ONTOLOGY FOR THE INTELLIGENCE COMMUNITY

C4I CENTER

O I C

2008

Towards Effective Exploitation and Integration of Intelligence Resources

December 3-4, 2008
George Mason University

ncor

IDI Innovative Decisions, Inc.

SAAB TECHNOLOGIES

SAAB

Table of Contents

Leveraging Emergent Ontologies in the Intelligence Community

Jim Starz, Jason Losco, Brian Kettler, Rachel Hingst, and Chris Rouff
Lockheed Martin Advanced Technology Laboratories
jstarz@atl.lmco.com

***Abstract* – The vision of a Semantic Web of intelligence knowledge has yet to be fully realized, in part because of the tough challenges of ontology engineering and maintenance. Recent developments on the World Wide Web and IC intranets demonstrate that individual users are willing to supply structured information conforming to de facto standards. This can be most prominently seen in "peer produced" folksonomies and knowledge bases such as Wikipedia and Intellipedia, its cousin. Though these structures lack the machine reasoning potential of highly engineered ontologies, for many purposes they are "good enough". This paper describes Contrail, a prototype information management application, that leverages an "emergent" ontology from Wikipedia to model a intelligence analyst's context and exploit that model to aid information retrieval, refinding, and sharing**

I. INTRODUCTION

The widespread adoption of Semantic Web and other ontology-based applications in the intelligence community (and indeed the wider web) is that quality ontologies are difficult to build, maintain, and exploit. Ontology engineering requires significant subject domain expertise and knowledge engineering skills. For all-source and other kinds of analysts, such ontologies span a broad range of subject domains, which are constantly evolving.

Wikipedia and Intellipedia are approaches to capturing this broad range of knowledge from the community without requiring pre-built ontologies. These knowledge bases are not without structure. A prominent example is the World Wide Web's Wikipedia, which contains over fifteen million pages. The structure for pages of the same type are very similar, illustrating that people are willing to provide structure in the form of lightweight ontology-like information. This similarity is discussed in the work on Wikitology [4] and dbpedia [1].

While such "ontologies" might not support formal automated reasoning system well, they can support other useful applications. Our research investigated leveraging emergent ontologies for the purposes of representing user models of analysts. The work used an ontology derived from Wikipedia. This paper describes our prototype application, its use of Wikipedia, and some preliminary results.

II. THE CONTRAIL TOOLS

The Contrail tools help analysts find, organize, re-find, and share unstructured and semi-structured information obtained from the Web (or Intelink), email, documents, and other sources [2]. While our focus is on intelligence analysts, these tasks are those of many knowledge workers. Contrail has been evaluated in several experiments with real intel analysts on open source intelligence tasks.

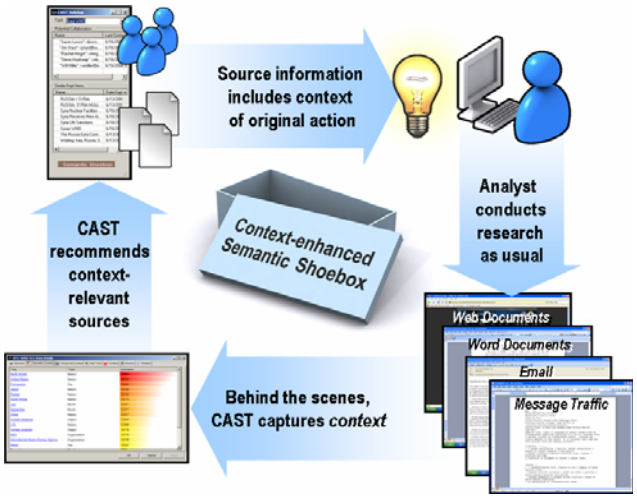


Fig. 1. High-Level Concept of Operations for Contrail Tools

Fig. 1 shows the high-level concept of operations for the Contrail tools as an analyst does her research online, she finds relevant items through web browsing, web searches, reading email, etc. Through instrumentation and logging services, Contrail is notified of these "information keeping actions", such as the bookmarking of a web page. Contrail then performs a semantic analysis of each kept information item's content using text analytics and other methods. Using the results of this analysis, Contrail updates its model of the analyst's context and stores a copy of the kept item in her Semantic Shoebox. A user's Semantic Shoebox can be thought of as a semantically grounded container for

accumulated pieces of information. Contrail supports the sharing and retrieval of kept items from other analyst's shoeboxes. The contextual knowledge appended to these items by Contrail helps one analyst quickly understand the potential relevance and pedigree of an item retrieved from another analyst's shoebox.

The Contrail Refinder tool, shown in Fig. 2, presents a more comprehensive view of a Semantic Shoebox and displays a variety of information (textually and graphically) associated with a kept item including its metadata, content, and context tags. A user may do a one button search to display those items most relevant to his current context. Contrail also presents context-relevant recommendations for stored items and potential collaborators in a desktop sidebar.

At the core of Contrail is its Context Aggregator which maintains and updates the user's context at each keeping action. Concepts and their instances (specific people, organizations, locations, etc.) are extracted from the text of the kept item using a commercial entity extractor. A spreading activation algorithm is used to find related concepts in a knowledge base (KB). These related concepts might not be explicitly mentioned in the text itself. Extracted and related concepts are thus associated with an activation level and the most active concepts represent the user's current context. Contrail's KB, grounded in hand-built OWL ontologies extending the SUMO [3].

This approach worked well, as judged in experiments with analysts who periodically reviewed Contrail's model of their contexts. Contrail's use of an ontologically-grounded knowledge base of concepts, however, presented significant ontology engineering and maintenance challenges, as well being limited by the underlying entity extractor used. These challenges – all potential barriers to Contrail's deployment – included the potential breadth required for ontologies and the handling of new concepts and entities in these dynamic domains.

III. USING WIKIPEDIA

To alleviate these issues, we have replaced the static ontology based context representation with one based on Wikipedia. We used IR based techniques to relate documents with pages in Wikipedia and associated a score with each relationship. One significant benefit of this approach is the elimination of the need for knowledge engineering to update the "ontology." Wikipedia serves as a publicly maintained emergent ontology, allowing for user context to shift as the world changes.

Specifically, keeping actions performed by the users associate their interests in particular documents or snippets of text. Based on this text, we query a Lucene index of Wikipedia to obtain pages that may be of interest to the user. A weighted merge of the query results is performed with their existing contextual information to form their updated user model.

It should be noted that given the scale of Wikipedia, such queries are very resource intensive. Despite this challenge, the results from leveraging the emergent ontology from Wikipedia appear promising.

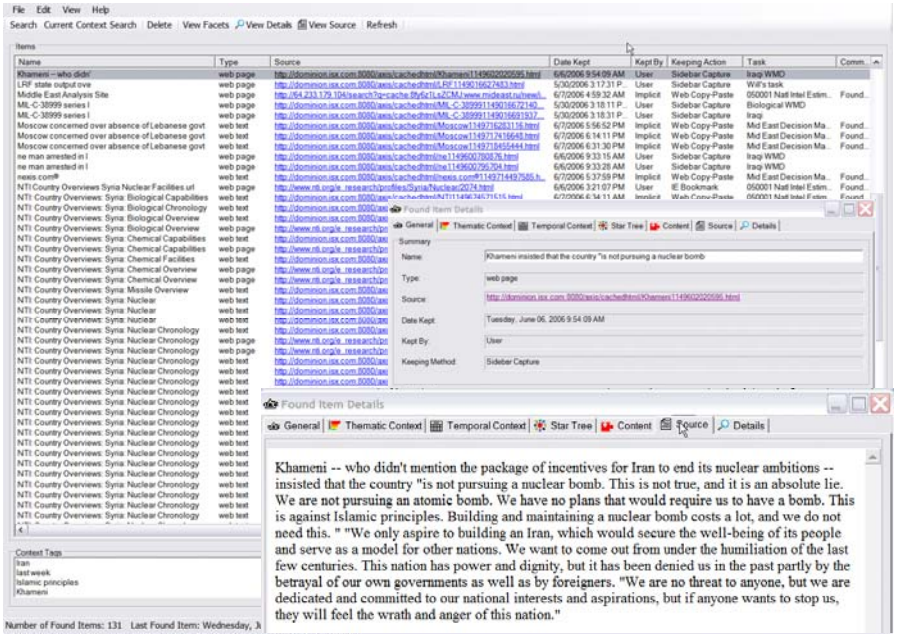


Fig. 2. Contrail Refinder (Item Browser, Item General Details, and Item Source Details screens)

IV. EVALUATION

Initial informal experimentation using this new approach for user modeling has shown significant improvements over using a traditional static ontology in representing user context. The new approach improves finding documents and collaborators. There was also anecdotal evidence that the biggest advantage occurred when new concepts and instances were present in the emergent ontology that could be immediately leveraged. An example of the differences is shown below.

TABLE 1
Example of context terms from static ontology and Wikitology derived terms

Static Ontology	Wikitology
Indonesia	United Malyas Nat. Org.
Malaysia	Ketuanan Melayu
Singapore	Mahatir bin Muhamed
June	Islam in Malaysia
2002	Anwar Ibraham

The Wikitology approach consistently provided more specific terms that may not easily be found in an ontology or by text analytics packages. Using the old approach, we found general terms would dominate the user context. The breadth of Wikipedia does add the potential for significant noise, such as pages about specific dates. Though Wikipedia is relatively comprehensive, for specific domains pages may not exist. For emerging concepts, it is critical to mirror Wikipedia and update the index regularly. The results of this evaluation will be documented in a future research paper.

V. FUTURE WORK

Our research agenda includes further investigations to determine new applications where emergent ontologies can be applied. This investigation will include tools leveraging these ontologies for enhanced semantic authoring. We also plan to investigate the extraction of rules from patterns in emergent ontologies. A major focus area will be handling the significant scale and rapid updates of Wikipedia. Both of the aspects provide significant challenges and opportunities. Finally, we plan to make additional extensions to the Contrail suite of tools to extend the representation of user models.

VI. CONCLUSION

In the large distributed nature of the World Wide Web, leveraging massive convergence in terminology and structure can be highly useful. While these structures may not replace formal ontologies, they can be appropriate for certain applications and they can help bridge a gap to more formal structures. We have demonstrated that the use of the ontological structure of Wikipedia for representing context has advantages over human-engineered ontologies for at least one application and likely many others.

ACKNOWLEDGEMENTS

Many of the concepts applied in this paper were motivated by conversations with Tim Finin of the University of Maryland at Baltimore County.

REFERENCES

[1] S. Auer, C. Bizer, J. Lehmann, G. Kobilarov, R. Cyganiak, Z. Ives: DBpedia: A Nucleus for a Web of Open Data. In Aberer et al. (Eds.): The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11–15, 2007. Lecture Notes in Computer Science 4825 Springer 2007, ISBN 978–3-540–76297–3.

[2] B., Kettler (2008). Putting Knowledge in Context to Facilitate Collaboration. In Proceedings of the 2008 International Symposium on Collaborative Technologies and Systems (May 19-23, 2008 in Irvine, CA). IEEE, 313-320.

[3] I. Niles, and A. Pease. 2001. Towards a standard upper ontology. In Proceedings of the international Conference on Formal ontology in information Systems - Volume 2001 (Ogunquit, Maine, USA, October 17 - 19, 2001). FOIS '01. ACM, New York, NY, 2-9.

[4] Z, Syed et al., "Wikipedia as an Ontology for Describing Documents", In Proceedings, Proceedings of the Second International Conference on Weblogs and Social Media, March 2008.

[5] M. Williams and J. Hollan. (1981). The Process of Retrieval from Very Long-Term Memory. Cognitive Science 5: 87-119.

Automatic Ontology Creation from Text for National Intelligence Priorities Framework (NIPF)

Mithun Balakrishna, Munirathnam Srikanth
Lymba Corporation
Richardson, TX, 75080, USA
Email: {mithun,srikanth}@lymba.com

***Abstract*—Analysts are constantly overwhelmed with large amounts of unstructured data. This holds especially true for intelligence analysts with the task of extracting useful information from large data sources. To alleviate this problem, domain-specific and general-purpose ontologies/knowledge-bases have been proposed to help automate methods for organizing data and provide access to useful information. However, problems in ontology creation and maintenance have resulted in expensive procedures for expanding/maintaining the ontology library available to support the growing and evolving needs of the Intelligence Community (IC). In this paper, we will present a semi-automatic development of an ontology library for the National Intelligence Priorities Framework (NIPF) topics. We use Jaguar-KAT, a state-of-the-art tool for knowledge acquisition and domain understanding, with minimized manual intervention to create NIPF ontologies loaded with rich semantic content. We also present evaluation results for the NIPF ontologies created using our methodology.**

***Index Terms*—ontology generation, National Intelligence Priorities Framework (NIPF).**

I. Introduction

Analysts are constantly plagued and overwhelmed by large amounts of unstructured, semi-structured data required for extracting useful information [1]. Over the past decade, ontologies and knowledge bases have gained popularity for their high potential benefits in a number of applications including data/knowledge organization and search applications [2]. The data processing burden on the intelligence analysts have been relieved with the integration of ontologies to help automate methods for organizing data and provide access to useful information [3].

Though a number of applications can and have benefited due to their integration with domain-specific and general-purpose ontologies/knowledge-bases, it is very well known that ontology creation (popularly referred to as the *knowledge acquisition bottleneck* [2]) is an expensive process [4], [5]. The modeling of ontologies for non-trivial domains/topics is difficult and time/resource consuming. The *knowledge acquisition bottleneck* problems in ontology creation and maintenance have resulted in expensive procedures for maintaining and expanding the ontology library available to support the growing and evolving needs of the Intelligence Community (IC).

In this paper, we present a semi-automatic development of an ontology library for the 33 topics defined in the National Intelligence Priorities Framework (NIPF). NIPF is the *Director of National Intelligence's (DNI's) guidance to the Intelligence Community on the national intelligence priorities approved by the President of the United States of America* [6].

Lymba's Jaguar-KAT [3], [7] is a state-of-the-art tool for knowledge acquisition and domain understanding. We use Jaguar to create rich NIPF ontologies by extracting deep semantic content from NIPF topic specific document collections while keeping the manual intervention to a minimum. In this paper, we discuss the technical contributions of automatic concept and semantic relation extraction, automatic ontology construction, and the metrics to evaluate ontology quality.

II. Automatic Ontology Generation

Jaguar automatically builds domain-specific ontologies from text. The text input to Jaguar can come from a variety of document sources, including Text, MS Word, PDF and HTML web pages, etc. The ontology/knowledge-base created by Jaguar includes the following constituents:

- Ontological Concepts: basic building blocks of an ontology
- Hierarchy: structure imposed on certain ontological concepts via transitive relations that generally hold to be universally true (e.g. ISA, Part-Whole, Locative, etc)
- Contextual Knowledge Base: semantic contexts that encapsulate knowledge of events via semantic relations
- Axioms on Demand: assertions about concepts of interest generated from the available knowledge; this is useful for reasoning on text

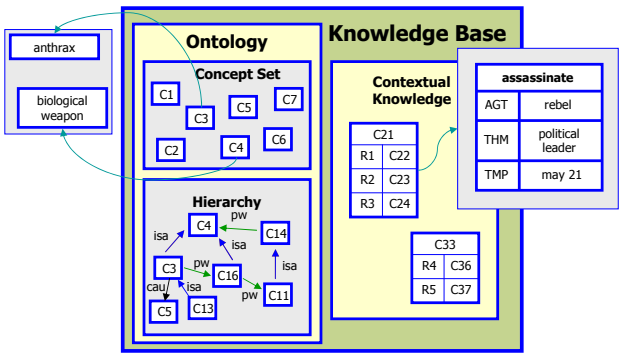


Fig. 1. An example Jaguar knowledge-base containing concepts, hierarchy and contextual knowledge.

Figure 1 shows an example Jaguar knowledge-base containing concepts, hierarchy and contextual knowledge. The

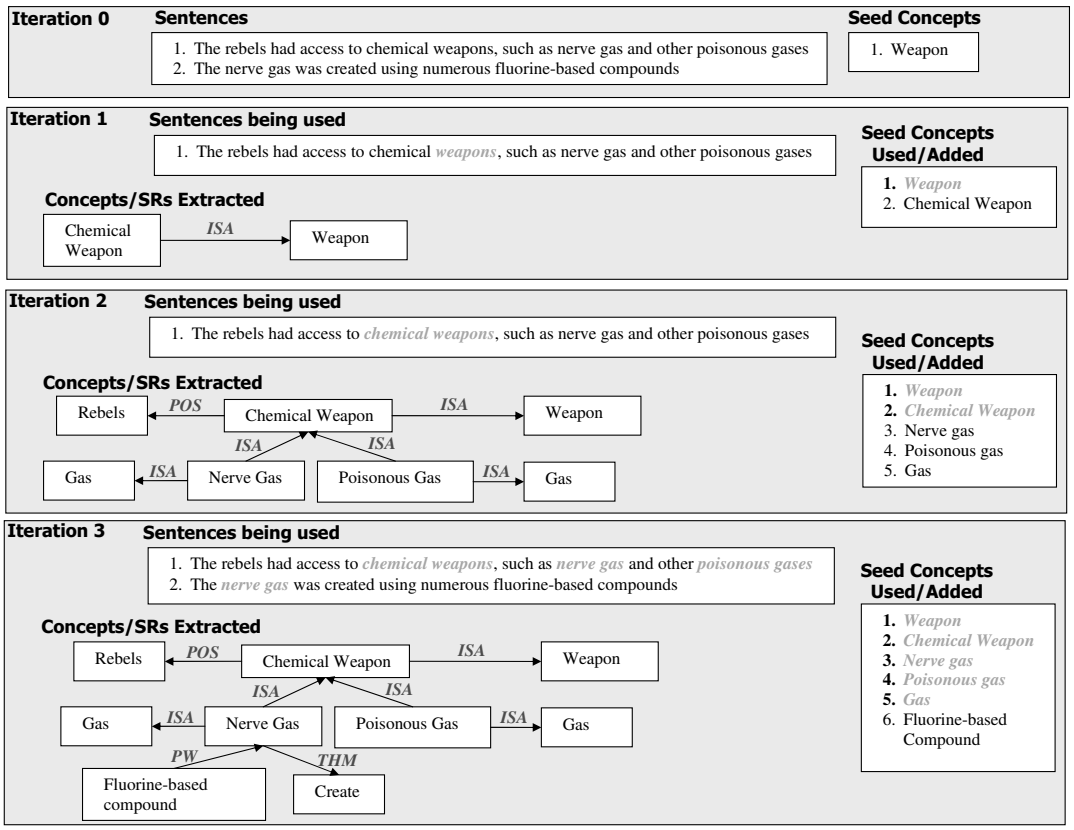


Fig. 2. An example depicting Jaguar's iterative process of extracting concepts and semantic relations of interest using seed concepts.

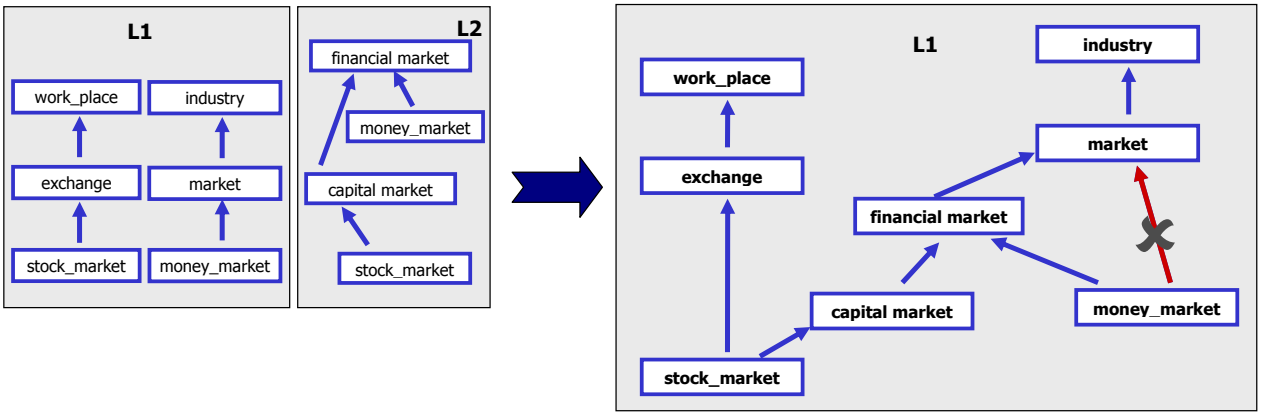


Fig. 3. An example depicting Jaguar's merging of two ontologies through conflict resolution algorithms.

input to Jaguar includes a document collection (Text, MS Word, PDF and HTML web pages, etc.) and a seeds file containing the concepts/keywords of interest in the domain. Jaguar's ontology creation involves complex text processing using advanced Natural Language Processing (NLP) tools, and an advanced knowledge classification/management algorithm. A single run of Jaguar can be divided into the following two major phases:

- Text Processing
- Classification/Hierarchy Formation

In Text Processing, the first step is to extract textual content from the input document collection. The text files then go through a set of NLP processing tools: named-entity recognition, part-of-speech tagging, syntactic parsing, word-sense disambiguation, coreference resolution, and semantic parsing (or semantic relation discovery) [8], [9]. The concept discovery module then extracts the concepts of interest using the input seeds set as a starting point and growing it based on the extracted NLP information [3].

