Automatic Generation of a Grounding Framework for Information Extraction

Anthony Stirtzinger, Steve Gorczyca, Joshua Powers and Matthew Dyke
Securboration Inc., Melbourne, FL, USA

Abstract - *The research described in this paper presents an approach to automatically generating a framework, called the Grounding Framework, which is applicable to any existing domain ontology and positions that ontology to be used as a model for information extraction. This approach allows a domain ontology to accurately reflect the domain while not compromising its structure and style in efforts to prepare it for use in information extraction. It also promotes the reuse and extension of existing ontologies for the purpose of information extraction without costly manual intervention. The research in this paper builds on existing research we have conducted for our Ontology Generation and Evolution Processor (OGEP) and Lexicalizing an Ontology.*

Keywords: Information Extraction, Knowledge Extraction, Ontology, Semantic, Framework

1 Introduction

Model-driven strategies for artificial intelligence have been successful in several problem spaces[1][2][3]. Common in these strategies is the idea that a top down approach to problem solving can lead to solutions gained from the use of common frameworks as opposed to implementation-specific solutions. These approaches leverage mechanisms that are guided (or driven) by an accurate representation of the environment (both problem space and solution space) in the form of models.

Several information extraction efforts have used ontological domain modeling to direct or improve extraction results. Most of these efforts require a combination of natural language processing (NLP), ontology development, and algorithms to match NLP discoveries with ontological concepts. Ontology plays the role of model in these model-driven applications. Generally there are two approaches for dealing with the ontology component of these solutions.

One approach is to generate the ontology either automatically or semi-automatically as in [4]. Strategies like these are tied closely to the NLP component, resulting in ontologies derived from the NLP mentions. Another approach is to build the ontology manually and then leverage the ontology during information extraction[5][6]. This type of approach provides more flexible separation from the NLP component and allows for domain declarations external to the particulars of a parsed corpus. However, there is a time and effort cost associated with building the ontology. Hybrid approaches[7] may start with a minimally sized ontology, then evolve that ontology from the NLP findings.

While the first approach greatly reduces the time required in the manual process of ontology building, the completeness and accuracy of the ontology remains in question. It is also difficult, if not impossible, to leverage assets of existing ontologies in combination with the generated ontology. The second approach requires the burden of manual construction, but usually results in a more accurate domain ontology. However, conventional ontology construction does not necessarily lend itself to the purpose of information extraction.

The research described in this paper presents an approach to automatically generating a framework, called the Grounding Framework, which is applicable to any existing domain ontology and positions that ontology to be used as a model for information extraction. This “automatic generation” approach allows the domain ontology to accurately reflect the target subject matter while not compromising efforts to prepare it for use in information extraction. It also promotes the reuse and extension of existing ontologies for the purpose of information extraction without costly manual intervention.

2 Motivation

As we have applied our Ontology Generation and Evolution Processor (OGEP) [8] more to information extraction as opposed to strictly ontology generation/evolution, critical needs have been uncovered. OGEP requires our Semantic Grounding Mechanism (SGM) constructs to exist in order to process NLP output and align it with ontological concepts. We have always created SGM constructs manually, which is time consuming, often taking weeks to complete. When a domain ontology already existed, but was not in the SGM structure, a manual alignment process was required. Therefore, to reduce the time of generating the SGM constructs and also allowing widespread reuse of existing ontologies, we initiated the effort described in this paper to automatically generate the Grounding Framework. This paper describes both the constructs and approach for generating the Grounding Framework from an existing domain ontology.

Since the focus of this research is based on the automatic generation of an SGM-compatible framework, our results will be shown in the context of successfully creating the Grounding Framework from an existing ontology. We present the results of a simple information extraction in this paper to demonstrate that the Grounding Framework can work with our existing SGM information extraction services.

