987 Jul 3;258(1):67-74.

[8] Glinkowski W, Kornacki M, Ambroziak M, Górecki A, "Relative optical density image analysis (RODIA) of digitized radiograms for tibial fracture healing monitoring", Int Congr Ser. 2004;1268:1292.

[9] Eckerson, Wayne W, "Performance Dashboards: Measuring, Monitoring, and Managing Your Business", John Wiley & Sons, 2006.

[10] Stephens, W. and Middleton, T., "Why has the uptake of Decision Support Systems been so poor?", In: Crop-soil simulation models in developing countries. 129-148 (Eds R.B. Matthews and William Stephens). Wallingford:CABI, 2002.

[11] Romero, C. and Ventura, S. Educational Data Mining: A Survey from 1995 to 2005. Expert Systems with Applications, 33(1), 135-146, 2007.

[12] D. Garlan and M. Shaw, "An introduction to software architecture," in Advances in Sofrware Engineering and Knowledge Engineering. New York World Scientific, 1993, vol. 1.