The classification module forms a hierarchical structure within the set of identified domain concepts via transitive relations that generally hold to be universally true (e.g. ISA, Part-Whole, Locative, etc). Jaguar uses well-formed procedures [7] to impose a hierarchical structure on the discovered concepts

set using the semantic relations discovered by Polaris [1] and with WordNet [10] as the upper ontology.

A. Automatically Building NIPF Ontologies

In this paper, we use Jaguar to create an ontology library for the 33 topics defined in NIPF. For each NIPF topic, we collected 500 documents from the web (the $Weapons$ topic was an exception and its collection had only 50 Wikipedia documents) and manually verified their relevance to the corresponding topic. We then use Jaguar to create an ontology, for each identified NIPF topic. Jaguar builds each ontology with rich semantic content extracted from the corresponding NIPF topic document collection while keeping the manual intervention to a minimum. These ontologies are fine-tuned to contain the level of detail desired by an analyst.

1) Extracting Textual Content: We first extract text from the input NIPF document collections and then filter/clean-up the extracted text. The NIPF text input to Jaguar comes from all possible document types, including MS Word, PDF and HTML web pages, and is therefore prone to having many irregularities, such as incomplete, strangely formatted sentences, headings, and tabular information. The text extraction and filtering mechanism of Jaguar is a crucial step that makes the input acceptable for subsequent NLP tools to process it. The extraction/filtering rules include, conversion/removal of non-ASCII characters, verbalization of Wikipedia infoboxes and tables, conversion of punctuation symbols, among others.

2) Initial Seed Set Selection: For each NIPF topic, Jaguar is provided with an initial seed set containing on average 51 concepts of interest. The seed set is used to determine the set of text sentences of interest in a topic's document collection. The initial seed set selection for the NIPF topic was performed manually based on the concepts found in the topic descriptions. The initial seed selection process is the *only manual step* that we use in our NIPF ontology creation process. We are currently exploring automated methods for creating the initial seed set using a combination of statistical and semantic clues in the document collection.

3) Concept and Relation Discovery: For each NIPF topic, the set of text files extracted from the document collection are processed through the entire set NLP tools listed in Section II. The NLP processed data files are then passed through the concept discovery module, which identifies noun concepts in sentences which are related to the NIPF topic target words or seeds. The concept discovery module analyzes the syntactic parse tree of each processed sentence and scans them for noun phrases. Though Jaguar has the capability to extract verb concepts by analyzing verb phrases, for our current NIPF ontology creation experiment, we focused only on noun concepts and their semantic relations. Each noun phrase is then processed and well-formed noun concepts are extracted based on a set of syntactic patterns and rules.

Noun concepts (which are part of the seed set), their semantic relations (extracted from the semantic parser, Polaris [8], [9]) and the noun concepts involved in semantic relations with the seed set concepts are added into data structures for subsequent processing into the ontology's hierarchy. The resulting data structures are processed and used to populate one or many semantic contexts, groups of relations or nested contexts which hold true around a common central concept. The seed set is then augmented with concepts that have hierarchical relations with the target words or seeds. The entire process of sentence selection, concept extraction, semantic relation extraction and seed concepts set augmentation is repeated in an iterative manner, n number of times (by default, n is set to 3). While processing the NIPF topic collections through Jaguar, we used ISA, Part-Whole and Synonymy semantic relations for automatically augmenting the seeds concept set. Figure 2 depicts this iterative process of extracting concepts and semantic relations of interest using seed concepts.

4) Creating Concept Hierarchies: The extracted NIPF topic noun concepts and semantic relations are fed to the classification module to determine the hierarchical structure. Certain hypernymy relations discovered via classification contain anomalies (causing cycles) or redundancies. Hence, we run them through a *conflict resolution engine* to detect and correct inconsistencies. The *conflict resolution engine* creates a NIPF topic hierarchy link by link (relation by relation) and follows a conflict avoidance technique, wherein each new link is tested for causing inconsistencies before being added to the hierarchy.

5) Ontology Merging: Although single runs of Jaguar yield rich NIPF ontologies, Jaguar's real power lies in providing an ontology maintenance option to layer ontologies from many different runs. Figure 3 depicts the process of merging two ontologies through conflict resolution algorithms. Jaguar can merge disparate ontologies or add new knowledge by using the aforementioned conflict resolution techniques. The merge tool merges the two ontologies' concept sets, hierarchies (using conflict resolution), and their knowledge bases (set of semantic contexts). Given two ontologies or knowledge bases, ontology merging is performed by enumerating the relations in the smaller ontology and adding them to the larger or reference ontology. A relation may either be represented by a similar relation in the reference ontology, may create a redundant path between concepts or may be a new relation that can be added to the reference ontology. The conflict resolution techniques are then used for handling the conflict induced in the ontology to generate a merged ontology. Merging is useful for distributed or parallel systems where small chunks of the input text may be processed on some portions of the system and then subsequently merged. It also provides a foundation for future work in contextual reasoning and epistemic logic. The resulting rich NIPF knowledge bases can be viewed at many different levels of granularity, providing an analyst with the level of detail desired.

III. Evaluation of Jaguar's NIPF Ontologies

Since the mid-1990s, various methodologies have been proposed to evaluate ontology generation/maintenance/reuse techniques [11]. All the proposed methodologies have focused

TABLE I
SUBSET OF SEMANTIC RELATIONS USED TO EVALUATE THE PERFORMANCE OF JAGUAR'S AUTOMATIC NIPF TOPICAL ONTOLOGY GENERATION FROM TEXT.

Semantic Relation	Definition	Example	Code
ISA	X is a (kind of) Y	[XY] [John] is a [person]	ISA
Part-Whole/Meronymy	X is a part of Y	[XY] [The engine] is the most important part of [the car] [XY] [steel][cage] [YX] [faculty] [professor] [XY] [door] of the [car]	PW
Cause	X causes Y	[XY] [Drinking] causes [accidents]	CAU

TABLE II
PERFORMANCE RESULTS FOR JAGUAR'S AUTOMATIC TOPICAL NIPF ONTOLOGY GENERATION FROM TEXT WITH RESPECT TO THE SEMANTIC RELATIONS DEFINED IN TABLE I.

Number of Annotators	NIPF Topic	Precision		Coverage		F-Measure	
		Correctness	Correctness+ Relevance	Correctness	Correctness+ Relevance	Correctness	Correctness+ Relevance
3	Weapons	**0.610090**	**0.501499**	0.702424	0.657122	**0.653009**	0.568859
1	Missiles	0.533867	0.485364	0.793775	**0.777747**	0.63838	**0.597715**
2	Illicit Drugs	0.471938	0.274506	0.801422	0.701122	0.594053	0.39454
1	Terrorism	0.388788	0.291019	**0.822285**	0.776206	0.527953	0.423323

TABLE III
SEMANTIC RELATION AND CONCEPT EXTRACTION STATISTICS FOR THE EVALUATED NIPF ONTOLOGIES PRESENTED IN TABLE II.

NIPF Topic	Unique Semantic Relations					Unique Concepts		
	ISA	PW	CAU	Others	**Total**	In ISA/PW/CAU	Others	**Total**
Weapons	1683	766	113	946	3508	2620	1012	3473
Missiles	2939	2296	646	2692	8573	5982	3539	7873
Illicit Drugs	2356	2040	817	5464	10677	5107	4982	7935
Terrorism	2590	4219	1497	5405	13711	7929	6247	11638

on some facet of the ontology generation problem, and depend on the type of ontology being created/maintained and the purpose of the ontology [12]. It is noted that not much progress has been achieved in developing a comprehensive and global technique for evaluating the correctness and relevance of ontologies [13].

$$
\begin{aligned}
Pr(Correctness) &= \frac{N_j(correct)+N_j(irrelevant)}{N_j(correct)+N_j(incorrect)+N_j(irrelevant)} \\
Pr\begin{pmatrix} Correctness \\ + \\ Relevance \end{pmatrix} &= \frac{N_j(correct)}{N_j(correct)+N_j(incorrect)+N_j(irrelevant)} \\
Cvg(Correctness) &= \frac{N_j(correct)+N_j(irrelevant)}{N_g(correct)+N_g(irrelevant)+N_g(added)} \\
Cvg\begin{pmatrix} Correctness \\ + \\ Relevance \end{pmatrix} &= \frac{N_j(correct)}{N_g(correct)+N_g(added)}
\end{aligned}
\tag{1}
$$

We evaluated the quality of Jaguar's NIPF ontologies by comparing them against manual gold annotations. Following the ontology evaluation levels defined in [12], our evaluations are focused on the *Lexical, Vocabulary, or Data Layer* and the *Other Semantic Relations* levels. For a NIPF topic, the ontology and document collection were manually annotated by several human annotators and used in the evaluation of the ontology. Viewing an ontology as a set of semantic relations between two concepts, the annotators:

- Labeled an entry *correct* if the concepts and the semantic relation are correctly detected by the system else marked the entry as *Incorrect*
- Labeled a *correct* entry as *irrelevant* if any of the concepts or the semantic relation are irrelevant to the domain
- From the sentences *added new entries* if the concepts and the semantic relation were omitted by Jaguar

The annotation rules provide feedback on the automated concept tagging and semantic relation extraction and also are used for computing precision (Pr) and coverage (Cvg) metrics for the automatically generated ontologies. Equations in (1) capture the metrics defined by Lymba to evaluate Jaguar's automatic topical NIPF ontology generation from text. In (1), $N_j(.)$ gives the counts from Jaguar's output and $N_g(.)$ correspond to counts in the user annotations. Table II presents our initial evaluation results for 4 NIPF topics using a subset of 3 semantic relations (ISA, PW and CAU relations) defined in Table I. Table III presents the semantic relation and concept extraction statistics for the four NIPF ontologies being evaluated in this paper.

We use the metrics defined in (1) to evaluate the ontologies against the manual annotations from different human annotators. The results in Table II represent the evaluation scores which have been averaged over the results for different annotators. The first column in Table II identifies the number of annotators for each topic. Jaguar obtained the best *Precision* results in both *Correctness* and *Correctness+Relevance* evaluations for the *Weapons* NIPF topic. Please note that as shown in Table III, smaller number of concepts/semantic-

relations were extracted for this topic due to its smaller collection size (50 documents versus the 500 document set for the other topics). The *Terrorism* NIPF topic obtained the best *Coverage* result for the *Correctness* evaluation and it was also very close to the best *Coverage* result obtained by the *Missiles* NIPF topic for the *Correctness+Relevance* evaluation. The *Weapons* NIPF topic obtained the best *F-Measure* result ($\beta = 1$) for the *Correctness* evaluation while the *Missiles* NIPF topic obtained the best *F-Measure* result for the *Correctness+Relevance* evaluation.

IV. Conclusions and Future Work

In this paper, we presented the semi-automatic development of an ontology library for the NIPF topics. We use Jaguar-KAT, a state-of-the-art tool for knowledge acquisition and domain understanding, with minimized manual intervention to create NIPF ontologies loaded with rich semantic content. We also defined evaluation metrics to assess the quality of the NIPF ontologies created using our methodology. We evaluated a subset of Jaguar's NIPF ontologies by comparing them against manual gold annotations. The results look very promising and show that a decent amount of knowledge was automatically and accurately extracted by Jaguar from the input document collection while keeping the manual intervention in the process to a minimum. We plan to perform further analysis of the results and identify methods for improving the precision and coverage of text processing and ontology generation.

References

[1] D. Bixler, D. Moldovan, and A. Fowler, "Using knowledge extraction and maintenance techniques to enhance analytical performance," in *Proceedings of International Conference on Intelligence Analysis*, 2005.

[2] P. Cimiano, *Ontology Learning and Population from Text: Algorithms, Evaluation and Applications*. Springer, 2006.

[3] D. Moldovan, M. Srikanth, and A. Badulescu, "Synergist: Topic and user knowledge bases from textual sources for collaborative intelligence analysis," in *CASE PI Conference*, 2007.

[4] E. Ratsch, J. Schultz, J. Saric, P. C. Lavin, U. Wittig, U. Reyle, and I. Rojas, "Developing a protein-interactions ontology," *Comparative and Functional Genomics*, vol. 4, no. 1, pp. 85–89, 2003.

[5] H. Pinto and J. Martins, "Ontologies: How can they be built?" *Knowlegde and Information Systems*, vol. 6, no. 4, pp. 441–464, 2004.

[6] "FBI: National Security Branch - FAQ," Last accessed on Jul 21, 2008, available at `http://www.fbi.gov/hq/nsb/nsb_faq.htm#NIPF`.

[7] D. I. Moldovan and R. Girju, "An interactive tool for the rapid development of knowledge bases," *International Journal on Artificial Intelligence Tools*, vol. 10, no. 1-2, pp. 65–86, 2001.

[8] A. Badulescu, "Classification of semantic relations between nouns," Ph.D. dissertation, The University of Texas at Dallas, 2004.

[9] R. Girju, A. M. Giuglea, M. Olteanu, O. Fortu, O. Bolohan, and D. Moldovan, "Support vector machines applied to the classification of semantic relations in nominalized noun phrases," in *Lexical Semantics Workshop in Human Language Technology (HLT)*, 2004.

[10] G. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[11] Y. Sure, G. A. Perez, W. Daelemans, M. L. Reinberger, N. Guarino, and N. F. Noy, "Why evaluate ontology technologies? because it works!" *IEEE Intelligent Systems*, vol. 19, no. 4, pp. 74–81, 2004.

[12] J. Brank, M. Grobelnik, and D. Mladenic, "A survey of ontology evaluation techniques," in *Data Mining and Data Warehouses (SiKDD)*, Ljubljana, Slovenia, 2005.

[13] A. Gangemi, C. Catenacci, M. Ciaramita, and J. Lehmann, "Modelling ontology evaluation and validation," in *European Semantic Web Symposium/Conference (ESWC)*, 2006, pp. 140–154.

Semantics for Information Sharing and Discovery in the Intelligence Community

Martin Thurn
Northrop Grumman Corp., 4805 Stonecroft Blvd., Chantilly VA 20151, 703-449-3803,
martin.thurn@ngc.com

***Abstract*—One of the distinguishing characteristics of the intelligence community is the strict security framework that is used to control classified information. A counterproductive side-effect of this strict security is that intelligence analysts are often not aware of information that is relevant to their analysis. Semantic technology and ontologies can help analysts discover relevant information even if that information is under the strictest controls and even if the analysts are not cleared to access the data. These techniques can be applied immediately within the current security framework of the intelligence community.**

***Index Terms*—Discovery, Information Sharing, Metadata, Redaction, Semantics**

I. Introduction

THE many agencies of the United States intelligence community – and the corresponding organizations of her friend and partner countries around the world – employ a strict security framework to protect and control classified information. The basis of this framework is that a person is granted access to a sensitive document only if they need to know those data to perform their duties.

This basis creates two immediate impediments to information sharing and discovery across the boundaries of security levels and compartments. When sensitivity classifications are assigned to an entire document, it prevents an unapproved analyst from seeing any portion of the document, even when the document may actually contain a mixture of sensitive and unclassified information. To make matters worse, it is often the case that an unapproved analyst is prevented from knowing even the existence of that document. In the former case, the analyst can at least ask for permission to read the document and fulfill her duties; in the latter case, there is virtually no hope for the analyst ever to see the data.

II. Physically-Separate Standard Semantic Metadata

We have developed an approach to discovering and sharing information that is particularly well-suited to the intelligence community, an approach based on physically-separate standard semantic metadata. "Metadata" is a general term that refers to data that describes other data. Metadata for a document may explicitly identify the title of the document, provide a table of all the geographic locations mentioned in the document, or include any other information about the properties or content of the document. "Physically separate" means that the metadata is stored in a separate file rather than being embedded within the data file itself – an important contrast to the dominant practice of embedding all metadata within documents. "Semantic" means that the metadata represents the meaning of the data, as opposed to just syntactic sugar. In particular, our approach focuses on expressing the semantics of the *content* of the document, i.e. the actual body text, rather than facts *about* the document which are typically found in the header. "Standard" means that the metadata is represented using semantics standards such as the Resource Description Framework (RDF) and Web Ontology Language (OWL). In addition, "Standard Semantic" means that the metadata strictly corresponds to an ontology so that the meaning is explicit and can be processed by automated tools.

Using physically separate semantic metadata for discovery is not a new idea – this is a technique that has been used successfully by libraries for centuries. A card (whether paper or electronic) in a card catalog is metadata for a book in a library's holdings. The card for a rare and delicate book is itself neither rare nor delicate, and therefore does not have to be subject to the same protections as the book itself. Whereas the book may be held in a special collection accessible only to approved scholars, the card describing the book can be publicly accessible, updated frequently, and copies can be distributed to other libraries. In contrast, metadata that is not physically separate from the data – metadata that appears in the front matter of a book, for instance – cannot be updated and can only be accessed by those who already have access to the book itself.

Within the intelligence community, working with physically separate metadata has all the advantages of working with catalog cards, and also solves fundamental security problems that stand in the way of discovery and sharing of information. There are two keys to this aspect of the solution. First, the physically separate metadata can be at a lower level of classification than the data itself. It is entirely possible that the very nature of the metadata makes it lower level; or the system can be specifically designed so that the metadata is of a lower classification, if necessary. Second, the physically separate metadata can be stored on a different network (or several different networks) than the original data. The bottom line is,

while an organization may not be able to share much of its data for security reasons, it may be able to share a great deal of metadata. That metadata will allow intelligence analysts to discover the existence of information that is important to them even if they have not been cleared to access the data itself.

It should also be noted that since electronic metadata files can be much larger than physical index cards in a traditional card catalog, the metadata may easily contain a wealth of valuable content information that can be exploited independently of the actual data file. Of course, the metadata might not have the same authority as the actual data (see the sample scenario below), but it certainly can be used to suggest hypotheses.

III. Ontologies for Discovery

Rich ontologies are essential to the success of the approach to discovery described here. Ontologies allow semantic searches to match even if the query concept is more specific or more general than the concept in the metadata. Semantic metadata is data about the meaning of the data. Meaning has the property that it can be abstracted, which is important for both discovery and security reasons. An aircraft ontology, for instance, may indicate that the B-2 is a stealth bomber, a stealth bomber is a type of bomber, and that bombers are a type of airplane. This will allow a semantic search for the concept "airplane" to discover documents that mention specific types of aircraft such as the B-2 (even when the documents do not contain the query word "airplane"). And if the fact that a B-2 was used for a particular mission makes a document classified, unclassified metadata can be generated by referring to the more abstract concepts of "stealth bomber" or, if necessary, "bomber" or just "airplane". By abstracting as little as possible to meet security requirements, the semantic metadata can make the maximum amount of information available for discovery and exploitation. Rich standard ontologies facilitate this type of searching and abstraction. In the ideal case, the ontologies themselves will be standards used across the intelligence community – a central topic of this conference.