3 Related Work

The Bank Ontology used throughout this paper was built using an upper level ontology, specifically the Basic Formal Ontology (BFO)[9]. BFO is an ontology that defines general universal types which can then be extended through sub-typing to define more specialized universals which comprise the content of a domain of interest. BFO is differentiated from other upper level ontologies such as DOLCE and SUMO by being narrowly focused on supporting the task of building domain ontologies for areas of scientific research. Along with BFO, the Bank Ontology made use of the Relation Ontology (RO)[10]. RO is similar to BFO in scale and purpose, but where BFO defines upper level objects and events, RO defines a set of core relations between objects, events and objects and events. In its Web Ontology Language (OWL) serialization, these relations are expressed using object properties. Finally, the RO_BFO_Bridge[11] was used. RO_BFO_Bridge is a serialization of the union of BFO and RO (in either OBO or OWL format) created by Chris Mungall.

The research described in this paper is dependent on the lexicalization of the domain ontology. [12] describes an automated approach for generating mapping candidates between WordNet synsets and target ontology objects. We use this approach to create mappings for our domain ontology to prepare it for automatically generating the Grounding Framework.

The WordNet lexical database[13] provides sense definitions for words (categorized into Noun, Verb, Adjective, and Adverb), called Synsets, along with a natural language descriptions, synonyms and relations to other senses. The sense relationships are categorized into hypernyms, hyponyms, holonyms and troponyms along others. These relationships allow checking whether a word belongs to a particular group (e.g. *dentist* is a *person*).

4 Approach

Our approach to automatically generating a grounding framework that provides an alignment mechanism between information extraction technologies and ontology is based on the following pre-exiting components: 1) a domain model representing the target for which to align the information; and 2) a lexical basis to relate the domain concepts to language.

The domain model is specified in an ontology language. For the tests performed in this research, the ontologies were expressed using Web Ontology Language (OWL). No other requirements are enforced on the domain model, however the quality of the lexicalization process (described later) increases when naming and/or label annotation of ontological components (classes and object properties) closely parallels domain definitions in the language that will be used for information extraction. During our research we tested with two basic ontology styles: 1) a domain ontology that did not make use of an upper level ontology; and 2) a domain ontology that made use of upper-mid-level ontology.

Our research made use of an automated process for generating mapping candidates between WordNet synsets and target ontology objects[12]. This research is intended to build upon [12] so that we may subsequently build a grounding framework for information extraction.

4.1 Grounding Framework Namespace

The Grounding Framework namespace is specified in OWL and is used as a base framework for representing both the domain ontology structures and the WordNet annotations in a way that they can be used to align unstructured text. Our original OGEP development used the SGM_0 ontology to define semantic grounding constructs that were recognized by the Semantic Grounding Mechanism (SGM) processor. The Grounding Framework is an evolution of the SGM_0 ontology that is more compatible with auto-generation. Both the SGM_0 and Grounding Framework ontologies are designed primarily to accomplish two functions.

First, SGM provides a way to extend the OWL Restrictions in the domain ontology such that they can be used to reason with partial evidence. This is done using the node-dependency-capability relationships contained with SGM as documented in [14].

Second, SGM is used to correlate the concepts in the domain ontology to the English language. This is accomplished through the use of the WordNet annotations as previously described along with verb-noun-qualifier attributions available within the SGM construct[7][8][14].

4.1.1 SGM Pattern in the Grounding Framework

Figure 1 shows the basic Node, Dependency, Capability pattern that supports the original SGM pattern. [7][8]and[14] describe more detail about the SGM pattern. The essence of the pattern is that a domain concept can be defined two ways. First, a domain concept can be defined by what it produces. We refer to this as a Capability. For example, the concept “exploded bomb” may have the capabilities of “smoke”, “fire” and “shrapnel”. Second, a domain concept can be defined by what must exist in order for the concept itself to exist. We refer to this as a Dependency. For example, the concept of “bicycle” may have dependencies of “wheel”, “seat”, “handle bar”, “frame”, and “pedal”. The SGM pattern provides a mechanism for defining the relationship of Domain Concept, Dependencies and Capabilities.

In the Grounding Framework ontology this pattern is realized as a gf:Node class and associated annotations. The three annotations are gf:groundsTo, gf:providesCapability and gf:requiresDependency. The values of the annotation are fully qualified ontology classes. Descriptions of the classes and annotations comprising this pattern follow.