Discovery based on physically-separate metadata is often viewed as a last resort – a technique to be used only when security restrictions prevent access to the data itself. Indeed, one could argue that it should be a last resort when only very basic document metadata (e.g. Title, Author, Date) is available. However, semantic metadata can be arbitrarily rich, containing a detailed, unambiguous, machine-interpretable version of the information contained in a document. Since rich metadata provides an unambiguous and direct representation of the meaning of a document, metadata can serve as a better basis for discovery and automatic exploitation than even the document itself. As rich semantic metadata becomes available for more and more documents in a repository, search recall should increase, because exact matches are not necessary; and as the metadata becomes richer, the precision should increase as well, since fine-grained concepts from an ontology are less ambiguous than English words. Once sufficiently rich semantic metadata is available, metadata-based discovery can exceed both the recall and the precision of keyword searching against full text documents.

IV. Sample Scenario of Semantic Discovery

An intelligence analyst is creating a map of the locations of certain objects of interest. In the past, creating such maps required reading intelligence cables that describe, in ordinary English, the locations of the objects at various times. The analyst would then have to type all the coordinates into a geographical information system (GIS) to create the map – a tedious and error-prone task.

In our approach, as each cable arrives, a metadata file is created that contains RDF descriptions of what objects were at what locations at what times based on standard ontologies. This RDF can be automatically generated using existing information extraction technology such as NetOwl from SRA International, TextTrainer from Northrop Grumman, or AeroText from Rocket Software. A semantic metadata search – either a live search initiated by the analyst, or an automated "batch" query that runs overnight – is then used to discover all the metadata files that describe locations of objects of interest. Having standard ontologies greatly facilitates the indexing and retrieval required for this type of search. Since RDF is completely structured, the resulting locations can automatically be loaded into the GIS application. As a result, maps that previously took weeks to create manually are now automatically generated in seconds more accurately from a more comprehensive set of sources.

After automatically generating a new map, the analyst sees an alarming pattern and decides to write a report. Of course, she can't use metadata as source information for a formal intelligence report, so she logs on to the data repository (to which she has access) to verify the pattern against the original reporting. However, she is denied access to several of the cables because they are stored in a restricted collection. Through official channels (referenced in the metadata) she requests access to the restricted collection, receives access, confirms the accuracy of the map, and produces an important report. In the past, she never would have seen the pattern in the first place because she wasn't aware of the reports in the restricted collection.

V. Ontologies for Information Sharing

The approach and claims described above for using semantic metadata to improve discovery hold true equally well for information sharing – one can simply view the sharing as a "push" of metadata across security boundaries whereas discovery is like a "pull". However, the use of ontologies and rich semantic metadata can enhance information sharing in a radical way.

Recall that our semantic metadata is represented in a standard language (RDF) that is well-defined and machine-interpretable, and that we can create rich ontologies in OWL

that are also machine-interpretable. For discovery, these ontologies enable semantic searching by abstracting the query concepts; to aid information sharing, ontologies can be used to automatically abstract or redact the semantic metadata itself.

Another feature of OWL is that it can encode inferences and other logical constructs which can then be automatically processed in software. Classification guides rules and policies can be represented in OWL, and the computer can automatically apply those rules and policies to semantic metadata. This allows the automatic redaction or abstraction classified metadata so that it conforms to the lower classification level. Semantic technologies that exist today enable us to automatically redact metadata for information sharing.

We can actually take this one step further. We can write a classification guide in OWL in such a way that a theorem prover can be used to *mathematically prove* that the redacted data does not violate any classification rules. Pellet is one example of a widely-used and well-respected open source theorem prover.

VI. Sample Scenario of Semantic Sharing

Local law enforcement has a need-to-know whenever FBI identifies an individual in the local community with terrorist connections. However, local law enforcement does not have the need-to-know (nor do they even care) the source or methods FBI used to obtain such information. In the past, whenever a new terrorist connection was established and documented, the entire data record was classified because it described how FBI obtained the information to create the connection. The only way local law enforcement came to know about the connection would be if an FBI agent read the entire report, distilled it down to an unclassified version, obtained the relevant approvals, and finally sent the information to local law enforcement.

In our approach, as each suspect interview summary report is generated, an RDF metadata file is generated containing names and known-terrorist connections. Again, this can be automatically generated using existing information extraction technology. This RDF metadata is automatically routed to local law enforcement via a fully accredited hardware/software guard device at the FBI network boundary. This guard reads the RDF, compares it to classification guides and policies encoded in OWL, and performs a logical redaction of the simple metadata facts. The redacted RDF metadata is then allowed to pass outside the FBI network and travels on to local law enforcement, where it can automatically be added to a database or reformatted into a textual message. Through official channels (referenced in the RDF), local law enforcement can request confirmation of the information at any later date.

VII. Conclusion

Discovering information in an environment with strict security constraints is a critical problem for the intelligence community. Physically-separate metadata can be used to overcome some of these problems. Metadata can have a lower level of classification than the data itself, and can reside on a different network than the data itself. In this way, more accessible metadata indexes can be created and exploited while fully maintaining the security of the source data. This means that even the most sensitive documents can be discoverable, and much of the information they contain can be exploited – even by analysts that have absolutely no access to the source documents themselves. Effective discovery and exploitation, however, depends on the availability of rich content metadata that is based on extensive ontologies.

There is an inherent conflict in the intelligence community between the responsibility to share information and the responsibility to protect it. This dilemma can be finessed by protecting *data* and sharing rich *metadata*. This approach can be implemented within the current strict security framework and will benefit significantly from the type of ontology work discussed at this conference.

Semantic technologies that exist today enable us to automatically convert documents to metadata, automatically redact that metadata to any security level, and automatically prove that the redaction is sound and complete.

Acknowledgment

Martin Thurn thanks Dr. Terry Patten for 20 years of friendship and mentoring, and for his pioneering work in computational linguistics, natural language processing, information extraction, and most recently, application of semantic technologies to the problem of secure information sharing.

References

[1] D. Nardi and R.J. Brachman, “An Introduction to Description Logics”, *The Description Logic Handbook*, Jan. 2003.

[2] F. Baader and W. Nutt, “Basic Description Logics”, *The Description Logic Handbook*, Jan. 2003.

[3] A. Uszok, J. Bradshaw, R. Jeffers, N. Suri, P. Hayes, M. Breedy, L. Bunch, M. Johnson, S. Kulkarni, and J. Lott, “KAoS Policy and Domain Services: Toward a Description-Logic Approach to Policy Representation, Decomfliction, and Enforcement”, *Proceedings*, pp 93-96 [IEEE 4th International Workshop on Policies for Distributed Systems and Networks, June 4-6 2003].

[4] J. Bradshaw, A. Uszok, R. Jeffers, N. Suri, P. Hayes, M. Burstein, A. Acquisti, B. Benyo, M. Breedy, M. Carvalho, D. Diller, M. Johnson, S. Kulkarni, J. Lott, M. Sierhuis, and R. Van Hoof, “Representation and Reasoning for DAML-Based Policy and Domain Services in KAoS and Nomads” [AAMAS ’03, July 14-18 2003, Melbourne Australia].

[5] N. Suri, J. Bradshaw, M. Burstein, A. Uszok, B. Benyo, M. Breedy, M. Carvalho, D. Diller, P. Groth, R. Jeffers, M. Johnson, S. Kulkarni, and J. Lott, “DAML-Based Policy Enforcement for Semantic Data Transformation and Filtering in Multi-agent Systems” [AAMAS ’03, July 14-18 2003, Melbourne Australia].

Semantic Wiki for Tactical Intelligence Applications: A Demonstration

Dan Reininger (dan@semandex.net), Jeff Mershon (jdm@semandex.net), Jef Armstrong (jef@semandex.net), Ray Kulberda (ray@semandex.net), Andrew Cohen (andrew@semandex.net), P. Robert Bullard (bob@semandex.net), David Ihrie (dihrie@semandex.net);
Semandex Networks Inc., 5 Independence Way, Suite 309, Princeton, NJ 08540 (609) 681-5382

***Abstract* — This paper demonstrates a *semantic wiki* application that helps tactical users manage data from diverse sources and multiple locations.**

***Index Terms* — semantic wiki, tactical intelligence, ontology, threat characterizations.**

INTRODUCTION

Military forces operating from Forward Operating Bases (FOBs) currently have inadequate means to collect and organize information in ways that can aid in rapid understanding of the evolving conditions and threats in an area. Until recently, only anecdotal evidence existed indicating that lots of structured and unstructured datasets were available "in the wild" but went underexploited by tactical users due to semantic and syntactic incompatibilities. [1].

We have conducted a study to quantitatively profile data sources of relevance to tactical intelligence operations in a counterinsurgency [2]. The viewpoint of our study is from the perspective of tactical ground military intelligence support to operations at the regiment, battalion or company level, particularly semi-independent task forces at these echelons. At this level, the intelligence element of a military organization often serves as the primary data repository and the principal data analysis cell that produces products to support decision-making. While various organizations assign specific information storage and analysis responsibilities to different sub-elements, the intelligence cell typically draws on a broad range of data sets and offers some level of support to virtually the full spectrum of counterinsurgency operations, from civil affairs (CA) and psychological operations (PSYOPS) to kinetic targeting.

The study compiled representative data sources used in theater during combat operations in Iraq and Afghanistan and identifies over 250 sources relevant to tactical operations of conventional and special operations forces engaged in a counterinsurgency. We identified more than 50 formats such as disparate spreadsheets or summarized in text reports that circulate in the field as e-mail attachments. These formats are easy to produce in the field but the information they contain is hard to exploit when it comes time to find quick answers to operational questions.

In this paper we present a demonstration of a *semantic wiki* application that helps tactical users manage these data sets "in the wild". The Semantic Wiki we have developed:

1. Integrates heterogeneous information coming from diverse sources and multiple locations;
2. Uses a flexible ontology that can be evolved by the user community to organize that information in a way that makes it easy for users to capture and understand how each piece of data connects. This makes it possible to analyze information interactions and dependencies;
3. Uses standard web technology such as REST Application Programming Interface to present and extract that information to other tools and systems.

We demonstrate how this semantic wiki application allows non-technical users to integrate and manage data sets in the field and answer contextual analytical questions from its data, without the assistance of specialized IT personnel.

SEMANTIC WIKI IMPLEMENTATION

A semantic wiki is one of the newly emerging Web 3.0 capabilities. Web 1.0 put information on-line by creating and connecting web pages with URLs and HTTP that computers could understand. Web 2.0 enabled people to easily publish information, leading to blogs, social networking and the "traditional" wiki. With the Web 3.0 *semantic wiki*, people and computers both use a common information structure, allowing each to optimize around the things they do best. Computers connect, monitor and process large quantities of data sources and information, while people are much better at observing, interpreting and connecting information. The common structure is a set of web pages representing people, events and other types of *entities*, with links connecting different types of pages according to an *ontology*. The structure of the ontology is accessible to computers and easily understandable by people.

The ontology is defined and maintained by the user community and drives the information organization. When

new information is *collected*, it is categorized and linked into the overall, evolving collection of linked pages (*semantic graph*) according to the structure provided by the ontology. Any type of entity may be represented in the ontology, from the general (person, facility, event, place, network) to the specific (financial withdrawal, graffiti, railroad siding). A new instance of one of these types (Person: John Doe) is created, structured and linked according to ontology. Thus John Doe will have person attributes such as height, gender, or ethnicity rather than event attributes such as type, location and time. The types of linkages that John Doe can have are also appropriate to a Person, such as father-of, employed-by, and similar connections. Compared to other semantic approaches, which utilize a fixed ontology, our approach recognizes and supports the notion that the relevant information structure has to vary over time to stay relevant. We allow this to be done by the community of users in the field to accurately track tactical understanding as situations evolve.

The Semantic Wiki is implemented using our commercial semantic engine, Tango. Like other semantic technologies, such as Twine [3], Zemanta [4] and Noovo [5], Tango is built on top of a relational database, and not an RDF store. The Tango meta-model may be thought of as being conceptually closer to an object/UML orientation.

Much of Tango — including the UI — is controlled through the schema. When it starts up, a schema, which is stored in a custom XML format, is read in from disk. A UML representation of the loaded schema is generated to disk. The schema can be updated while Tango is running, and the schema changes persisted to disk. These dynamically introduced schema changes are properly reapplied if Tango is ever restarted.

We recently added an OWL counterpart to the UML generator, and an OWL file is also generated at start up. The OWL and UML representations can also be generated on demand while the application is running. This is important because a goal of ours is to support dynamic lists of concept instances within the ontology. Users define temporal, spatial and semantic constraints for set membership into these lists and group them to create threat characterizations, and we want to be able to capture that user knowledge and make it available to other services via OWL. Instance data can be returned in a variety of formats, including KML, our own TORI XML [6] structure, and JSON, and we recently introduced support for RDF.

In an effort to keep the ontology OWL-DL compliant, certain features of our meta-model are not currently exported, including relationship certainty, and evidentiary associations.

The main driver for our support of OWL/RDF is to facilitate re-use of data by other emergent analytical tools and systems that can deal with OWL/RDF structured data [7]. As the integration and interoperability efforts with other systems continue within our on-going projects we expect to receive feedback on the ontology and its structure, and identify future user requirements from the program transitions we will be doing next year.

DEMONSTRATION

The demonstration is based on some of the current capabilities we have developed under the ONR Large Tactical Sensor Networks project in support of Marine Corps Intelligence needs [8]. This Semantic Wiki Implementation for Marines (SWIM) combines Marine Corps customized data connections, ontology and threat models with our commercial semantic engine, Tango. We use the ontology and threat models provided by tactical intelligence users with current counterinsurgency operations experience to detect and issue indications and warning alerts on enemy threats in progress based on those previous observations. The demonstration uses representative but unclassified IMINT, SIGINT and HUMINT data. IMINT data includes suspicious event data from processed UAV video with focus on activities of vehicles possibly involved in threat activities. HUMINT and SIGINT data includes representative formats and entity types from tactical and national data sets. As this data is collected, reports are created and the data is presented to software applications and analysts who semantically link it based on the ontology. Fig. 1 shows the customized SWIM data processing pyramid, with the raw data at the lower level and the tactical intelligence analyst interacting with the semantic wiki on top.

From a capabilities perspective, our demonstration focuses on three important specialized types of concepts supported within the Semantic Wiki: Smart Lists, Characterizations and Semantic Widgets:

Smart Lists — A Smart List is a set of pages that match any criteria, such as new people entering a controlled area (HUMINT), calls from a monitored phone (SIGINT) during a certain time of day, or vehicles behaving erratically in the vicinity of an operation (IMINT). The Semantic wiki keeps every list dynamically up-to-date and can be combined with alerts for a powerful mechanism to monitor virtually any change to data relevant to the mission.

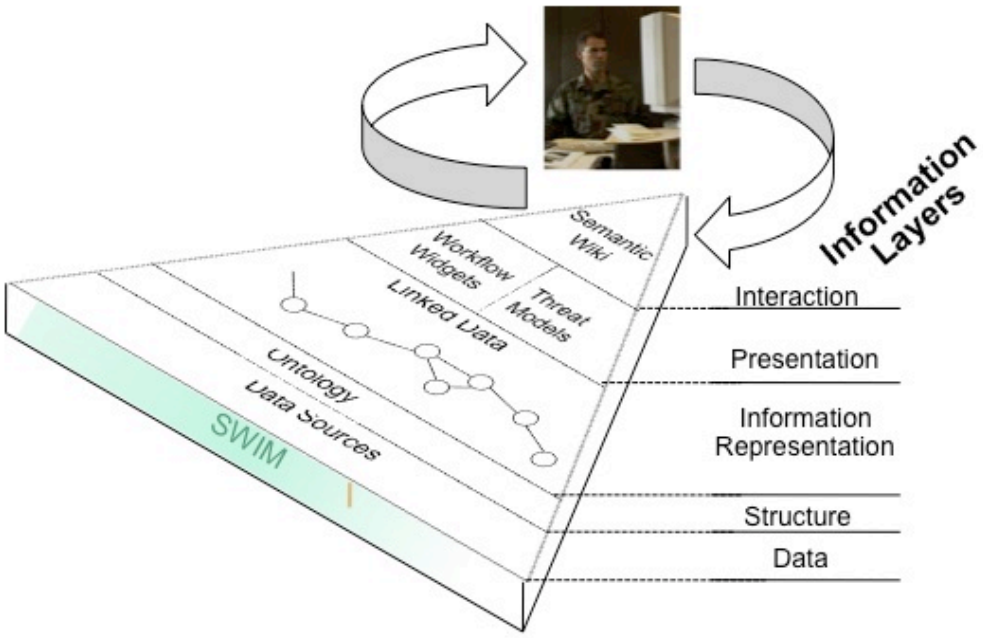


Fig. 1. Semantic Wiki for Tactical Intelligence Applications — Data Processing Pyramid

Characterizations — User-definable characterizations are the method tactical intelligence analyst can use to ask specific operational questions and determine if the information to answer them is available. A simple characterization might be used to mark as suspicious anyone who contacts a person on a watch list (represented as a Smart List). More complex characterizations can provide alert "clues" based on threat models of enemy Tactics, Techniques and Procedures (TTPs), or link specific devices to events based on complex associations.

One example of a characterization is the question: "Is this individual still at this location?", based on the operational need to verify information before a raid. Specific indicators could include SIGINT clues such as tipoff phone calls or a sudden absence of phone calls; IMINT might show vehicles leaving an area or people scattering through a field. HUMINT indications might involve an enemy operative seen buying food in a different town than expected. And, the characterization can combine these into both logical and temporal patterns: a flurry of calls followed by silence, with vehicles seen leaving an area shortly thereafter is a much stronger indicator than any of those detectable features in isolation.

A second example involves operational questions around whether an informant can be trusted. A call from a known bad guy may or may not be suspicious, since most informants associate with unsavory characters. However, a call from an unknown phone originating in the vicinity of a facility where suspicious activities have been observed represents a much more suspicious pattern.

We demonstrate specific examples of how characterizations help answer contextual questions such as: "Are these events a threat precursor, based on known tactics and trends?"

Semantic Widgets — The Semantic Wiki dashboard is home to widgets: mini-applications that let a tactical analyst perform common tasks and provide fast access to information. Because all the data on the Semantic Wiki is conformant to the ontology, the output of one widget can be linked to be the input to another, allowing users to create analytical pipes that capture best practices and serve to maintain knowledge continuity across rotations.

ACKNOWLEDGMENT

This material is based upon work supported by the Office of Naval Research under Contract No. N00014-07-C-0218 and DARPA under contract LM TT0705405 and 5R-44LM008474-03 (NIH). The views and findings expressed here do not necessarily reflect the views of these organizations.

REFERENCES

[1] Todd Hughes, "Toward Semantic Integration of Data in the Wild", Invited Talk, Ontology for the Intelligence Community (OIC-2007), November 28-29, 2007, Columbia, Maryland.

[2] P. Robert Bullard, "Sources of Tactical Data: A Study and Quantitative Profile", CUI project report prepared by Semandex for DARPA IXO, May 20, 2008.

[3] Twine is a product of Radar Networks, San Francisco,CA., United States, www.twine.com

[4] Zemanta is a product of Zemanta Ltd., London, UK, www.zemanta.com

[5] Noovo is a product of Noovo, LLC, Palo Alto, CA, United States, www.noovo.com

[6] *Tango Representational State Transfer (REST) API Guide*, Semandex Networks Inc., October, 2008.

[7] S. Stoutenburg, et.al., "Ontologies for Rapid Integration of Heterogeneous Data for Command, Control and Intelligence" In Proceedings of Ontology for the Intelligence Community (OIC-2007), Editor: Kathleen Stewart Hornsby, November 28-29, 2007, Columbia Maryland. http://ftp.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-299/Proceedings.pdf

[8] Martin Kruger, Large Tactical Sensors Networks, BAA 07-026, Industry Brief, May 17, 2007. http://www.onr.navy.mil/02/baa/docs/07-026_07_026_industry_briefing.pdf

Ontology of Evidence

Kathryn B. Laskey, David A. Schum, Paulo C. G. Costa
The Volgenau School of Information Technology and Engineering,
George Mason University, Fairfax, VA 22030 USA
[klaskey, dschum, pcosta]@ gmu.edu

Terry Janssen
Lockheed Martin, IS&GS/GSS,
Herndon, VA 20171 USA
terry.janssen@lmco.com

***Abstract*— Intelligence analysts rely on reports that are subject to many varieties of uncertainty, such as noise in sensors; deception or error by human sources; or cultural misunderstanding. To be effective, intelligence analysts must understand the relationship between reports, the events or situations reported upon, and the hypotheses of interest to which those events or situations are evidential. Computerized support for intelligence analysts must provide assistance for managing evidential reasoning. For this purpose, computational representations are needed for categories and relationships related to evidential reasoning, such as hypotheses, evidence, arguments, sources, and credibility. This paper describes some of the entities and relationships that belong in an ontology of evidence, and makes the case for the fundamental importance of a carefully engineered ontology of evidence to the enterprise of intelligence analysis.**

***Index Terms*— Evidence, probabilistic ontologies, intelligence analysis, inferential reasoning, source credibility**

I. Introduction

Evidential reasoning is fundamental to the practice of intelligence analysis. Much of an intelligence analyst's time is spent constructing complex chains of argument from evidence to conclusion, weighing the force of each argument and the credibility of its component sources, and arriving at overall judgments that, while falling short of certainty, provide useful inputs to decision makers. Reports that give rise to intelligence assessments are characterized by many varieties of uncertainty: noise in sensors; deception or error by human sources; poor understanding of situation or context. To be effective, intelligence analysts must understand the relationship between reports, the events or situations reported upon, and the hypotheses of interest to which those events or situations are evidential.

It follows that effective computerized support for intelligence analysts must support processes of evidential reasoning. For this purpose, computational representations are needed for categories and relationships related to evidential reasoning, such as hypotheses, evidence, sources, credibility, and the like.

Some have argued that computational representations of evidential categories and relationships, while necessary to intelligence analysis, do not belong in an ontology. Ontology, the argument goes, is the systematic study of existence: the categories of things that can exist and the relationships they can bear to one another. In the field of information systems, the term has come to mean the engineering discipline of constructing computational representations of various domains of application. By contrast, epistemology is the study of knowledge: how agents come to know about things that exist. The ontologies we construct, the argument goes, should be about what *is*, not what *might or might not be*, or what agents can reasonably *infer* from available evidence.

Computational support for intelligence analysts requires the ability to represent, store, and manipulate evidence, hypotheses, and arguments relating evidence to hypotheses. Such representations must be stored in a computational structure, which, for want of a better term, we might call an epistemological repository. Let us consider what such an epistemological repository might contain. It would represent concepts such as hypothesis, evidence, source, and report. It would contain relationships such as relevance of evidence to hypothesis, or the source-of relationship connecting a source with a report produced by the source. It would be quite natural to construct the representation using the languages and tools commonly applied in the discipline of ontological engineering. In other words, this epistemological repository would look rather like a domain ontology, where the domain being represented is epistemology – the field devoted to how we use evidence obtained from the world around us to arrive at knowledge about the world. The natural person to build this repository would be someone schooled in constructing such representations – that is, an ontological engineer. To call such a repository an ontology of evidence would hardly seem unreasonable.

In this paper, we argue for the fundamental importance of a carefully engineered ontology of evidence to the enterprise of intelligence analysis for the need for an ontology of evidence, and describe some of the entities and relationships that such an ontology would represent.

II. Evidence and Argument

Schum [1] has written a systematic treatise on evidence and its role in constructing arguments. All evidence, according to Schum, has three major credentials: relevance, credibility, and inferential force or weight. Relevance concerns the degree to which the evidence bears upon the hypothesis under consideration. Credibility means the degree to which the evidence is believable; whether or not the evidence is

Manuscript received November 1, 2008.

trustworthy. Inferential force concerns the strength of the relationship between evidence and hypothesis – the degree to which the evidence sways our belief in the hypothesis.

Evidence can come from diverse types of sources (e.g. physical sensors, human reports, direct tangible evidence such as objects or documents), each with different degrees of relevance, levels of credibility, and force.

As examples of the factors bearing the credibility of a source, evidence coming from physical sensors needs to be evaluated with respect to environmental conditions, distance from observer, and physical characteristics of the respective sensor. Human sensors, on the other hand, must be scrutinized with respect to opportunity, competence, and veridicality. Opportunity concerns whether the person was in a position to have observed the event or verified the fact. Competence concerns whether the source was capable of making the distinction in question. Veridicality concerns whether the source is telling the truth. Clearly, there may be complex chains of inference involved in ascertaining any of these factors influencing credibility. Approaches for dealing with the weight or strength of evidence include both qualitative and quantitative aspects of the reasoning process adopted to draw inferences from it (e.g. probability theory, logical reasoning, etc).

A vital (and too often overlooked) distinction to be made is the difference between an event and evidence that the event occurred, or between a fact and evidence that the fact obtains. Schum uses the notational device of an asterisk to make the distinction between event or fact *E* and evidence *E** relating to *E*. It is important to note that *E** does not entail *E*; the inference to *E* depends on the credibility of the source of *E**.

We do not always have the luxury of a direct report *E** on an event or fact *E* of interest. We may need to reason indirectly from a report *R** to an event or proposition *R* whose truth bears on the truth of *E*, and from there to *E* itself. Collections of interrelated propositions can be chained together into complex arguments. We often think of an argument as a linear chain from evidence through a collection of intermediate conclusions to a final conclusion. However, each link in such a chain must be justified. A judgment must be made that each antecedent in the chain is relevant to its consequent. The evidential force of each link must also be established. These judgments often require evidential reasoning in their own right. Schum uses the term *ancillary evidence* to refer to evidence about the nature and force of an evidential relationship. Intelligence analysts require support for keeping account of chains of argument and the ancillary evidence on which their force depends.

III. Probabilistic Treatments of Evidence

The past century has brought broad appreciation of the statistical regularities underlying the seeming complexity of physical, biological, psychological, and societal phenomena [2]. Computational advances are enabling automated and semi-automated support for many "knowledge tasks" once thought to be the exclusive province of human cognition. Intelligence analysts increasingly rely upon computerized systems that allow them to catalog, organize, and explore the implications of large collections of reports and other evidence. Quantitative measures of the strength of evidence are useful as a way to summarize and communicate the implications of large bodies of evidence. A natural candidate for such summarization, with a long and respected intellectual tradition behind it, is probability. Systematic deviations of intuitive human reasoning from the tenets of probability theory (*e.g.*, [3]) have been cited as justification for heuristic approaches to combining strength of evidence (e.g., [4]). Nevertheless, naturalistic human reasoning can usefully be treated as a computationally bounded approximation to a probabilistic norm (*c.f.*, [5], [6]). There is a robust literature on the use of probability and decision theory to support human inference and decision making, and to protect against errors that can occur in naïve human reasoning (e.g., [7], [8]). Furthermore, heuristic techniques introduced as cognitively natural ways to overcome perceived disadvantages of probability theory have been shown to admit a probabilistic interpretation (e.g., [9]). When the independence conditions justifying the probabilistic interpretation are met, such heuristic weighting factors can work well, but they can produce disastrous results when applied without regard to whether these conditions are met. There is no match for probability theory in its generality, logical coherence, and well-developed methodological base. For this reason, we focus on probability theory as a logically justified approach to combining numerical measures of evidential force.

We provide several examples to illustrate how probability can be used to represent and reason about credibility, to combine reports from different sources, and to handle subtleties such as dependence relationships that can stymie naïve heuristic weighting schemes. Our examples are deliberately kept simple to illustrate the key points. They are not intended to represent the full complexity of the evidential reasoning problems faced in real applications. Nevertheless, they illustrate the building blocks from which a more sophisticated reasoning capability can be constructed.

Figure 1 shows a Bayesian network that illustrates the combination of three independent pieces of evidence regarding the whereabouts of Osama bin Laden. Prior to receiving the reports, the probability is 3% that he is in Kandahar. After receiving the first report, the chance increases to 11%. After a

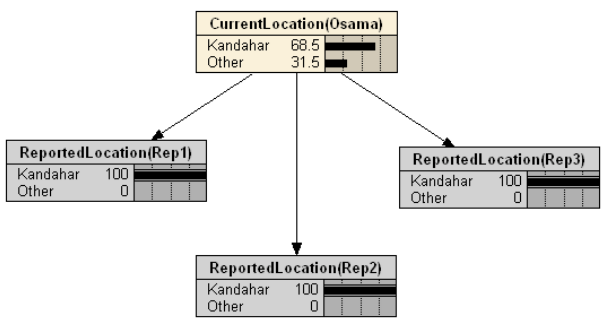


Figure 1: Three Independent Reports Increase Probability of Hypothesis from 3% to 69%

second report, the probability is 35%; the third report brings the probability to 69%. The figure shows the situation after the third report has been received. The top rectangle represents hypotheses about bin Laden's location and their probabilities (Kandahar at 69%; Other at 31%). The three reports are shown below the location hypotheses. The gray color indicates that they have been specified as evidence, with 100% probability assigned to the actual reported location. Figure 2 extends this example to explicitly represent report credibility. The figure now shows credibility hypotheses (low, moderate and high) for the three reports. If we had specified no evidence about the credibility values, the results would have been the same as Figure 1. But if we specify that the credibility of the third report is low, then the probability decreases to 55% that bin Laden is in Kandahar. That is, lowering the credibility of a report decreases its evidential force, resulting in less change in belief when the report is received.

Our final example illustrates an issue not easily accounted for by heuristic methods for assigning and combining evidential weights. Suppose we discover that two of the reports, which we had originally treated as independent, may have actually come from the same informant. We can treat this case by explicitly representing a hypothesis for whether the reports came from the same source. In Figure 3*a*, we indicate that the sources of the two reports are different. In this case, they can be treated as independent evidence items, and the resulting belief in bin Laden's location is the same as in Figure 1. However, if we specify that the sources are the same (Figure 3*b*), the probability that bin Laden is in Kandahar is reduced to 35%, the same as if we had received only two independent reports. The structural assumptions (the independence relationships represented in the graphs) together with the numerical probability values ensure that subtleties such as source credibility and common sources are properly accounted for in evidential reasoning.

Additional treatments of probabilistic representations of relevance and credibility in evidential reasoning can be found in [10] and [11].

IV. A Probabilistic Ontology of Evidence and Inferential Reasoning

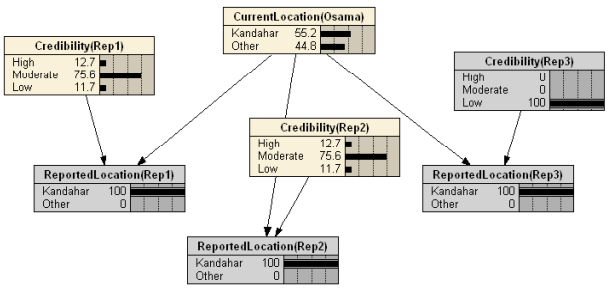
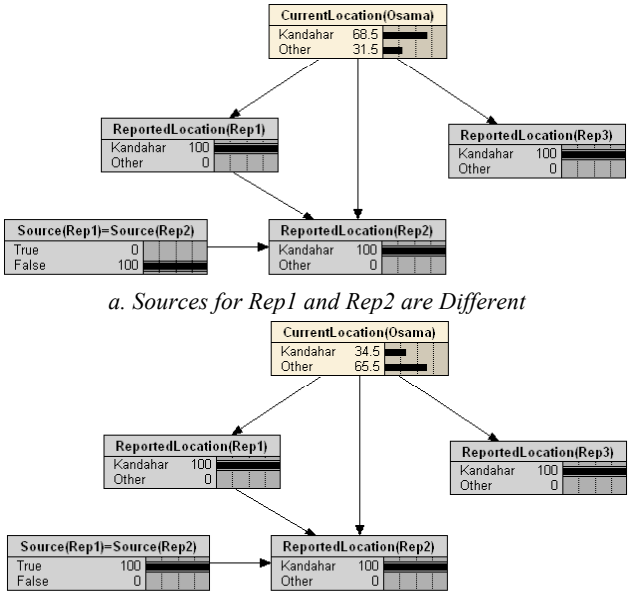


Figure 2: Low Credibility Reduces Force of Report

The above concepts pertain to the use of evidence as an informational asset and to the inferential process that transforms it into knowledge. This is clearly a multi-disciplinary subject. Practitioners from many disciplines can profit from a formalization of the discipline of evidential reasoning. Due to its heavy dependence on evidence in almost every aspect of its operations, the domain of intelligence analysis would be a prime beneficiary of an ontology of evidence. Benefits of an ontology of evidence include a common, shared vocabulary for important features and relationships that occur across different applications of evidential reasoning, as well as the ability to share information among diverse systems.



a. Sources for Rep1 and Rep2 are Different

b. Sources for Rep1 and Rep2 are the Same

Figure 3: Common Source Reduces Force of Report

Despite considerable diversity and individual variation in the conduct of investigation and analysis, there are fundamental common structures and processes. Examples include assessing the credibility and relevance of individual items or of masses of evidence, or constructing reasoning chains to connect evidence to hypothesis. A formal representation of evidence and evidential relationships provides the obvious benefit of allowing analysts to query a knowledge base not just for conclusions (e.g., "Where is Osama bin Laden?"), but also for the evidence on which the conclusions are based (e.g., "What is the evidence that bin Laden is in Kandahar?") Analysts can reason about the relevance of evidence to hypotheses, the credibility of sources, errors that may be common to several evidential reasoning chains, and other subtleties of evidential reasoning.

There has been an increasing emphasis in recent years in sharing knowledge among intelligence applications. An ontology of evidence and inferential reasoning is a first step in that direction. Ontologies provide shared representations of the entities and relationships characterizing a domain, into which vocabularies of different systems can be mapped so to provide interoperability among them. Techniques for making semantic information explicit and computationally accessible

are key to effective exploitation of evidence from diverse sources, with distinct grades of credibility and relevance. Shared formal semantics enables systems with different internal representations to exchange information, and provides a means to enforce business rules such as access controls for security.

However, traditional ontologies do not provide a principled means to ensure semantic consistency with respect to issues of uncertainty related to credibility of sources, relevance of evidence, and other aspects of the evidential reasoning process. Because uncertainty is a fundamental aspect of evidential reasoning, this is a serious deficiency.

When faced with the challenge of representing uncertainty in an ontology, the natural tendency is to introduce a means to annotate property values with information regarding their level of confidence. This approach addresses only part of the information that needs to be represented in a full ontology of evidence. To understand why more is needed, consider the example from Section II above, in which evidence from several sources is combined to draw an inference about the current location of Osama bin Laden. We saw that the inferential force of each report depended not only on that report's credibility, but also on whether the information from which it was derived overlapped with the information on which another report was derived. In other words, we need to represent not just a single credibility number, but information about how that credibility was derived. An assessment from source x, in order to be used in conjunction with evidence coming from other sources would not only state that (say) "with 75% probability, Osama bin Laden is in Kandahar." To be part of a comprehensive probabilistic model capable of performing sophisticated evidential reasoning, such a statement would have to include the supporting evidence and how its credibility affects the overall assessment. A simple example would be "with 75% probability, *given* reports that his physician was spotted in a local market (evidence E1) and that a radio communication regarding his whereabouts was intercepted (evidence E2)," accompanied by information clarifying how this number changes as the credibility of E1 and E2 varies. Further, as new evidence accrues, a sophisticated evidential reasoning system must be capable of capturing the impact of additional evidence on the body of evidence being analyzed. As an example, if a source were found to be a double agent, the credibilities of all reports from that agent would need to be called into question. A system that relies on or can represent only numerical weights of individual arguments cannot cope with the complexity and dynamic aspect of real world multi-source evidential reasoning.

In short, annotating a standard ontology with numerical probabilities is not sufficient, as too much information is lost due to the lack of a good representational scheme that captures structural constraints and dependencies among probabilities. Over the past several decades, semantically rich and computationally efficient formalisms have emerged for representing and reasoning with probabilistic knowledge (e.g., [12]). A true probabilistic ontology must be capable of properly representing the nuances these more expressive languages were designed to handle. We have argued elsewhere (e.g. [5]) that for domains characterized by uncertainty, probabilistic ontologies ([13], [14]) provide a principled means to represent the structural and numerical aspects of evidential reasoning. Indeed, many researchers have pointed out the importance of structural information in probabilistic models (e.g. [15], [16]), and it is well known that some questions about evidence can be answered entirely in structural terms ([1], page 271). Shafer ([17], pages 5-9) argues that probability is more about structure than it is about numbers. Numerical probabilities enable quantitative assessment of the force of evidence, which depends on the strength of relevance and credibility arguments. Exploring the details of probabilistic ontologies is not in the scope of this work, but the interested reader is referred to http://www.pr-owl.org.

Finally, apart from the advantages of knowledge sharing tools to the Intelligence Analysis domain, it is important to foresee the institutional and cultural implications of systematizing and standardizing vocabulary and semantics of evidential reasoning. The very difficulties an effective information-sharing scheme is meant to overcome can become obstacles to its widespread adoption. Given the nature of the field, with highly personalized approaches to analysis, a knowledge tool may encounter resistance if it is perceived as threatening deeply ingrained processes. Yet, the increasing demands within the Intelligence community for effective exchange create an opportunity for developing standardized representations and approaches. This is an important and difficult issue. A probabilistic ontology of evidence is a promising first step to provide a structure for knowledge sharing that is sufficiently flexible to address the demands of the multiple approaches currently used to handle evidential reasoning.

V. Summary and Conclusions

After identifying some concepts regarding the process of transforming masses of evidence into knowledge, we explored the need for formal representations of evidential processes as a means to provide cross-fertilization among domains that depend on processes of evidential reasoning. Among these, intelligence analysis is paradigmatic. We proposed a probabilistic ontology of evidence as a key enabler of this vision. Implementation of this concept must be cognizant of institutional and cultural barriers. In conclusion, we argue that the benefits of effective evidential reasoning and knowledge sharing tools far outpace the difficulties in implementing them.

Acknowledgements

The authors gratefully acknowledge helpful comments from anonymous reviewers of an earlier draft of this paper.

References

[1] D. A. Schum, *Evidential Foundations of Probabilistic Reasoning*. New York: John Wiley & Sons, Inc., 1994.

[2] G. Gigerenzer, Z. Swijtink, T. Porter, L. Daston, J. Beatty, and L. Kruger. *The Empire of Chance: How Probability Changed Science and Everyday Life. Cambridge, Cambridge University Press, 1990*.
[3] D. Kahneman, P. Slovic, and A. Tversky, eds., *Judgement Under Uncertainty: Heuristics and Biases*. Cambridge: Cambridge Univ. Press, 1982.
[4] S. Bringsjord, J. Taylor, A. Shilliday, M. Clark and K. Arkoudas. "Slate: An Argument-Centered Intelligent Assistant to Human Reasoners," in F. Grasso, N. Green, R. Kibble and C. Reed (*eds.*) *Proceedings of the 8th International Workshop on Computational Models of Natural Argument* (*CMNA 8*), Patras, Greece, 2008. http://kryten.mm.rpi.edu/Bringsjord_etal_Slate_cmna_crc_061708.pdf.
[5] L. Martignon and K. B. Laskey. "Taming Wilder Demons: Bayesian Benchmarks for Fast and Frugal Heuristics." In *Simple Heuristics that Make us Smart.* The ABC Group, Oxford University Press, 1999.
[6] J. R. Anderson and M. Matessa. "Explorations of an Incremental, Bayesian Algorithm for Categorization." *Machine Learning* 9(4), 275-308, 1992.
[7] D. Von Winterfeld and W. Edwards. *Decision Analysis and Behavioral Research*. Cambridge, U.K.: University Press, 1986.
[8] W. Edwards, R. F. Miles, Jr., and D. von Winterfeldt, *Advances in Decision Analysis: From Foundations to Applications*. Cambridge, U.K.: University Press, 2007.
[9] D.E. Heckerman. "Probabilistic interpretations for MYCIN's certainty factors". In J. Lemmer and L. Kanal, editors, *Uncertainty in Artificial Intelligence Vol 1*, pages 167-196, Amsterdam: Elsevier, 1986.
[10] S. Mahoney, D. Buede, and J. Tatman. "Patterns of Report Relevance." *Proceedings of the Third Annual Bayesian Modeling Applications Workshop*, 2005. http://www.intel.com/research/events/bayesian2005/docs/Mahoney-ReportRelevance.pdf.
[11] E. Wright and K. Laskey. "Credibility Models for Multi-Source Fusion." *Proc. 9th International Conf. on Information Fusion*, 2006. http://ite.gmu.edu/~klaskey/papers/Wright_Laskey_Credibility.pdf
[12] K. B. Laskey, "MEBN: A Language for First-Order Bayesian Knowledge Bases" *Artificial Intelligence*, 172(2-3), 200
[13] K. B. Laskey, P. C. G. Costa, and T. Jensen, "Probabilistic Ontologies for Knowledge Fusion," in *Proc. 11th International Conf. on Information Fusion*, Cologne, Germany, 2008.
[14] P. C. G. Costa, "Bayesian Semantics for the Semantic Web," PhD Dissertation, Dept. of Sys. Eng. and Op. Res., George Mason Univ. 315p, Fairfax, VA, USA, 2005.
[15] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo, CA, USA: Morgan Kaufmann Publishers, 1988.
[16] J. B. Kadane, and D. A. Schum, *A Probabilistic Analysis of the Sacco and Vanzetti Evidence*. New York: John Wiley & Sons, 1996.
[17] G. Shafer, "Combining AI and OR," University of Kansas School of Business, Working Paper No. 195, April 1988.