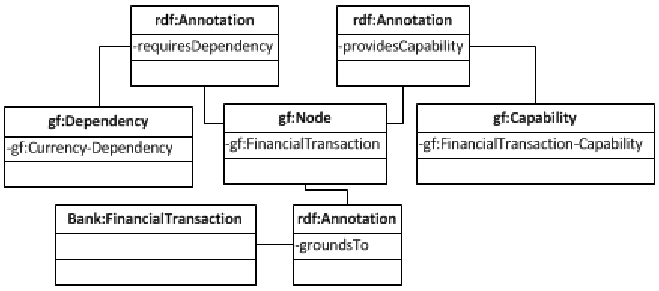


Figure 1 Basic SGM Pattern

gf:Node - Nodes are the ontology representations of the classes in the target ontology. Each node class will have a one to one correspondence to a class in the target ontology, and the class will be annotated with the 'groundsTo' annotation that points to the target class. Nodes require one or more Dependency and provide one or more Capability.

gf:groundsTo - The subject of a groundsTo rdf:annotation must be a gf:Node subclass. The value of the annotation must be the fully qualified name of the domain class that the gf:Node subclass grounds to, meaning that any individual of the gf:Node subclass is believed to be an instance of the domain class with some confidence level.

gf:Capability - Capabilities define the domain concepts that are produced (or left behind) if the targeted domain concept exists or existed.

gf:providesCapabaility – Annotation that is used to relate a gf:Node to a gf:Capability. This is accomplished by attaching the annotation to gf:Node with the value of the annotation being the fully qualified name of the gf:Capability.

gf:ConceptDependency - Dependencies indicate pre-requisites for a node; domain concepts that must exist in order for the target domain concept to exist. Dependencies are generally derived from the domain ontology owl:Restrictions on the target domain concept.

gf:requiresDependency - Annotation that is used to relate a gf:Node to a gf:Dependency. This is accomplished by attaching the annotation to gf:Node with the value of the annotation being the fully qualified name of the gf:Dependency.

4.1.1.1 Subclassing in Grounding Framework

The Grounding Framework ontology has defined subclasses for gf:Node, gf:Dependency, and gf:Capability. This subclassing parallels logic in the SGM ontology and the reasoning algorithms that rely on the SGM ontology. Subclasses for gf:Node include gf:Actor, gf:Concept and gf:Physical. Subclasses for gf:Dependency and gf:Capability mimic this subclassing pattern with gf:ActorDependency, gf:ConceptDependency, gf:PhysicalDependency, gf:ActorCapability, gf:ConceptCapability, and gf:PhysicalCapability.

This subclassing supports the ability to classify domain concepts into the three subclass categories to simulate a very simple upper level ontology. The research described in this paper classified all domain concepts as gf:Concept for purposes of simplification.

4.1.2 NLP Annotations

The Grounding Framework is used to support information extraction. During this process the Grounding Framework is used by the consumer of NLP output. Therefore, the Grounding Framework includes provisions for storing the NLP output. Storage of NLP output insures traceability between the information source and the domain ontology that is the subject of the extraction effort. Figure 2shows the ontology structures used to store the NLP output followed by a brief description of each.

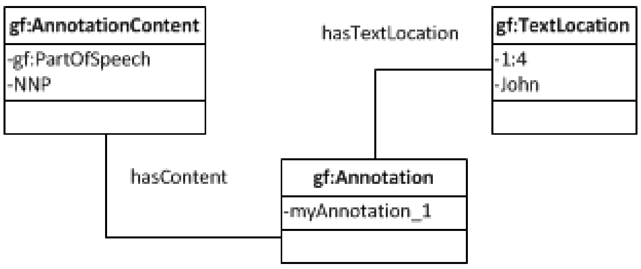


Figure 2 NLP Output in Grounding Framework

gf:Annotation - An ontology class that is used to store the information extracted from an NLP annotation (e.g. PartOfSpeech, Entity, etc.).

gf:TextLocation – An ontology class that is used to store the text location information extracted from an NLP annotation (e.g. PartOfSpeech, Entity, etc.). This information includes source document name and offset information of the NLP annotation.

gf:AnnotationContent – An ontology class that is used to store NLP annotation type-specific information. The Grounding Framework currently supports the following subclasses of gf:AnnotationContent: DTContent, FacilityContent, GPEContent, NamedEntity, OrganizationContent, PartOfSpeech, PersonContent, PosContent, StanfordDependency and WordNetContent.