The Ontology of Systems

LTC Kristo S. Miettinen, North Central Information Operations Center, Coraopolis PA 15108
email: kristo.miettinen@us.army.mil; phone: (585) 269-5592

***Abstract*— Systems analysis comprehends systems in terms of an ontology that relates any system, its elements, and its environment in terms of their functional, structural, and behavioral relations. At the heart of systems ontology is "design", the combination of two interactive loops: one loop relating the system to its environment, the other loop relating the system to its parts. For systems analysis, *e.g.* intelligence analysis of remotely sensed facilities in denied territory, these loops consider structure, function, and process in the context of environment to develop information (what), knowledge (how), and understanding (why) of the system and elements being studied. This exposition presents the interactive loops of design in systems ontology, treating analysis of Soviet national missile defenses as an example of successful application of systems ontology.**

***Index Terms*—Ballistic missile defense, cold war, intelligence analysis, ontology, systems methodology**

I. Introduction to Systems

The analysis of design in systems ontology leans heavily on the modern concept of a system, especially the definitions of "system" due to Bertalanffy and Ackoff.

Bertalanffy (1969, pp. 55-56) defined systems as follows: "A system can be defined as a set of elements standing in interrelations. Interrelation means that elements, *p*, stand in relations, *R*, so that the behavior of an element *p* in *R* is different from its behavior in another relation, *R*'. If the behaviors in *R* and *R*' are not different, there is no interaction, and the elements behave independently with respect to the relations *R* and *R*'."

Ackoff's subsequent restatement suppresses explicit mention of the relations among elements (1981, pp. 15-16; see also 1972, 1974): "A system is a set of two or more elements that satisfies the following three conditions. (1) The behavior of each element has an effect on the behavior of the whole… (2) The behavior of the elements and their effects on the whole are interdependent… the way each element behaves and the way it affects the whole depends on how at least one other element behaves… (3) However subgroups of the elements are formed, each has an effect on the behavior of the whole and none has an independent effect on it."

Ackoff's and Bertalanffy's definitions are compatible, but Ackoff's definition avoids explicitly introducing the relations *R* as explaining differences in behavior of *p*, leaving the interdependencies unexplained. This leads to abandonment of reductionism, which is characteristic of systems thinking. Bertalanffy's definition is important for illuminating why it is that systems have the kinds of irreducibility that are made implicit in Ackoff's definition: it is the relations of the elements to the system and to one another that give the elements their system-dependent properties on the one hand, and the system its emergent properties on the other. In a nested system-of-systems, Bertalanffy's definition helps to explain what Ackoff's definition asserts, particularly the distinction between functions and purposes.

Ackoff concludes from his definition that every element of a system has essential properties that belong to it only by virtue of its being an element in the system, and also that every system has essential properties that belong to none of its elements, either individually or in aggregation. Systems analysis exploits these two ontological conclusions to locate function among the essential properties of an element that it has only in virtue of its being in a system, and to locate the purpose being served by a function among the essential properties of the system that belong to none of its elements. These are ontological razors for winnowing candidate functions and candidate purposes in systems analysis.

II. Design in Systems Ontology

A. Definitions of "Design"

"Design" as a verb is a rational or economic act of requirements transformation. In engineering, requirements are transformed through many stages: from user requirements to system operational requirements through conceptual design, from system operational requirements to element functional requirements through preliminary design, and from element functional requirements to production requirements (specifications, schematics *etc.*) through detailed design.

Engineering design develops efficient applications of resources to satisfy needs. The economic or rational aspect of design, combined with functional allocation in design, distinguishes designs from other arrangements of parts for a collective purpose by the economy of means to an end so that nothing is invoked other than what is functionally justified.

In keeping with the definition of designing as an inherently rational or economic activity, "design" as a noun is the rationale for the requirements transformations understood in the structural, functional, and process relationships between the system, its environment, and its parts or elements.

The outputs of engineering design are product and production specifications in sufficient detail to eliminate interpretation in the production process, rather than any cognitive basis for requirements transformations. "Design" as a noun is not the outcome of "design" as a verb; schematics and specifications are not designs but rather summaries of design sufficient for production. That there is more to a design than is captured in schematics and specifications is evident when designs are protected as proprietary, or delivered from a vendor to a customer in cases of contracting design, or

archived for future use. What is included in an archived design, or in a design delivered under a standard contract, or is protected as proprietary when safeguarding designs, includes performance analyses, trade studies, and the development of those alternative system concepts that were evaluated but not, in the end, chosen for production. What is included in the object called a "design" is the entire rationale for the requirements transformations specified in the design process.

Complementing the distinction between the noun "design" and the products of the activity called "design" is the distinction between comprehending the design of something, *e.g.* a surface-to-air (SAM) missile complex, and apprehending the prior occurrence of an act of design; to acknowledge the design of something is only to judge that the relationships between elements and their capabilities at successive hierarchical levels of nested systems are rational or economical. The rationality of design is ontological (specific to the relations among elements), and specifically an analytical rationality (comprehensibility) rather than an etiological rationality. The cause of rationality in design is not the rationality of any designer, but rather the environmental, technical, and economic constraints within which the system is realized. Failing to appreciate this distinction, by insisting on the rationality of causal agents, leads to a characteristic failure of analysis discussed in section IV.b below.

B. Function and Purpose

Functions are not arbitrary properties of system elements; they must be among those properties that are essential to the element as an element, in light of the essence of systems (the interdependence of behaviors of systems and elements). This distinguishes the intercept function of an anti-ballistic-missile (ABM) in a national missile defense (NMD) system from its non-functional trans-sonic boom. Claiming that the sonic boom is non-functional is to claim that there is no system that can be fully analyzed in terms of the ontology of systems, whose design leads to the ascription of any function or purpose to the sonic boom of an ABM. Any well-formed system comprising the ABM will avoid such ascriptions; any putative system whose analysis entails such ascriptions for the sonic boom of the ABM will fail to converge on a design, as discussed in section IV.a below.

Similarly, the ends served by the functions of the elements (*i.e.* the purposes of the system) are among those properties of the whole system that are essential to the system as a system. For instance, if a function of a search radar in a ballistic-missile defense (BMD) system is cueing targeting radars, and if re-entry vehicle (RV) destruction is the purpose served by that function, then this entails (1) that RV destruction is an emergent property of the BMD system, (2) that the search radar is an element of that system, and (3) that the search radar does not cue targeting radars apart from its belonging to a BMD system.

Functions and purposes are separated by one hierarchical layer in a nested system-of-systems, but purposes at one level are not the same as functions at the next, except by coincidence. So, for instance, if a function of a search radar in a BMD system is to cue targeting radars, and if RV destruction is a purpose of the BMD system, then that does not entail that cueing targeting radars is a purpose of the search radar (*i.e.* an end served by functions of elements of the radar such as the antenna, transceiver, beam-former, power supply *etc.*), nor does it entail that RV destruction is a function of the BMD system in the national defense architecture. Both of these hypotheses are, in practice, reliable starting points for iterative systems analysis, but they are not necessary consequences of search radar function or BMD system purpose.

C. Analogy of Engineering and Analysis

Design in systems ontology is the combination of two interactive loops, one addressing the relationship of the system to its environment, the other addressing the relationship of the system to its parts. In systems engineering, the two loops are called preliminary design and detailed design, while in systems analysis they are called expansion and reduction. Analysis mirrors the structure of engineering even when analysis is conducted without access to system designers, because of the ontological commitments of scientific realism regarding systems: systems being what they are, they must be analyzed (and designed, if designed at all) in terms of the underlying reality of systems, which involves the two loops of design.

Viewed from the perspective of any arbitrary element Y_b (a functionally specified constituent of a system X), preliminary design of X and expansion of Y_b both determine the function of Y_b as a contribution to the comprising whole X, while detailed design of X and reduction of Y_b determine the structure of Y_b and how it works.

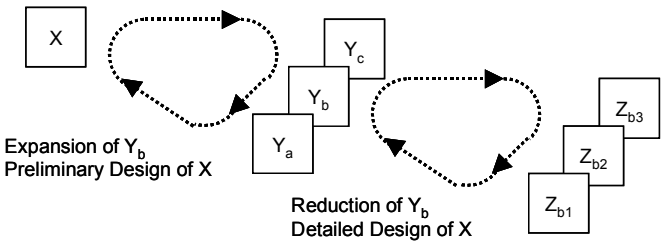


Fig. 1. Nested design loops of systems methodology

The relationship between the systems engineering design of X and the systems analysis of one of its elements Y_b is illustrated in figure 1 above for a system X consisting of elements Y_i, each of which in turn consists of sub-elements Z_{ij}. The nesting can continue indefinitely in both directions: X can be an element of some other larger comprising super-system W, and each Z_{ij} can in turn be an object of either design or analysis, so that the preliminary design of X may also be part of the detailed design of W, and the detailed design of X may comprise the preliminary designs of the Y_i and the conceptual designs of the Z_{ij}.

Figure 1 offers an opportunity to distinguish functions from purposes using Bertalanffy's definition of a system. Consider the relations R_{zb} found among the elements Z_{bj} in the reduction of Y_b, and the relations R_y found among the elements Y_i in the

expansion of Y_b. The functions of the elements Z_{bj} serve purposes inherent in Y_b, and the function of Y_b serves a purpose inherent in X. The question to consider is whether the function of Y_b and the purposes inherent in Y_b are identical. Systems ontology answers "no, except by coincidence", because the function of Y_b is among those properties that Y_b has in virtue of relations R_y rather than any alternative R'_y, while the purposes inherent in Y_b are among those properties that Y_b has in virtue of relations R_{zb} rather than any alternative R'_{zb}. The function of Y_b and the purposes inherent in Y_b are both at the same hierarchical level (*i.e.* they are both in Y_b), but they are determined by distinct relations R_y and R_{zb} at adjacent hierarchical levels, and therefore they are not identical, though they may correspond to one another.

D. Relating Structure, Function, and Process

As summarized by Gharajedaghi (1999, pp. 112-113), the design approach to systems analysis iteratively examines structure, function, and process to develop understanding in terms of design. In the ontology of systems, process and structure co-produce function in the context of environment, so that inquiry necessarily becomes iterative because of the cyclic graph ontology of systems. Structure, function, and process are each co-produced by the others, as well as co-producing each other. Therefore, developing new understanding of each necessarily modifies understanding of the others, in a converging sequence of mutual dependence.

The producer/product relationship is Singer's framework for explanation in the world of complex objects without sufficient causation. In this framework, producers are necessary but not sufficient for their products, in the manner of acorns being necessary but not sufficient for oak trees. Singer (1924, 1959) uses the producer/product relationship to develop a pragmatic theory of choice, purpose, and free will, and extends the relationship in various ways to account for reproducers, co-producers, potential producers, and other analogues for biological and ecological systems. Following Churchman (1971, 1979), systems analysis uses the same ontological framework for developing an objective theory of function and purpose. Function is a joint product of structure and process in the context of a purpose inherent in the essential characteristics of a comprising system.

III. Analysis of Soviet National Missile Defense

Sparked by a 1953 joint letter of seven Marshals recommending national missile defense (NMD), the Soviet Politburo approved their first plan for NMD in 1954. This plan, implemented in stages, adapted the SA-1 SAM in an ABM role, and developed the Sary Shagan missile test range as well as the Triad targeting radar and the Hen House phased-array radar. Among the achievements of this first Soviet NMD program was the successful 1961 interception of an SS-4 warhead by a modified SA-1 interceptor (called V-1000) at an altitude of 25 kilometers over Sary Shagan, using a conventional explosive warhead. This interception integrated all of the elements of NMD, with a Hen House radar initially acquiring the target at a range in excess of 1000 kilometers and passing targeting data to Triad radars and the interceptor launch site (Lee, 1997).

Following this successful test, operational deployment of missile defense systems began in 1962-63, with simultaneous construction of the Moscow zonal missile defense system (with its characteristic Dog House and Pillbox radars), and the Soviet national BMD system, with its Hen House and Pechora-class large phased array radars (LPAR), most famously the LPAR at Krasnoyarsk.

American intelligence analysis of Soviet missile defense development could only rely on external observations of various kinds, such as operating frequencies and pulse durations collected from Soviet radars, observation of tests at Sary Shagan, and overhead photographs of missile installations. Analyses of this evidence were based on the ontology of systems. During the mid-1960s, while systems analysis of Soviet missile defense failed to understand the significance of many tests conducted at Sary Shagan or the relationship between the Hen House radar network and the Moscow missile defense network, US national intelligence estimates (NIE) nonetheless correctly determined that the Soviets were deploying NMD. These assessments were ultimately challenged in the late 1960s as the USA and the Soviet Union began negotiating what would become the 1972 ABM treaty, and the diplomatic community imposed a change in the nature of evidence required for those claiming that the Soviets had deployed NMD (Lee, 1997), since Soviet authorities denied deploying NMD and the treaty forbade it.

The 1960s-era systems analyses of Soviet NMD proceeded from fixing observed Soviet interceptor limitations (especially their slow speed, about 2 kilometers per second, and their languid initial acceleration) as technological design constraints under the ontological razor of rational economy of means, and concluding from this that any Soviet NMD would have to operate in battle management mode rather than point defense or perimeter defense mode. With this in mind, the question of whether the Soviets were deploying NMD was analytically reduced to four core questions, all potentially answerable from available intelligence methods:

[1] Were the SA-5 and the SA-10 interceptors dual-function SAM/ABMs?
[2] Were the Hen House and Pechora-class LPAR radars passing target tracking data to missile defenses?
[3] Was there a central ABM command authority with a command, control, and communications (C3) system?
[4] Did the SAM/ABM missiles have nuclear warheads?

All NIE participants agreed that if the answers to these questions were "yes" (and they were), then the Soviets were deploying NMD (Lee, 1997).

Several things are noteworthy about these questions. An overarching feature of systems analysis in this case was that inferences of purpose (NMD) and function (ABM) were being made without any testimony of the system's designers (which would become available in the 1990s, corroborating the 1960s-

era analysis). The inference was based only on capabilities that NMD systems should have that air defense systems would not, given rational and economic relationships among system elements under the constraints of prevailing Soviet technology. This is consistent with function and purpose being matters of ontology, matters of the nature and relationships among things as they are, rather than being dependent upon the intentions of causal agents, or otherwise contingent upon causal history.

All four core questions address issues of function or purpose through analysis of relations. For instance, the distinction between a SAM and an ABM depends on how the interceptor is integrated with its associated radars, specifically with the function that the interceptors and radars co-produce. Similarly, whether the SA-5 and SA-10 interceptor missiles had nuclear warheads depended on the proximity of nuclear storage facilities to the missile launch sites.

This case also illustrates a characteristic of systems analysis of artificial systems: an ontological analysis often develops functional ascriptions which contradict the claims of authorities, a characteristic amply documented in Ackoff's many writings on his analyses of government and UN agencies, corporations, charities, *etc*.

IV. Failures of the Analysis of Soviet NMD

A. Failures of Systems Analysis

The various failures of systems analysis of Soviet NMD described by Lee are instructive. For instance, the failure to rationalize the sequence of tests at Sary Shagan and the failure to understand the relationship between the Hen House and Dog House radars (in fact there was none) were both due to the same mistake, made by analysts at the beginning of Soviet missile defense deployment in the early 1960s and corrected a few years later: what was in fact two separate systems, with distinct interceptor models, distinct radar models, and distinct areas of responsibility (Moscow on the one hand and the Soviet Union on the other) was analyzed as though it was all one system whose area of responsibility was a topic of contention. The problem of correct delimitation of a system in systems analysis remains difficult, and inspiration remains part of the solution (Zandi, 2000; Churchman, 1971, 1979).

It is important to note in the case of Soviet NMD that the consequence of initial failure to properly distinguish and delimit the systems was not a conclusive faulty analysis, but rather it was failure of the ontological analysis to converge. This is characteristic of ontological systems analysis, that rather than confidently reaching erroneous conclusions from false premises, it dissolves into a muddle when its underlying premises are incorrect.

B. Other Failures of Analysis

Other failures after the analysis of the 1960s reflect departures from analysis methods of systems ontology, rather than failure of systems analysis to understand Soviet NMD. For instance, the mistaken projection by western experts of mutually assured destruction (MAD, with its implicit disavowal of NMD), upon the Soviet leadership as the Soviet national nuclear strategy stemmed from the non-systems-ontology assumption of rationality on the part of system designers (as "rational" nuclear policy was then understood in the west), rather than the weaker systems ontology assumption of rationality of design relations among elements of a system. This kind of strong assumption may not be an error in other fields (*e.g.* it is a core assumption of the diplomatic theory of *realpolitik*), but it is unwarranted in systems analysis, and in this specific case it turned out to be materially false.

A related error committed in mis-analyzing Soviet NMD was the inference from high presumed cost and low presumed effectiveness of NMD to the conclusion that the Soviets weren't deploying NMD, because doing so would be uneconomical, or because NMD just wouldn't work. This is an example of misplacing the economy inherent in systems from the relationship of elements (an ontological matter) to the decisions and motives of owners, or making the unwarranted assumption that a systems must work to have designs. For these and other reasons systems analysis emphasizes understanding the design without attempting to understand either the designer or the beneficiary, without even assuming that any designer or beneficiary exists. Only the manifest relationships of system elements are understood rationally; understanding the designer or the motives that lead to existence of the design are not part of the ontological analysis.

V. Conclusion

Design in systems ontology consists of two interactive loops, one relating the design object to its environment, the other relating the design object and its elements. The analysis of any system's design develops information, knowledge, and understanding of the system and its elements presuming that rational and economic relations among system elements determine structure, function, and process in the context of environment. This method is capable of discerning functions and purposes that are not apparent from structures alone, or from analogy with structures of known function.

References

[1] Ackoff, R. L. and Emery, F. E., 1972, *On Purposeful Systems*, Aldine-Atherton Press, Chicago.

[2] Ackoff, R. L., 1974, *Redesigning the Future*, Wiley, New York.

[3] Ackoff, R. L., 1981, *Creating the Corporate Future*, Wiley, New York.

[4] Bertalanffy, L. von, 1969, *General Systems Theory*, Braziller, New York.

[5] Churchman, C. W., 1971, *The Design of Inquiring Systems*, Basic Books, New York.

[6] Churchman, C. W., 1979, *The Systems Approach and its Enemies*, Basic Books, New York.

[7] Gharajedaghi, J., 1999, *Systems Thinking*, Heinemann, Boston.

[8] Lee, W. T., 1997, *The ABM Treaty Charade: A Study in Elite Illusion and Delusion*, Council for Social and Economic Studies, Washington.

[9] Singer, E. A., 1924, *Mind as Behavior*, Adams Press, Columbus, OH.

[10] Singer, E. A., 1959, *Experience and Reflection*, C. W.Churchman, ed., University of Pennsylvania Press, Philadelphia.

[11] Zandi, Iraj, 2000, "Science and engineering in the age of systems", presented at "What is Systems Engineering?", Intn. Council on Syst. Engr. (INCOSE), Sept 19 2000.

Ontology-based technologies — Technology transfer from bioinformatics?

Fabian Neuhaus, NIST

I. Introduction

In the call for paper for OIC 2008 the description of the conference contains the following optimistic outlook:

> New approaches are required to enable greater flexibility, precision, timeliness and automation of analysis in response to rapidly evolving threats. Ontology-based technology as applied in the areas such as bioinformatics has demonstrated the possibility of gains along all of these dimensions. The time is ripe to extend these gains to other spheres.

Ontology-based technologies clearly offer great potential for the intelligence community. In this paper I will discuss whether the intelligence community could adopt technologies that have been proven successful in bioinformatics. For this purpose we have to consider how biologists apply these technologies and how their needs differ from the needs of the intelligence community.

II. Kinds of knowledge

Biologists have been very successful at representing biological knowledge in a machine-readable form with the help of ontology-based technologies. However, we should not take for granted that the technologies that work for biologists would be appropriate for the intelligence community, because the kind of knowledge gathered by the intelligence community differs in important respects from biological knowledge. While the intelligence community is interested in individual people and organizations, biologists are producing scientific knowledge that consist of more or less general laws. Even in cases where biologists use terms from ontologies to describe the results of individual experiments, these results are formulated as laws; for example, laws like 'if a fruit fly has the mutation x, then the fly will have red eyes'. Biologists are only interested in the properties of individual animals or plants if these properties might provide evidence for or against a general hypothesis. For this reason, it is usually not necessary, and often not even possible, for biologists to keep track of the individual entities that they are experimenting with; e.g., no biologist would care to uniquely identify the individual fruit flies of a population, let alone the individual RNA molecules in a particular sample. In contrast, for the intelligence community it is crucial to identify individual persons of interest, to keep track of them over time, and to gather information about them. Furthermore, it is not the primary purpose of the intelligence community to produce and test general hypotheses.

III. Reasoning with instances

Most biological ontologies are written either in the OBO Flat File Format [1] or in OWL DL [2]. These ontologies are used primarily as controlled vocabularies; so far the use of biological ontologies for automatic reasoning has been surprisingly limited. However, even when biologists reason with the content of their ontologies, their needs typically differ from these in the intelligence community. Biologists are interested in type-level reasoning (so-called 'TBox reasoning'); the intelligence community is primarily interested in instance-level reasoning (so-called 'ABox reasoning'). For example, a biologist might be interested in the query 'What types of mutation lead to red eyes in fruit flies?' but a biologist would never enter the query 'Find all the fruit flies that have red eyes'. The reason is, of course, that biologists do not care about individual fruit flies; and they do not keep track of the individual animals.

In contrast, analysts in the intelligence community are interested primarily in instance-level queries about individual people and organizations and their properties and relations. For example, a typical query might be 'Find all people known to be member of Hamas, currently residents of Paris, and have been in Tehran in the last three years'. Since instance-level reasoning is irrelevant for biologists the OBO-format, which is the knowledge representation language that has been tailored to their needs, does not even allow assertions about instances. Consequently, all tools based on it do not support instance-level reasoning. Ontologies that are written in OWL DL can be used with reasoners like Pellet or Racer[1], which support instance-level reasoning. However, in spite of impressive performance improvements, as of 2008 these reasoners are not able to cope with the large-scale instance-level reasoning (ABox reasoning) that would be required by the intelligence community [3], [4], [5], [6].

IV. Time

Another difference between biological knowledge and the knowledge gathered by the intelligence community is related to time. Biological laws (and other natural laws) are timeless in the following sense: if a law like 'if a fruit fly has the mutation x, then the fly will have red eyes' is true then it is not only true now, but also at any given other time. Of course, this does not mean that biologists do not care about change over time. Evolutionary biology is strongly concerned with the changes of DNA that give rise to new species, and developmental biologists study the processes and changes that lead from fertilization to an adult organism. But while the individual organism changes over time during its development

[1]Any mention of commercial products or companies is for information only and does not imply recommendation or endorsement by the author or the National Institute of Standards and Technology.

(e.g., today's caterpillar is tomorrow's butterfly) the truth-value of statements about development in biological ontologies (e.g., 'The pupal stage follows the larval stage') does not change over time. As a result biologists have no need to express that a statement is true only with respect to a given time.

In contrast, much of the knowledge the intelligence community needs to represent is time-relative. For this reason, it turns out that the knowledge representation languages used by biologists do not meet the needs of the intelligence community. For example, it would be trivial to express a statement like 'All leaders of Hamas are located in the Gaza strip' in the OBO-format or in OWL DL but there is no straightforward way to express 'All leaders of Hamas are located in the Gaza strip on August 27, 2008.' The OBO-format cannot express statements about instances, but in OWL DL the same problem arises for statements about instances: e.g., there is no straightforward way to express 'John has been married to Sue in 2004 and John is married to Anne in 2008' in OWL DL.[2]

V. Sources

Biology, as any evolving science, contains competing theories that are inconsistent with each other. To maintain consistency, biologists limit the scope of their ontologies to textbook knowledge – knowledge that has been vetted by the community and is considered part of the scientific consensus. Obviously, this approach would not work for the intelligence community, which has to deal with conflicting information from unreliable sources. For this reason, it is crucial for the intelligence community to represent not only the information itself but also the sources of the information. A knowledge representation language suitable for the intelligence community would enable the representation of statements like 'Source x claims that Khaled Mashal will be in Tehran on August 17th or 19th'. One major advantage of representing sources and the information they provide within the same formalism is that the sources are treated as first-class citizens in the knowledge base and can be used in queries like: 'Are there two independent sources who claim that Khaled Mashai will be in Tehran?' or 'Provide source x and source y inconsistent information?'

The representation of and the reasoning about sources of information is far beyond the scope of the OBO-format as well as OWL DL. It is possible to stretch the boundaries of first-order logic in a way that one can represent information about sources. However, the resulting ontology is rather convoluted, and my experiments with Prover9 (a first-order logic reasoner [7]) showed that as a result the reasoner had difficulties to answer even fairly simple queries. A knowledge representation language that is designed to handle this kind of expression is the IKRIS Knowledge Language (IKL), an extension of the Common Logic Interchange Format [8], [9], [10]. Unfortunately, there are no reasoning engines for IKL available at this time.

[2]Note that it is possible to represent statements whose truth-values change over time in OWL DL, but the resulting ontologies are rather convoluted, and – at least in my opinion – OWL DL is a poor choice for ontologies that are intended to support reasoning with these kind of statements.

VI. Conclusion

There are some skills that biologists have developed when they adopted ontology-based technologies that might be relevant for the intelligence community: techniques to build and maintain large scale ontologies, evaluation methodologies, and general design principles for ontologies. However, biologists and the intelligence community deal with very different kinds of knowledge and create ontologies for different purposes. Thus the lessons that the intelligence community can learn from biologists will be limited: (i) The knowledge representation languages used by biologists do not meet the needs of the intelligence community. OWL DL is more expressive than the OBO-format, but since OWL DL offers no straight forward ways to deal with time-relative statements and offers no way to reason over the sources of statements OWL DL is still not expressive enough. (ii) Existing OWL DL reasoners are not able to handle the amount of instance-level reasoning that the intelligence community requires. (iii) Since the tools developed for biologists work with ontologies either in the OBO-format or in OWL DL it follows that these tools will not be useful for the work of the intelligence community.

References

[1] J. Day-Richter. The OBO Flat File Format specification, version 1.2. http://www.geneontology.org/GO.format.obo-1_2.shtml

[2] P.F. Patel-Schneider, P. Hayes, I. Horrocks. OWL Web Ontology Language semantics and abstract syntax. http://www.w3.org/TR/owl-semantics/

[3] Z. Pan Benchmarking DL reasoners using realistic ontologies. http://www.mindswap.org/2005/OWLWorkshop/sub6.pdf

[4] E. Sirin, B. Parsia, B.C. Grau, A. Kalyanpur, Y. Katz. Pellet: A practical OWL-DL reasoner. *Web Semantics: Science, Services and Agents on the World Wide Web*, vol 5, issue 2, 2007, 51-53.

[5] Racer Systems. Release Notes for RacerPro 1.9.2 beta. http://www.sts.tu-harburg.de/%7Er.f.moeller/racer/Racer-1-9-2-beta-Release-Notes/release-notes-1-9-2.html

[6] J. Bock, P. Haase, Q. Ji, R. Volz. Benchmarking OWL reasoners. In F. van Harmelen, A. Herzig, P. Hitzler, Z. Lin, R. Piskac, G. Qi: *Proceedings of the Workshop on Advancing Reasoning on the Web: Scalability and Commonsense*, 2008.

[7] http://www.cs.unm.edu/ mccune/mace4

[8] P. Hayes, C. Menzel. IKL Specification Document. http://www.ihmc.us/users/phayes/IKL/SPEC/SPEC.html

[9] P. Hayes. IKL Guide. http://www.ihmc.us/users/phayes/IKL/GUIDE/GUIDE.html

[10] ISO/IEC 24707. Information technology – Common Logic (CL): a framework for a family of logic-based languages.

Intelligence Analysis Ontology for Cognitive Assistants

Mihai Boicu, Gheorghe Tecuci and David Schum

***Abstract*—This paper presents results on developing a general intelligence analysis ontology which is part of the knowledge base of Disciple-LTA, a unique and complex cognitive assistant for evidence-based hypothesis analysis that helps an intelligence analyst cope with many of the complexities of intelligence analysis. It introduces the cognitive assistant and overviews the various roles and the main components of the ontology: an ontology of "substance-blind" classes of items of evidence, an ontology of believability analysis credentials, and an ontology of actions involved in the chains of custody of the items of evidence.**

***Index Terms*—cognitive assistant, ontology, evidence-based hypothesis analysis, types of items of evidence, chains of custody**

I. The Complexity of Intelligence Analysis

Intelligence analysts face the difficult task of analyzing masses of information of different forms and from a variety of sources. Arguments, often stunningly complex, are necessary in order to link evidence to the hypotheses being considered. These arguments have to establish the three major credentials of evidence: its *relevance*, *credibility*, and *inferential force or weight*. *Relevance* considerations answer the question: *So what?* How does this item of information bear on any hypothesis being considered? *Credibility* considerations answer the question: *Can we believe what this item of information is telling us? Inferential force or weight* considerations answer the question: *How strongly does this item of evidence favor or disfavor alternative hypotheses we are considering?* Establishing these three evidence credentials always involves mixtures of *imaginative and critical reasoning*. Indeed, as work on an analytic problem proceeds, we commonly have evidence in search of hypotheses at the same time with hypotheses in search of evidence. First, various hypotheses and lines of inquiry must be generated by analysts who imagine possible explanations for the continuous occurrence of events in our non-stationary world. Second, considerable imagination is required in decisions about what items of information should be considered in the analytic problem at hand. But critical reasoning in intelligence analysis is equally important. No item of evidence comes with its relevance, credibility, and inferential force or weight credentials already established. These credentials must be established by defensible and persuasive arguments which have to take into account that our evidence is always *incomplete*, usually *inconclusive*, frequently *ambiguous*, commonly *dissonant*, and it comes to us from sources having any gradation of *credibility* shy of perfection [1].

But the inherent complexity of the analysts' tasks are only part of their problems. In many cases, analysts are not given unlimited time to generate hypotheses and evidence and to construct elaborated and careful arguments on all elements of the analysis at hand. One way of describing this problem is to say that analysts will neither have the time, or the necessary evidential basis, for *drilling down* or decomposing all elements of the problem being considered. In many instances, analysts are faced with the necessity of having to make various assumptions in which certain events are believed "as if" they actually occurred. And always, the world is evolving and the yesterday's analysis needs to be updated with new items of evidence discovered today.

II. Disciple-LTA: Analyst's Cognitive Assistant

Disciple-LTA is a unique and complex analytic tool that can help an intelligence analyst cope with many of the complexities of intelligence analysis [2], [3]. The name *Disciple*, by itself, suggests that it learns about intelligence analysis through its interaction with experienced intelligence analysts. The word "disciple" has synonyms including: learner, advocate, supporter, and proponent. The addition "*LTA*", refers to the fact that Disciple learns analysis [L], it can serve as a tutor [T] for novice and experienced analysts, and it can assist [A] in the performance of analytic tasks, e.g. in current or in finished intelligence analyses. Disciple-LTA has two very

Manuscript received October 31, 2008. This work was supported in part by several U.S. Government organizations, including the Air Force Office of Scientific Research (FA9550-07-1-0268), the Air Force Research Laboratory (FA8750-04-1-0257), and the National Science Foundation (0750461). The US Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official polices or endorsements, either expressed or implied, of the Air Force Office of Scientific Research, the Air Force Research Laboratory, the National Science Foundation or the U.S. Government.

Dr. Mihai Boicu is Assistant Professor of Applied Information Technology and Associate Director of the Learning Agents Center in the Volgenau School of Information Technology and Engineering, George Mason University, 4400 University Dr., Fairfax VA 22030, USA (phone: 703-993-1591; fax: 703-993-9275; e-mail: mboicu@gmu.edu).

Dr. Gheorghe Tecuci is Professor of Computer Science in the Volgenau School of Information Technology and Engineering and Director of the Learning Agents Center at George Mason University, Fairfax VA 22030, USA, and Visiting Professor and former Chair of Artificial Intelligence at the US Army War College (e-mail: tecuci@gmu.edu).

Dr. David Schum holds the rank of Professor in the Systems Engineering and Operations Research Department in the Volgenau School of Information Technology and Engineering, and in the School of Law, at George Mason University, Fairfax VA 22030, USA. He is also Honorary Professor of Evidence Science, University College London, UK (e-mail: dschum@gmu.edu).

distinct differences from other knowledge-based or rule-based "expert systems" developed in the field of artificial intelligence over the years. Such systems are developed by knowledge engineers who attempt to capture and represent the heuristics or rules of the experienced expert users so that they could be preserved and utilized in new situations. This is a very long and difficult process that results in systems that are even more difficult to maintain. But Disciple-LTA is qualitatively different from these earlier expert systems.

Instead of being programmed by a knowledge engineer, Disciple-LTA learns its expertise directly from expert analysts who can teach it in a way that is similar to how they would teach a person. However, when it is first used by an expert analyst, Disciple-LTA does not engage in this interaction with a blank mental tablet. Disciple-LTA already has a stock of established knowledge about evidence, its properties, uses, and discovery. Some of this knowledge may not be already resident in the minds of its expert users, who apply their experience with certain analytic contexts that *Disciple* will learn. So, Disciple does learn about specific intelligence problems from its users, but it can combine this knowledge with what it already knows about various elements of evidential reasoning. Conventional expert systems can be no better than the expertise of the persons whose heuristics are trapped; this represents a "ceiling" on the suitability of these earlier systems. But this ceiling is actually the "floor" for Disciple-LTA, since this system incorporates basic knowledge of the evidential reasoning tasks analysts face in addition to the substantive expertise of the analysts who interact with it.

One basic feature of Disciple-LTA is that it provides the analyst the opportunity to decompose a complex problem into finer levels; i.e. it rests upon a "divide and conquer" strategy for dealing with the analytic complexity of hypothesis in search of evidence. In particular, it allows "top-down" decompositions to deduce from a stated hypothesis what needs to be proven in order to sustain this hypothesis. This decomposition eventually results in the identification of possible sources of evidence relevant to this hypothesis. Consider, for example, the problem of assessing whether Al Qaeda has nuclear weapons. This problem can be reduced to three simpler problems of assessing whether Al Qaeda has reasons, has desires, and has ability to obtain nuclear weapons. Each of these simpler problems is further reduced to even simpler ones (e.g. by considering specific reasons, such as deterrence, self-defense, or spectacular operation) that could be solved either based on the available knowledge or by analyzing relevant items of evidence. An abstraction of these decompositions is presented in the left-hand side of Fig. 1. Let us consider "Spectacular operation as reason" which is a short name for "Assess whether Al Qaeda considers the use of nuclear weapons in spectacular operations as a reason to obtain nuclear weapons." As indicated in the left-hand side of Fig. 1, to solve this hypothesis analysis problem Disciple-LTA considered both favoring evidence and disfavoring evidence. Disciple-LTA has found two items of favoring evidence, EVD-FP-Glazov01-01c and EVD-WP-Allison01-01, and it has analyzed to what extend each of them favors the hypothesis that Al Qaeda considers the use of nuclear weapons in spectacular operations as a reason to obtain nuclear weapons. EVD-FP-Glazov01-01c is shown in the bottom right of Fig. 1.

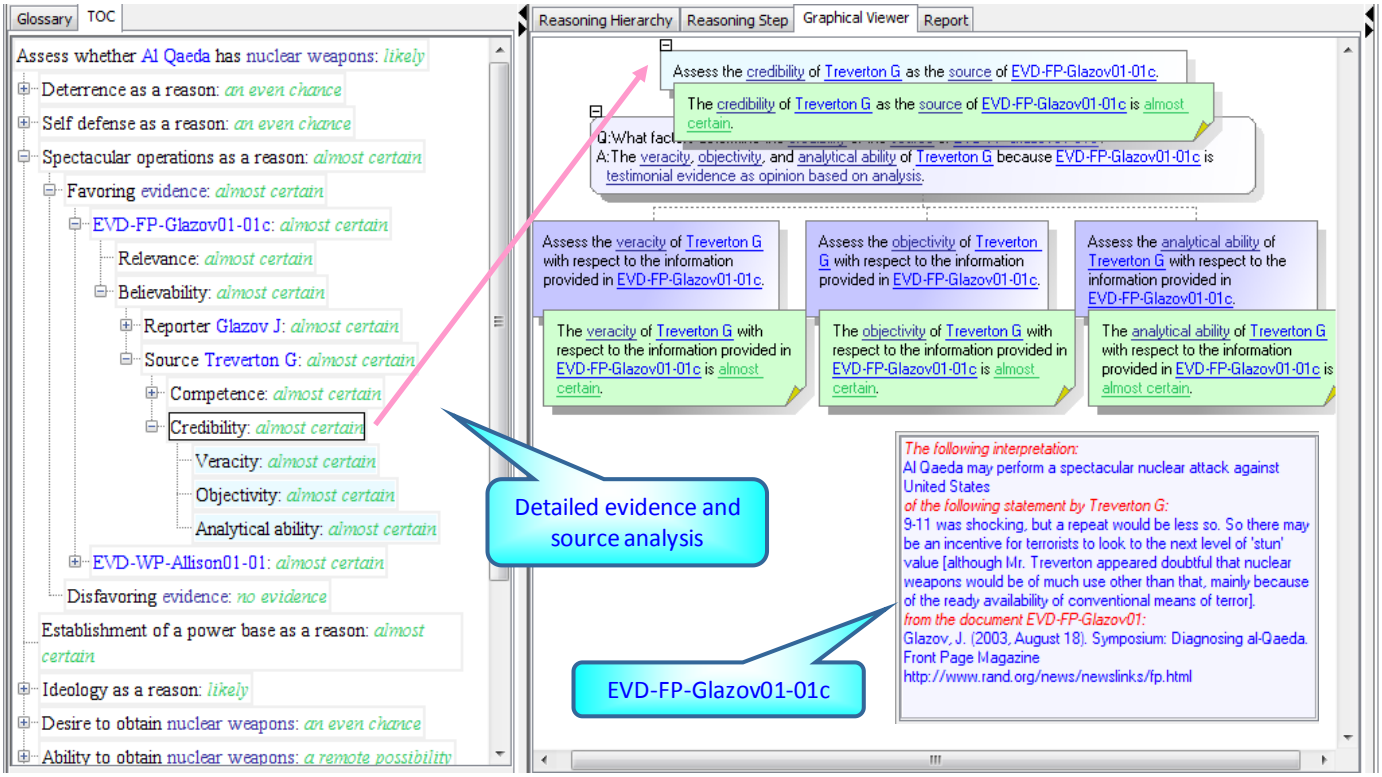


Fig. 1. Hypothesis analysis through problem reduction and solution synthesis.