4.1.3 WordNet Annotations in Grounding Framework

The Grounding Framework also contains some custom rdf:Annotations. These are used to provide the linkages for gf:Nodes (Dependencies, Capabilities) and the association of both domain ontology concepts and WordNet lexical groundings. For this research, the decision was made to represent this critical SGM information as rdf:Annotations as opposed to ontology classes or individuals. While this may potentially limit the ability to reason across this information using DL Implementation Group (DIG) reasoners, this strategy was chosen to encourage a loosely coupled relationship with regards to the original domain ontology structures. Following is a short description of each of the rdf:Annotations.

gf:wordnet3.0SenseKey - The subject of a wordnet3.0SenseKey annotation is associated with the SenseKey that is the value of the annotation.

gf:wordnet3.0SynSet - The subject of a wordnet3.0SynSet annotation is associated with the SynSet that is the value of the annotation.

gf:requiresDependency - Annotated on subclasses of gf:Node to indicate that all instances of that node must have a dependency of the class named in the annotation value.

gf:matchesDependencyClass - Annotated on a subclass of gf:Capability to indicate the dependency class that instances of the annotated class may fill.

gf:matchesCapabilityClass - Annotated on a subclass of gf:Dependency to indicate the capability class that instances of the annotated class may be filled by.

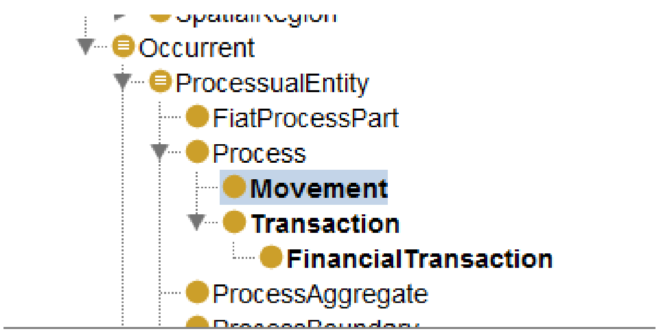
The following sections illustrate examples of instantiations of Grounding Frameworks using a small ontology that we refer to as the Bank Ontology.

4.2 Grounding Framework Instantiation

We created the Bank ontology to test and validate the algorithms that automatically generate an instance of the Grounding Framework from the prerequisites described in Section 4. This ontology will be used to show examples of a generated Grounding Framework in the following sections.

4.2.1 Bank Ontology to Grounding

Bank Ontology is a small ontology that contains a few concepts about the banking domain. These concepts are connected to the BFO and RO upper ontologies via subClassing. Figure 3 shows a segment of the Bank Ontology classes with domain-specific classes in bold and Upper/Mid-Level ontology classes in normal font.



Occurrent
ProcessualEntity
FiatProcessPart
Process
Movement
Transaction
FinancialTransaction
ProcessAggregate

Figure 3 Bank Ontology Classes

When creating the Grounding Framework, we reflect the class hierarchy of the domain ontology via gf:ConceptNode classes and the owl:subClassOf property between them. Maintaining the domain ontology's class hierarchy inside the Grounding Framework allows for two significant capabilities. First, the inheritance relationships defined in RDF can be reasoned about using DIGs reasoners. Second, during the information extraction process of matching NLP discoveries to Grounding Framework concepts, the traversal of hierarchical relationships provides a convenient mechanism for evaluating hypernym and hyponym hierarchies in WordNet. Figure 4 shows as sample of the domain classes instantiated as gf:Concept nodes in the Grounding Framework.

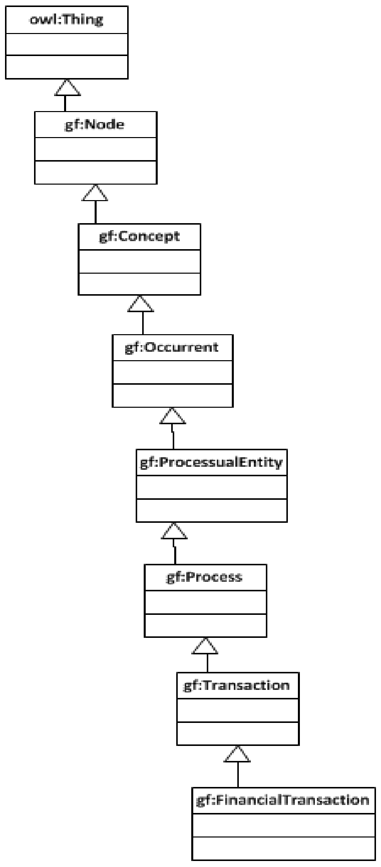


Figure 4 Bank Ontology Classes as gf:Concept Nodes

4.2.2 Bank Ontology Restrictions to Grounding

The Grounding Framework creation process also preserves the owl:Restrictions associated with each class. However, this information is not represented as an owl:Restriction, rather it is accomplished through the structures that exist within the Grounding Framework to support the SGM concepts. These structures include: 1) a set of custom annotations that are associated with each gf:Node type class; and 2) the use of gf:Dependency and gf:Capability class types.

The reason that owl:Restrictions are not used directly in the Grounding Framework is that we want to provide the ability to partially reason about the existence of domain entities. For example, in the Bank Ontology domain ontology, restrictions on a FinancialTransaction require the existence of a Transaction along with a has_participant relationship to some Currency. Unless both of these conditions are satisfied, the reasoner will not assert the existence of a FinancialTransaction. When performing information extraction, it is often beneficial to make partial assertions, or come to conclusions using partial evidence and associate a confidence or probability value to that conclusion. In the case where a transaction has been detected in the context of information extraction, but no currency is currently available from the corpus, there may be value in asserting the “possible” existence of a FinancialTransaction to some degree of certainty that is less than 100%. Figure 5 shows the domain restrictions for the FinancialTransaction class.

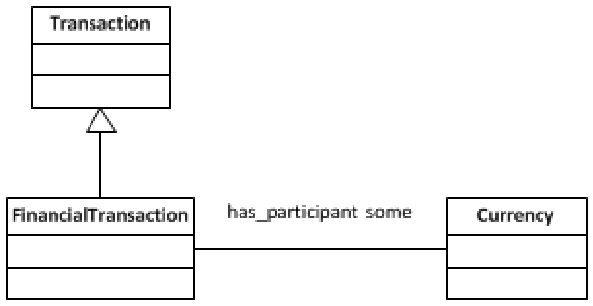


Figure 5 FinancialTransaction Restrictions

The custom annotations that are associated with each of the gf:Node class types to preserve the owl:Restriction declarations. These annotations consist of gf:groundsTo, gf:providesCapability, and gf:requiresDependency. These structures are converted to runtime graph structures during the information extraction processing. In this runtime structure, spread activation techniques can be applied to reason about the existence of domain concepts in unstructured text corpus. Below is a list of the custom annotations:

groundsTo:Bank#FinancialTransaction

providesCapability:GFl#FinancialTransaction-Capability

providesDependency:GF#Currency-Dependency

providesDependency:GF#FinancialTransaction-Dependency

4.2.3 Bank Ontology WordNet Grounding

The primary purpose of the Grounding Framework is to position the domain ontology for use during information extraction processing. To carry out this purpose, the Grounding Framework needs relationships that bind the domain ontology concepts to natural language. WordNet Synset and SenseKey information is used for this purpose. Leveraging the lexical annotations provided by [12], the Grounding Framework maintains the lexical groundings by attaching them to gf:Dependency classes.

As previously discussed, each of the owl:Restriction declarations from the domain ontology are captured through the use of gf:Dependency classes and custom annotations on gf:Node classes. The linkage of the Dependency classes to the English language is specified by gf:wordnet3.0SenseKey and gf:wordnet3.0Synset annotations. These annotations are attached to the Dependency class representative of its domain class counterpart.