It is a fragment from a magazine article published in the Front Page Magazine by Glazov J. where he cites Treverton G. who stated that Al Qaeda may perform a spectacular nuclear attack against United States [4]. To analyze EVD-FP-Glazov01-01c, Disciple-LTA considered both its relevance and its believability [1], [5]. The believability of EVD-FP-Glazov01-01c depends both on the believability of Glazov J. (the reporter of this piece of information) and the believability of Treverton G. (the source). The believability of the source depends on his competence and his credibility. The credibility of Treverton G. depends on his veracity, objectivity, and analytical ability. When the analyst clicks on a problem, such as "Credibility" from the left-hand side of Fig. 1, Disciple-LTA displays the details on how it solved that problem, as shown in the right-hand side of Fig. 1. For example, to "Assess the credibility of Treverton G as the source of EVD-FP-Glazov01-01c" Disciple-LTA has assessed his veracity, objectivity, and analytical ability. Then the results of these assessments (almost certain, almost certain and almost certain) have been combined into an assessment of the credibility (almost certain). Disciple-LTA may use different synthesis functions for the solutions (such as, minimum, maximum, average, etc.), depending on the types of the problems. A abstraction of the synthesis process is displayed in the left hand side of Fig. 1, where the solutions appear in green, attached to the corresponding problems. Notice that this problem-reduction/solution-synthesis approach enables a natural integration of logic and probability.

In some situations the analysts will not have the time to deal with all of the complexities their own experience and Disciple-LTA makes evident. In other situations, analysts will not have access to the kinds of information necessary to answer all questions regarding elements of an analysis that seem necessary. In such situations Disciple-LTA allows the user to decompose ("to drill down") an analysis to different levels of refinement in order to reach conclusions about necessary analytic ingredients, by providing mechanisms necessary to identify assumptions that are being made and by showing the extent to which conclusions rest upon these assumptions [3].

For evidence in search of hypotheses, Disciple-LTA allows the construction of "bottom-up" structures in which possible alternative hypotheses are generated. No computer system, even Disciple-LTA, is capable of the imaginative thought required to generate hypotheses and new line of inquiry. But Disciple-LTA can assist in this process by prompting the analyst to consider the inferential consequences of chains of thought that occur in the process of generating hypotheses and new lines of inquiry and evidence.

The following sections will discuss the general features of the intelligence analysis ontology of Disciple-LTA.

III. Knowledge Base Structure for Sharing and Reuse

In addition to the separation of knowledge and control (which is a characteristic of all the knowledge-based systems), Disciple-LTA is characterized by an additional architectural separation at the level of the knowledge base. Its knowledge base is structured into an object ontology that defines the concepts of the application domain, and a set of problem solving rules expressed in terms of these concepts. While an ontology is characteristic to an entire domain (such as intelligence analysis), the rules are much more specific, corresponding to a certain type of applications in that domain, and even to specific subject matter experts. This separation allows one to easily share and reuse the ontology developed for a given intelligence analysis application, when developing a new one. Additionally, the ontology in Disciple-LTA is organized as a distributed hierarchy of several ontologies, which further facilitate its sharing and reuse, as well as its development and maintenance.

IV. Multiple Roles for Ontology

The object ontology plays a crucial role in Disciple-LTA and in cognitive assistants, in general, being at the basis of knowledge representation, user-agent communication, problem solving, knowledge acquisition and learning [6]. First, the object ontology provides the basic representational constituents for all the elements of the knowledge base, including the problems, the problem reduction rules, and the solution synthesis rules. The ontology language of Disciple-LTA is an extension of OWL-light [7] that allows the representation of partially learned concepts and features. A partially learned feature may have both it domain and its range represented as plausible version space concepts [6]. One may also define different symbolic probability scales, such as Kent, DNI, IPCC or legal [8], and automatically convert from one to another and into the Bayesian probabilities. For example, the left hand side of Fig. 2 shows the symbolic probabilities for likelihood, based on the DNI's standard estimative language, while the right hand side shows the corresponding Bayesian probability intervals. The ontology also allows the representation of items of evidence that may contain different or even contradictory views on some entities.

Symbolic Intervals for likelihood

Symbolic Interval Name	Interval
no evidence	[0.0, 0.0]
a remote possibility	(0.0, 0.2)
unlikely	[0.2, 0.4)
an even chance	[0.4, 0.6]
likely	(0.6, 0.8]
almost certain	(0.8, 1.0]

Fig. 2. Symbolic probabilities for likelihood.

Second, the agent's ontology enables the agent to communicate with the user and with other agents by declaring the terms that the agent understands. As illustrated in the upper-right part of Fig. 1, the agent uses natural language phrases where the terms from the ontology appear in blue. Consequently, the ontology enables knowledge sharing and reuse among agents that share a common vocabulary which they understand. Third, the problem solving rules of the agent are applied by matching them against the current state of the agent's world which is represented in the ontology. The use of partially learned knowledge (with plausible version spaces) in reasoning, allows solving of problems with different degrees of

confidence [2]. Fourth, the object ontology represents the generalization hierarchy for learning, general rules being learned from specific problem solving examples by traversing this hierarchy [2], [3], [6].

V. Ontology of "Substance-blind" Classes of Items of Evidence

Being able to categorize evidence is vitally necessary for many reasons, one of the most important being that we must ask different questions of and about our evidence in the process of intelligence analysis in which we encounter different recurrent forms and combinations of evidence. If we were not able to categorize evidence in useful ways we might not be aware of many different questions we should be asking of our evidence. However, asked to say how many kinds of evidence there are, we could easily say that there is near infinite amount, if we considered its substance or content. This presents a significant problem: how can we ever say anything general about evidence if every item of it is different from every other item? Fortunately there is a "substance-blind" way of categorizing evidence that does not rely at all on its substance or content, but on its inferential properties: its relevance and believability.

Disciple-LTA includes an ontology of "substance-blind" classes of items of evidence. Some of the classes based on their believability attributes are shown in Fig. 3 [1].

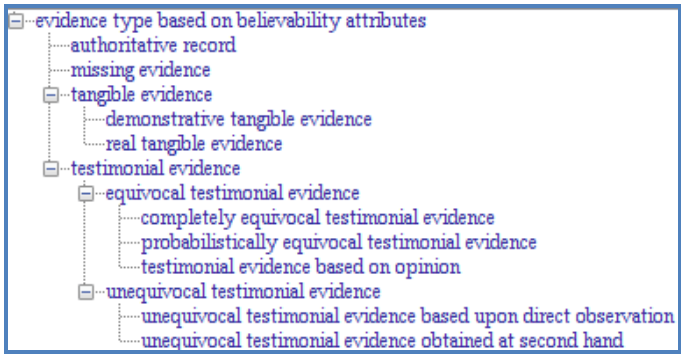


Fig. 3. "Substance-blind" classes of items of evidence.

If you can pick up the evidence yourself and examine it to see what events it might reveal, we say the evidence is *tangible* in nature such as objects, documents, images, and tables of measurements. We distinguish between *real tangible evidence* which is an actual thing itself (such as a captured weapon component), and *demonstrative tangible evidence*, which is a representation or illustration of this thing (such as a diagram of that component). Now suppose you have nothing you can examine for yourself and must rely on someone else who has made some observation and who will tell you about the occurrence or nonoccurrence of some event. This is called *testimonial evidence*, as in a HUMINT report from an asset. This person may state unequivocally that some event has occurred or has not occurred. Of great concern is how the person providing testimonial evidence obtained the information reported. Did this person make a *direct observaton* or did he/she learn about the occurrence or nonoccurrence of the reported event from another person, in which case we have *secondhand or hearsay evidence*. Moreover, there are classes of evidence mixtures, such as *testimonial evidence about tangible evidence*. It would not be uncommon in intelligence analysis to encounter evidence obtained through a chain of sources (see section VII).

VI. Ontology of Believability Analysis Credentials

As discussed above, the "substance-blind" ontology of classes of evidence is based on their *believability* and *relevance* credentials. That is, there are specific credentials for each such class. For example, the believability of a source of *direct testimonial evidence* depends on the source's *competence* and *credibility* [1], [5]. Assessments of the competence of a source require answers to two important questions. First, did this source have *access* to, or did actually observe, the events being reported? If it is believed that a source did not have access to, or did not actually observe the events being reported, we have very strong grounds for suspecting that this source fabricated this report or was instructed what to tell us. Second, we must have assurance that the source *understood* the events being observed well enough to provide us with an intelligible account of these events. So, access and understanding are the two major attributes of a human source's competence. Assessments of human source credibility require consideration of entirely different attributes: *veracity* (or *truthfulness*), *objectivity*, and *observational sensitivity under the conditions of observation*. Here is an account of why these are the major attributes of testimonial credibility. First, is this source telling us about an event he/she believes to have occurred? This source would be untruthful if he/she did not believe the reported event actually occurred. So, this question involves the source's *veracity*. The second question involves the source's *objectivity*. The question is: did this source base a belief on sensory evidence received during an observation, or did this source believe the reported event occurred either because this source expected or wished it to occur? An objective observer is one who bases a belief on the basis of sensory evidence instead of desires or expectations. Finally, if the source did base a belief on sensory evidence, how good was this evidence? This involves information about the source's relevant *sensory capabilities and the conditions under which a relevant observation was made*.

Answers to these competence and credibility questions require information about our human sources. But one thing is abundantly clear: *the competence and credibility of HUMINT sources are entirely distinct. Competence does not entail credibility, nor does credibility entail competence.* Confusing these two characteristics invites inferential disaster **Error! Reference source not found.**. Disciple-LTA includes an ontology of these credentials and Fig. 1 shows an example of using such credentials in analyzing the believability of an item of evidence.

VII. Ontology of Actions from Chains of Custody

A crucial step in answering questions on the believability of the items of evidence involves having knowledge about the chain of custody through which the testimonial or tangible item has passed en route to the analyst who is charged with assessing it. Basically, establishing a chain of custody involves identifying the persons and devices involved in the acquisition, processing, examination, interpretation, and transfer of evidence between the time the evidence is acquired and the time it is provided to intelligence analysts. Lots of things may have been done to evidence in a chain of custody that may have altered the original item of evidence, or have provided an inaccurate or incomplete account of it. In some cases original evidence may have been tampered with in various ways, the analysts risking of drawing quite erroneous conclusions from the evidence they receive. Suppose we have an analyst who is provided with an item of testimonial evidence by an informant who speaks only in a foreign language. We assume that this informant's original testimony is first *recorded* by one of our intelligence professionals; it is then *translated* into English by a paid translator. This translation is then *edited* by another intelligence professional; and then the edited version of this translation is *transmitted* to an intelligence analyst. So, there are four links in this conjectural chain of custody of this original testimonial item: recording, translation, editing, and transmission. Various things can happen at each one of these links that can prevent the analyst from having an authentic account of what our source originally provided. Fig. 4 shows how the believability of the testimonial evidence provided to the analyst (EVD-Wallflower-5) depends on the believability of the testimony of the informant (i.e. EVD-Wallflower-1), but also on the believability of the *Recording*, *Translation*, *Editing*, and *Transmission* actions. Disciple-LTA has an ontology of actions that may be involved in a wide variety of chains of custody for different types of evidence, such as HUMINT, IMINT, SIGINT or TECHINT. For example, Fig. 5 shows the representation of a translation action. The believability of this translation depends both on the translator's competence (in the two languages, as well as the subject matter being translated) and on his/her credibility.

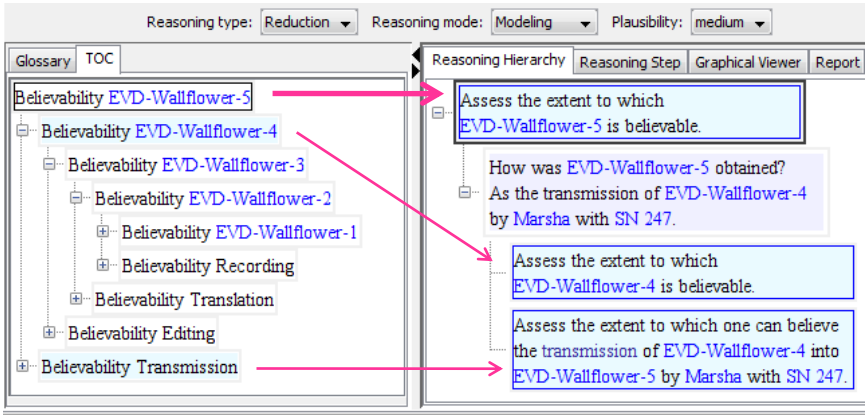


Fig. 4. Evaluating the believability of an item of evidence obtained through a chain of custody.

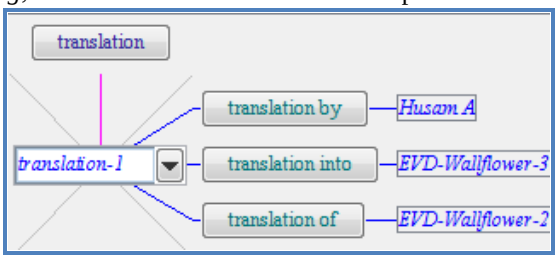


Fig. 5. Action involved in a chain of custody for an item of evidence.

VIII. Lessons and Stories about Intelligence Analysis Concepts

Disciple-LTA can be used to helps new intelligence analysts learn the reasoning processes involved in making intelligence judgments and solving intelligence analysis problems. In particular, its ontology includes lessons and stories about a wide range of intelligence analysis concepts, such as the lesson on veracity illustrated in Fig. 6 [5]. Moreover, its stock of established knowledge about evidence, its properties, uses, and discovery, makes it a suitable educational tool even for expert analysts.

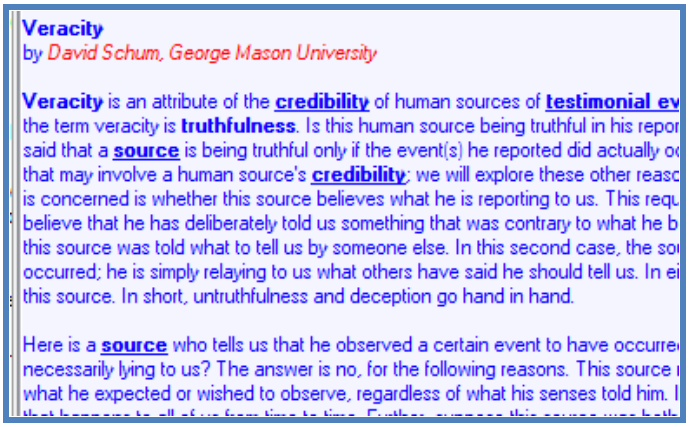


Fig. 6. Fragment from the lesson on veracity.

References

[1] Schum D.A., The Evidential Foundations of Probabilistic Reasoning, Northwestern University Press, 2001.

[2] Tecuci G., Boicu M., Marcu D., Boicu C., Barbulescu M., Ayers C., Cammons D., Cognitive Assistants for Analysts, Journal of Intelligence Community Research and Development, 2007.

[3] Tecuci G., Boicu M., Marcu D., Boicu C., Barbulescu M., Disciple-LTA: Learning, Tutoring and Analytic Assistance, Journal of Intelligence Community Research and Development, 2008.

[4] Glazov J., Symposium: Diagnosing Al Qaeda, *FrontPageMagazine.com*, 18 August 2003.

[5] Schum D.A., Lessons and Stories about Concepts Encountered in Disciple-LTA, Research Report 2, Learning Agents Center, 2007.

[6] Tecuci G., Boicu M., A Guide for Ontology Development with Disciple, Research Report 3, Learning Agents Center, 2008.

[7] W3C, W3C Recommendation, *OWL Web Ontology Language Overview*, http://www.w3.org/TR/owl-features/, February 2004.

[8] Weiss, C., Communicating Uncertainty in Intelligence and Other Professions, *International Journal of Intelligence and CounterIntelligence*, 21:1, pp 57-85, 2008.

Common Logic for an RDF Store

Robert MacGregor, Ph.D.
Franz Inc.
Oakland, CA
bob.macgregor@gmail.com

Abstract — The advent of commercial tools that support reasoning and management of RDF data stores provides a robust base for the growth of Semantic Web applications. There is as yet no analogous set of tools and products to support advanced logic-based applications. This article examines issues that arise when seeking to combine the expressive power of Common Logic with the scalability of an RDF store.

Index Terms — Common Logic, RDF, Semantic Web, higher order logic.

I. Introduction

Franz Inc is researching the possibility of implementing a Common Logic [1] parser and query processor for our RDF [2] store, AllegroGraph. There is a very wide scope for how large a subset of the language is implemented, and more significantly, what kinds of reasoning are supported. Here we discuss several of the issues and possibilities.

A primary goal for marrying Common Logic (CL) to AllegroGraph is to achieve scalable logical inference over a dynamic fact base. We believe the AllegroGraph infrastructure is well suited for crafting a scalable reasoner, however, expressive logics are inherently non-scalable, and there are severe trade-offs that must be made to achieve reasoning over billions of statements.

As part of the presentation we will discuss our proposal for implementation and requirements for the intelligence community. We will discuss a number of the tradeoff as described below.

II. Common Logic Implementation

A full-fledged implementation would include the following features:

(i) A user-friendly query language, based on CL, that supports arbitrary boolean expressions in the "where" clause (here, we are imagining augmenting a SQL-like select-from-where syntax).

(ii) Optional support for logic operators that make the closed-world assumption and unique id assumptions. We include these because classical negation and universal quantification operators are inherently non-scalable.

(iii) A CL-based rule language.

(iv) CL definitions.

(v) Extensible operators.

Just having a CL-based query language would be a big improvement on the current state of RDF-based tools. Unlike SPARQL (introduced below), it would have a "real" (model-theoretic) semantics, it would have clean syntax that assumes a calculus-like rather than an algebraic formulation of clauses, it would be expressive, and it wouldn't break your fingers when you type it.

The inclusion of a definition language that subsumes OWL (easy) would allow for a calculus that spans the range of RDF-based languages with a single syntax.

There are several "sweet spots" that could be supported; a sweet spot being a language subset that supports sound and complete reasoning while being at least moderately scalable. We will note these as we examine various trade-offs.

III. A Baseline System

The baseline is a simple query language with atomic ground assertions. Here, we explicitly exclude the possibility of rules or definitions. Such a language would allow arbitrary CL expressions to be evaluated against the fact base. However, both universal quantifiers and classical negation operators will always evaluate to false (or unknown) in this scheme; the former due to the absence of any kind of "closure" operator, and the latter due (notionally) to the absence of statements that can logically contradict one another.

The simplest way to "achieve closure" is to admit operators that define local closed world assumptions. For example, a negation-as-failure operation assumes that the facts within the scope of a single query are situationally-complete. A closed-world universal quantifier makes a similar assumption. Both of these operators are completely scalable. In other words, we can add them without reducing our expectations on how scalable our language is. The strategy of embedding the closure within a language operator, rather than, say, within a predicate, minimizes the scope of the closures, and allows both open and closed-world reasoning to be applied to the same model.

Alternatively, one could permit assertion of OWL-like operators (e.g., max-cardinality and all-values-from) to achieve closure. Introducing these operators immediately eliminates the possibility of scalable reasoning, or complete reasoning, or both. Here, we are discounting databases where large numbers of asserted and derived have been laboriously loaded and then compiled, yielding a base that melts on the first update; a "dynamic" scalable application will support real-time updates as a matter of course.

Our baseline would not be complete without a transitivity operator. Transitive closure is the most important of all the classes of inference. Simplest would be to include the equivalent of an 'owl:TransitiveProperty' declaration, but practical experience has shown that the addition of a transitive closure operator to the query language syntax has important benefits. Specifically, it is useful to be able to define transitive closures over compound binary expressions; something that OWL can't do. AllegroGraph includes specialized accelerators for computing transitive closures.

IV. Adding Rules

The addition of Horn-like rules significantly increases the utility of the language. Here we face a choice. If our rules are recursive, then syntactic constraints must be placed upon both the heads and bodies of our rules if we are to retain completeness. This results in a Prolog-like semantics, with a CL syntax. The accompanying reasoning can be made moderately scalable.

Alternatively, we can degree that our rules are non-recursive. Here, we restrict our rules to having atomic heads, but we can permit arbitrarily expressive tails. This scheme is both scalable and complete. We know this because these rules do not actually increase the expressive power of the query language. Instead they are a convenience (a major convenience). There are relatively few examples of systems built using non-recursive rules; however, limited practical experience has revealed that most recursive Horn rules can be reformulated into equivalent non-recursive rules combined with an (expressive) transitive closure operator. The most serious drawback to this scheme (non-recursive rules) is that it is theoretically uninteresting. There is nothing semantically to write about, so there are no papers on the subject.