During the SGM information extraction process, these annotations are used to match NLP mentions with domain ontology concepts through the use of synonym, hypernym, and hyponym comparisons. The WordNet annotations for the FinancialTransaction-Dependency class in the Grounding Framework are listed below:

wordnet3.0SenseKey "(dealing%1:04:02::)"

wordnet3.0SenseKey "(dealings%1:04:00::)"

wordnet3.0SenseKey "(financial%3:01:00::)"

wordnet3.0SenseKey "(fiscal%3:01:00::)"

wordnet3.0SenseKey "(transaction%1:04:00::)"

wordnet3.0Synset "(01106808n)"

wordnet3.0Synset "(02847894a)"

5 Results

To test our approach we focused on two areas. First, can we successfully generate the Grounding Framework from existing ontology? Second, once the Grounding Framework is created, is it compatible with the existing SGM matching algorithms? Each of these tests along with their results is discussed in the following sections.

5.1 Generation from Existing Ontology

Three different domain ontologies were used to test the generation of the Grounding Framework.

5.1.1 Bank Ontology

This ontology was built expressly for the purposes of the research. As previously discussed, this ontology is a small ontology that also makes use of the BFO and RO upper level ontology. The Bank ontology has 56 classes and 24 object properties using 6 namespaces. Generation time for the lexicalization was 6.14 seconds while generation time for the Grounding Framework was 2.98 seconds.

5.1.2 Wine Ontology

The Wine Ontology[15] is a popular sample ontology used in the OWL specification documents. The Wine Ontology has 138 classes and 16 object properties using 2 namespaces. Generation time for the lexicalization was 6.5 seconds while generation time for the Grounding Framework was 3.34 seconds.

5.1.3 Large BFO Ontology

In order to test a larger ontology and one that incorporated the concepts of BFO but was not developed with the intentions of testing the Grounding Framework, we used what we refer to as the Large BFO Ontology. The Large BFO Ontology has 14 ontology files, 1530 classes and 95 object properties. Generation time for the lexicalization was 114 seconds while generation time for the Grounding Framework was 61 seconds.

5.1.4 Grounding Framework Generation Conclusions

Based on these tests, it is evident that significant time savings can be realized when compared to manually creating SGM compliant structures. Our testing showed that an experienced developer can manually create a Grounding Framework at approximately 2 minutes per domain ontology class and 2 minutes per object property. The automatically generated times per class/object property for each ontology are below:

Bank Ontology	0.114 seconds per class/object property
Wine Ontology	0.063 seconds per class/object property
Large BFO	0.107 seconds per class/object property

Our tests also demonstrated that multiple types of ontology can be utilized for generating a Grounding Framework.

In comparing the manually created Grounding Framework versus the automatically generated versions, two shortcomings are recognized. First, the depth of the SGM definition is completely driven by the domain ontology. This often limits the richness in the Dependency and Capability definitions, thus limiting information extraction results. Second, the total reliance on the lexicalizing of the ontology using WordNet also reduces the richness of the semantic definitions. This approach excludes domain specific terms, slang and derivatives not found in WordNet. Both of these shortcomings are addressed in Section 6 and in [12].

5.2 Testing SGM Compatibility

The final testing involved parsing test sentences with the SGM information extraction algorithm using the automatically generated Grounding Framework. The results were compared against those obtained using a manually generated Grounding Framework over the same sentences.

5.2.1 Test Sentence One

Results of matching the second test sentence to the domain ontology using the Grounding Framework follows.

‘John Smith’ is identified as a person’s name, and a personCapability individual is asserted. ‘entered’ is identified as ‘movement’ verb through WordNet, and a movementCapability individual is asserted. ‘Summit Bank’ is not identified as a named entity. These results are the same as the original OGEP SGM processing.

5.2.2 Test Sentence Two

Results of matching the first test sentence to the domain ontology using the Grounding Framework are as follows.

‘John Smith’ is identified as a person’s name, and a personCapability individual is asserted. ‘withdrew’ is identified as ‘transaction’ verb through WorldNet, and a transactionCapability individual is asserted. ‘$10,000’ matches a currency format and is assigned the monetaryValueCapability and the currencyCapability. These results are the same as the original OGEP SGM processing.

5.2.3 SGM Compatibility Conclusions

Based on the simple parsing tests as described, we have concluded that the automatically generated Grounding Framework is completely compatible with the SGM matching algorithms.

6 Future Work

The original objectives of our research were accomplished; primarily, develop a mechanism to automatically generate the Grounding Framework from existing, lexicalized ontology. One of the limitations encountered when using existing ontology was the classic model-based issue which is the fact that the Grounding Framework is completely reliant on the composition of the domain ontology (i.e. the model).

Since domain ontology is generally not developed with information extraction in mind, there are potential weaknesses implicit between the ontological construction and what is most beneficial for information extraction. The primary area of concern was the lack of depth in the owl:Restrictions within the domain ontology. SGM has been shown to produce the best results when a large number of restriction-derived Dependencies are present for the ontological concepts that are to be matched. This is not a typical means for constructing domain ontology.

Future work is planned to introduce supplementary logic into the Grounding Framework generation process to attempt to dynamically enrich current ontological concepts as information extraction targets. One potential strategy is to leverage lexical databases such as WordNet, VerbNet, or FrameNet to elaborate the owl:Restriction entities in the context of language derivations and relationships.

7 Acknowledgements

This material is based in part upon work supported by the Intelligence Advanced Research Projects Activity (IARPA) via Air Force Research Laboratory contract number

FA8650-10-C-7062. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, AFRL or the U.S. Government.

8 References

[1] A. Deshpande, C. Guestrin, S. Madden, J. Hellerstein and W. Hong, “Model-driven data acquisition in sensor networks”; VLDB '04 Proceedings of the Thirtieth international conference on Very large data bases - Volume 30.

[2] Songnian Zhou, “A Trace-Driven Simulation Study of Dynamic Load Balancing”; IEEE Transactions on Software Engineering, Vol. 14, No. 9. (September 1988), pp. 1327-1341.

[3] K.Kiili, “Digital game-based learning: Towards an experiential gaming model”; The Internet and Higher Education, Vol. 8, No. 1. (MarchJanuary 2005), pp. 13-24.

[4] Hoifung Poon and Pedro Domingos “Unsupervised Ontology Induction from Text”; Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 296–305, Uppsala, Sweden, 11-16 July 2010.

[5] Maria Vargas-vera , Enrico Motta , John Domingue , Simon Buckingham Shum , Mattia Lanzoni, “Knowledge Extraction by using an Ontology-based Annotation Tool”; In K-CAP 2001 workshop on Knowledge Markup and Semantic Annotation.

[6] Adrian Silvescu, Jaime Reinoso-castillo, Vasant Honavar, “Ontology-Driven Information Extraction and Knowledge Acquisition from Heterogeneous, Distributed Autonomous Biological Data Sources”; In Proceedings of the IJCAI-2001 Workshop on Knowledge Discovery from Heterogeneous, Distributed, Autonomous, Dynamic Data and Knowledge Sources.

[7] C. Anken, A. Stirtzinger and B. McQueary. “Goal-Driven Semi-Automated Generation of Semantic Models”; In Proceedings of the IEEE Symposium on Computational Intelligence for Security and Defence Applications (CISDA), July, 2009.

[8] A. Stirtzinger, C. Anken. “Semi-automated ontology generation and evolution”; In Proceedings of the SPIE, Volume 7347, pp. 734706-734706-10, April, 2009.

[9] Grenon P, Smith.B., Goldberg L.: Biodynamic ontology: applying BFO in the biomedical domain. Stud Health Technol Inform. 102 (2004) 20-38

[10] Smith B, Ceusters W, Klagges B, Kohler J, Kumar A, Lomax J, Mungall CJ, Neuhaus F, Rector A, Rosse C Relations in Biomedical Ontologies.Genome Biology, 2005, 6:R46

[11] Downloadable from: http://www.obofoundry.org/ro/

[12] Joshua Powers and Anthony Stirtzinger, “Lexicalizing an Ontology”; conference proceedings 2011 International Conference on Information and Knowledge Engineering (IKE) July 18-22, 2011.

[13] G. A. Miller. “WordNet: A Lexical Database for English”; Communications of the ACM, Vol. 38 No. 11: 39—41, December, 1995.

[14] Attila Ondi, Anthony Stirtzinger: Information Discovery Using VerbNet: Managing Complex Sentences. IC-AI 2010: 268-276.

[15] W3C. Wine Ontology; http://www.w3.org/TR/owl-guide/wine.rdf, December, 2003.