Finally, one could design a system that combines recursive and non-recursive rules. This is a quite viable option. The only caveat is that only highly-disciplined users are likely to reformulate as many rules as possible into non-recursive equivalents. The benefits of doing so would be orders of magnitude increases in query performance, but your average user might not master the technique.

V. Definitions

We face another choice when we add definitions into the mix. If we interpret our definitions as if-and-only-if rules, then we have abandoned hope of scalable inference. Alternatively, we can apply an asymmetric (if but not only-if) interpretation to Horn-like definitions to achieve an expressivity equivalent to Horn rules These are superior to one-directional rules, because the only-if portion can be reserved for constraint-checking/data validation. A single syntax should suffice for either interpretation of a definition (asymmetric or bi-directional); one can envision using a single set of definitions for both scalable inference and small-scale but rich inference.

VI. Inference

Tableaux-based reasoners appear to be inherently non-scalable over dynamic databases. Instead, we focus chiefly on rule-based reasoners. There are three basic classes of rules: (1) backward-chaining rules, (2) forward-chaining rules, and (3) rewrite rules. Backward-chaining rules are the best-behaved. They are relatively insensitive to database updates (cache-busting will occur, but it is manageable) and they are moderately scalable.

Forward-chaining rules are more powerful (from a completeness standpoint) than backward rules. However, some form of truth maintenance is required to manage derived facts, and bitter experience has shown that truth maintenance does not scale. Hence, this option is not viable for large scale applications.

Rewrite rules (also called "triggers") are essentially forward rules that don't bother to clean up after themselves when updates are made. Instead, they are interpreted as having a sematics external to the system. This makes them highly useful, but it is "buyer beware" when it comes to semantics. Rewrite systems have difficulty managing the trigger portion of very expressive rules. For the handling of expressive rewrite rules, we recommend the introduction of "trigger" clauses into the syntax. The assumption is that such rules will fire only when updates to the fact base are detectible by the trigger portion(s) of the rule; other (presumably more expensive) clauses in the rule will not be monitored. Most uses of rewrite rules (e.g., Jess rules) are applied only to modest sized databases.

The extensible operator feature allows arbitrarily complex operators to be added to the language. This allows for exotic operators like "cut" or modals to be added. This is possible because the specialist mechanism includes hooks into the internals of the query executor. High-end inference can be achieved by adding additional operators to the rule engine that include their own logic interpreters.

AllegroGraph's implementation of CL will use the extension mechanism to provide access via CL to its built-in geospatial, temporal and social network analysis features.

VII. Common Logic And RDF

Scalable logic-based applications will most likely be built on top of an RDF triple store. It makes sense to ask what contribution Common Logic can make in this context. In fact, a query language based on Common Logic would have a number of advantages. This is due in part to the fact that SPARQL, the defacto standard in the RDF world, has a number of serious deficiencies that discourage its use for higher-level logic applications.

SPARQL [3] is a W3C-recommended query language for RDF data. It has been designed to enable expressions of common, everyday queries in a style that mimics a syntax used elsewhere to express atomic ground assertions. The majority of developers of RDF stores provide implementations of SPARQL; this has significantly spurred the growth of RDF-based tools and technology.

One serious drawback of SPARQL is that it takes the "kitchen sink" approach to syntax. SPARQL has two "and" operators, two "or operators, and an awkward division between predicates evaluated against the store versus predicates evaluated by other means (e.g., equality, inequality, etc.). Rather than treating the context/graph dimension as simply one additional argument (to a triple), it adds orthogonal syntactic constructs that interleave with the already cumbersome triples and filters. While simple SPARQL queries are fairly readable, when complexities such as disjunctions are utilized, SPARQL queries become very difficult to compose and interpret.

In the logic world, a primary weapon to counter syntactic complexity is to base the semantics of a logic on a small number of primitive operators, and to define the remaining operators as compositions of the primitives. In this case, the bulk of language syntax may be regarded as syntactic sugar; this makes the job of implementing the language much more manageable. This is how, for example, KIF [4] and Common Logic have been defined. SPARQL has taken the opposite approach; it has a large number of different semantic operators, and is defined in terms of a procedural semantics rather than a declarative semantics. That means that the traditional compositional semantics approach cannot be applied to SPARQL.

The combination of bloated syntax and an essentially non-existent semantics means that SPARQL cannot readily server as a foundation for the addition of rules, modal operators, and other higher level constructs. This leaves the field open to competing languages such as Common Logic.

VIII. Expressive Power Examples

In this section, we look at some simple examples where the expressive power of Common Logic can be applied to treat representational problems that are difficult or impossible to solve using a SPARQL-like language. We will use a KIF-like syntax to express our rules.

A common claim made by many RDF advocates is that "the Semantic Web is open world". Practical experience indicates that this statement is a complete falsehood; in fact, not only are there "pockets" of assertions in most semantic networks best treated using close-world semantics, but these "pockets" tend to be the locus of the highest-valued information. Therefore, a practical Semantic Web language will include constructs to treat close-world models.

Consider the predicate "single", as in "not married". It is conventional to treat the definition of the "single" predicate as the closed-world negation of the predicate "married", e.g.,

```
(<= (married ?p)
     (exists (?s) (spouse ?p ?s)))

(<= (single ?p)
     (not (married ?p)))
```

In other words, if you don't know that a person is married, assume s/he is single. This isn't guaranteed to be true; but its the way that personnel data is utilized a great deal of the time.

The semantics of closed-world negation can either be assumed to attach to the underlying domain model, or to be attributed to a logic operator. In the latter case, since 'not' denotes classical negation, we would replace 'not' by a specific negation-as-failure operator (variously called 'thnot', 'unsaid', etc.) to achieve the desired semantics, e.g.,

```
(<= (single ?p)
     (unsaid (married ?p)))
```

Next, consider universal quantification. The rule below states that you are "off the hook" if all of your children have graduated from college:

```
(<= (off-the-hook ?p)
     (forall (?c)
          (implies (child ?p ?c)
              (graduated-from-college ?c)
```

The trick to evaluating this predicate in a practical domain lies chiefly in determining if the set of children known for an individual Fred constitutes the complete set of Fred's children. This kind of information is typically hard to locate. Instead of looking for a guaranteed answer, it is more typical to query for all of Fred's children, and ask if each of those retrieved has graduated. This answer can be trusted as far, and only as far, as the closed-world assumption holds.

The ability to make closed-world assumptions about sets of entities is critical to many real-world applications. Having a universal quantifier in the language enables this reasoning to be computed endogenously, rather than relegating it to the procedurally-evaluated portion of an application.

One would also like aggregate entities to be treatable within a logic. Here is a (somewhat simplistic) definition of the term "family":

```
(<= (family ?p ?fam)
   ?fam = (setof (?r) (or (spouse ?p ?r)
                          (child ?p ?r))))
```

Query languages such as SQL and SPARQL do not allow for explicit universal quantifiers in their syntax. This has two consequences: (i) it limits the kinds of universal quantification expressible in these languages (SQL has various aggregate operators; SPARQL makes no provisions for universal quantification); (ii) it requires that scope rules for variables be implicit rather than explicit, which works well most of the time, but not always. Here is an example representing a simplification of an application that this author encountered, where the lack of an explicit existential quantifier (and accompanying scoping) made composing the query difficult. The (simplified) problem is to query for two degrees of distance from Kevin Bacon, based on a 'knows' relationship. Here is the query expressed *without* reference to existential quantification:

```
(select ?x (where
     (or (?x = Kevin)
        (and (knows Kevin ?x1)
           (or (?x = ?x1)
              (and (knows ?x1 ?x2)
                 (?x = ?x2))))))
```

The query succeeds only if the variable ?x1 is the same throughout the query. In many quantifier-free languages (e.g., SPARQL) variables in parallel disjuncts can have the same name but not be considered the same variable. This is done for a very good reason; however, it means that we can't be sure how the above query will be evaluated without a detailed inspection. The actual query found in the application was more complex than this, because the entities were related by more than one predicate. If you replace

```
(knows ?x1 ?x2)
```

above by

```
(or (knows ?x1 ?x2) (likes ?x1 ?x2)
```

then you will have a better approximation of the complexity of the query in the application. Doing so makes the scoping that much more tenuous. In fact, the query language used in the application turned out to have scoping rules that assumed that the variable ?x1 was *not* unique across the query. This made it necessary to rewrite the query, approximately doubling its size. On the other hand, if we have an explicit existential quantifier, none of this "guessing" is necessary:

```
(select ?x (where
    (or (?x = Kevin)
        (exists (?x1)
           (and (knows Kevin ?x1)
             (or (?x = ?x1)
                 (and (knows ?x1 ?x2)
                     (?x = ?x2)))))))
```

Lastly, a host of logic-based applications find it useful (and in a cognitive-sense, "necessary") that the language support n-ary predicates and n-ary functions. The Franz product features a suite of geospatial,

temporal, and semantic network reasoners that are best exploited using queries that employ n-ary predicates. The query below evaluates:

Retrieve important people known to Bob who attended a meeting in or near Berkeley, CA in November, 2008.

```
(select (?p)
     (and
     (ego-group bob knows ?group 2)
     (actor-centrality-members
            ?group knows ?p ?importance)
     (participant ?event ?p)
     (instance ?event Meeting)
     (interval-during ?event "2008-11-01"
                             "2008-11-05")
     (contains (geo-box-around
            (location Berkeley) 5 miles)
            (location ?event))))
```

Here the 'ego-group' predicate is a distance-2 Kevin Bacon computation (note how much simpler it is than the previous query).

Another comment on the "Kevin Bacon" query: When the relationship predicate is the same on all layers, then a built-in version of the computation can be expected to execute significantly faster than the same computation phrased in logic. However, the original query referenced a different predicate at the first level than the second, and referenced four different predicates at that second level, so a built-in operator was not available. The moral being that built-ins are not a universal panacea for expressiveness.

This section has surveyed a sample of Common Logic language constructs to suggest that users benefit both by (i) the ability to program a larger portion of their applications within the logic, rather than resorting to procedural manifestations, and (ii) that use of more expressive constructs can reduce the complexity of the resulting rules and queries, making the language more usable by humans.

IX. SUMMARY

Adding a Common Logic interface and interpreter to an RDF store would provide a spectrum of possible benefits. At one end, a careful exploitation of CL features would provide "heightened" versions of semi-conventional query processing, over a dynamic, scalable platform. At the high-end, one can contemplate experimenting with combinations of powerful reasoners operating over relatively small sets of data interacting with the large-scale query engine.

The implementation community for Common Logic needs to produce a target specification that is both "doable" and useful to a significant class of applications. There is a chicken-and-egg component, since one needs to have an expressive language available to appreciate why and how one can use it.

REFERENCES

[1] International Standard ISO/IEC 24707 Information technology — Common Logic (CL): a framework for a family of logic based languages. URL: http://standards.iso.org/ittf/PubliclyAvailableStandards/c039175_ISO_IEC_24707_2007(E).zip.

[2] RDF/XML Syntax Specification (Revised) W3C Recommendation 10 February 2004. URL: http://www.w3.org/TR/rdf-syntax-grammar/.

[3] SPARQL Query Language for RDF W3C Recommendation 15 January 2008. URL: http://www.w3.org/TR/rdf-sparql-query/.

[4] Knowledge Interchange Format draft proposed American National Standard (dpANS) NCITS.T2/98-004. http://logic.stanford.edu/kif/dpans.html.

Unification of Geospatial Reasoning, Temporal Logic, & Social Network Analysis in an RDF Database

Jans Aasman
Franz, Inc.
555 12th Street, Suite 1450
Oakland, CA 94607
+1-510-452-2000
ja@franz.com

Abstract - This paper is about a new type of event database that enables efficient reasoning about things, people, companies, relationships between people and companies, and about places and events. The event database is built on top of a scalable distributed RDF triple store that can handle literally billions of events. Like objects, events have at least one actor, but usually more, a start-time and possibly an end-time, a place where the event happened, and the type of the event. An event can have many additional properties and annotations. On top of this event database we implemented libraries for RDFS++ logic reasoning, for geospatial and temporal capabilities, and an extensive social network analysis package. This paper focuses on a query framework that makes it easy to combine all of the aforementioned capabilities in a user friendly query language.

Index Terms – Geotemporal logic, geospatial reasoning, RDF database, graph database, RDFS, OWL, SPARQL, social network analytics, business intelligence, event-based systems, event-driven architectures, metadata, semantic technologies.

I. INTRODUCTION

This paper describes the design and use of a unifying query framework for geospatial reasoning, temporal logic, social network analytics, RDFS and OWL in Event-based systems [1]. In this introduction we will first go into why we need such a framework and the requirements for such a framework.

The reason for such a framework can be answered by looking at the vision of the semantic web and understanding how companies use semantic technologies. Tim Berners-Lee, James Hendler and Ora Lassila's Scientific American article (May, 2000) [2] provides a compelling vision of the Semantic Web. It contains some interesting use cases for what the Semantic Web will bring. These use cases assume that software agents know how to roam the web and reason over things, people, companies, relationships between people and companies and about places and events. Clearly these agents need a query capability that supports a combination of description logic, geospatial reasoning, temporal reasoning, and knowledge about the social relationships between people.

The commercial vendors of Semantic Technologies also see a number of use cases that all center around events and require the aforementioned query capabilities. We currently see companies using large data warehouses with very disparate RDF based triple stores describing various types of events where each event has at least two actors, usually a begin and end time, and very often a geospatial component. These events are literally everywhere: in Health Care applications we see hospital visits, drugstore visits, and medical procedures. In the Communications Industry we see telephone call detail records, including location. An email and calendar database of a large company is nothing more than a social network database filled with events in time and, in many cases, space. In the Financial Industry every transaction is essentially an event. In the Insurance Industry claims are important events and they desperately need more activity recognition. In the Intelligence community basically everything revolves around events and actors. The REWERSE program from the 6th Framework Programme of the EU Commission [3] is one of the few systematic efforts to combine RDFS/OWL with geotemporal reasoning, although the social aspect hasn't been addressed yet. The recent book "The Geospatial Web" [4] currently provides the state of the art overview on how to work with people and events on a web scale and what kind of applications we might expect in the near future.

II. FRAMEWORK REQUIREMENTS

The Semantic Web community has made great strides in the area of ontologies and description logic, and some initial work in the areas of geospatial reasoning [5], temporal reasoning [6], social network analysis [7], and event ontologies [8]. All of this is based on RDF as the data representation. Based on this W3C standard the combination of all these different reasoning capabilities in one unified framework will propel further industry adoption of Semantic Technology. Given that we have seen a direct need for query capabilities that handle geospatial/temporal/social/rdfs/owl, we have designed a framework. The main requirements we identified were:

1. User and programmer friendly: We wanted the framework to be an extension of SPARQL, with SPARQL as the foundation. Certainly the framework should not be anymore complex than SPARQL. SPARQL is relatively user friendly, and as languages go, the adoption rate is such that one could make the argument that it is sufficient to address most use cases.

2. Implementer friendly: We need many people to experiment with this proposed framework such that the Semantic Web community can converge on a standard.

3. Efficient: Given that we work with very large databases with millions of events where the response time has to be on the sub second level, the implementation of the query language and query engine needs to be very fast

4. We want the query language to work on distributed databases. Currently we've designed the query engine to work on federations of triple stores. Once we develop efficient caching techniques for distributed RDF knowledge stores residing all over the web, it will also be efficient for agents that need to roam the web.

5. Practical & Easily Extendible: We want the API to be such that it can be easily modified to allow for ongoing experimentation.

6. Works well with RDFS and OWL reasoning.

III. DISCUSSION

In the remainder of the paper we show how we can combine geospatial reasoning, temporal logic, social network analytics, and RDFS reasoning all in one query language.

One question that people ask who are familiar with triple stores is: how can this work efficiently on very large data sets containing billions of triples? Most first generation triple stores store the URIs and literals that constitute the parts of a triple as strings in a dictionary. So, when doing range queries over numeric values, for example, "select * from person where age > 50", the triple store engine has to go through each value for the predicate 'age'. One way around this is to add btrees for every numeric type but that in general is a very inefficient solution in triple stores. The triple store that we use is AllegroGraph which is actually a hybrid between a relational database and triple store, the internal representation of the triples is such that is allows for very efficient range queries.

A. Temporal Reasoning

Our temporal reasoning is based on James Allen's Interval Logic [9]. This logic looks at all the 13 ways two temporal intervals can relate to each other. We provide predicates for each of Allen's 13 interval predicates. Note that we do purely quantitative temporal reasoning. So if you provide a number of events with a start time and an end time or a duration then we can perform queries like the following. This example will return all intervals ?i2 that happened in interval ?i1.

(select ?x (interval-during ?i1 ?i2))

Temporal reasoning uses the range query capabilities to the fullest extent. If you want to find all the events that happened between Jan. 1, 2008 and Jan. 2, 2008, the triple store performs a straight triple query with only one cursor scan. It is still possible to blow up the query time spectacularly by doing things like

(select (?x ?y) (point-before ?x ?y))

as that will generate every before/after pair. However, we do consider that to be the responsibility of the user. In many cases a query optimizer can warn for that or rearrange the clauses to bind ?x or ?y.

B. Geospatial Primitives

Our original intention of adding Geospatial capabilities was not so much to compete with existing spatial databases but instead make it very easy for RDF users to be able to deal with locations of objects very efficiently. In order to make this fast we implemented a variation of an R-Tree to encode two-dimensional data very efficiently directly in the triple indices [10]. A detailed description of how this geospatial representation works can be found in the geospatial tutorial included with the AllegroGraph documentation [11]. Currently we support a number of predicates that can be used in the query language. Some examples of the predicates:

(geo-distance ?x ?y ?dist) -> given, x and y, return distance

(geo-within-radius ?x ?y 10.0) -> find y within 10 miles from x

(geo-inside-polygon ?polygon ?place ?lon ?lat)

For our benchmarking we use the open source GeoNames database that can be freely downloaded from GeoNames.org [12]. The database contains nearly 7 million points of interest on earth. From interesting points in nature, to populated areas, to schools and churches, etc. Each point has 12 features such as asciiname, the local name, elevation level, longitude, latitude, population, etc. Actually, it is not a database but a csv file that programmers can modify as necessary. For our purposes we obviously transform it into RDF triples. We can retrieve all 459 geo-points around Berkeley less than 4 miles away in less than 5 milliseconds. We would argue that the basic retrieval speed is comparable to or better than current commercial spatial databases. Here are some typical example queries that you can do on the GeoNames database:

Find the distance between Oakland and the one and only Berkeley in California.

(select (?dist)

```
(q ?x geo:name “Oakland”)
(q ?y geo:name "Berkeley")
(q ?y geo:admin1_code "CA")
(geo-distance ?x ?y ?dist))
```

Put in a Google map all the places within 10 miles from Oakland

```
(google-map  (select (?name ?lat ?lon)
            (q ?x geo:asciiname “Oakland”)
            (geo-within-radius ?x ?y 10)
            (q ?y geo:asciiname ?name)
            (q ?y geo:isAt5 ?pos)
            (pos->lon/lat ?pos ?lon ?lat)))
```

C. Social Network Analysis (SNA)

Many RDF resources are about people and relationships between people, or between people and companies, or between companies and other companies. We added Social Network Analysis methods to make it easier to reason about relationships and groups. The functions that we provide address the five basic questions from Social Network Analysis. (1) How far is person A from person B, (2) if there is a link between A and B then how strong is this relationship, (3) given a particular actor A, in what group does this actor ‘live’, (4) given an actor in a group, how important is this actor in the group and finally, (5) given a group, how dense are the relationships in the group and does this group have a leader or a set of leaders. The SNA library encompasses a set of well know SNA algorithms. We provide a set of general functions and have developed the concept of a generator. A generator is basically a function that takes as an input one node and then creates a set of output nodes. The search functions and SNA functions that we provide take these generators as first class arguments. For example: say we have a database with relationships between people, the generator ‘knows’ will take as an input a person and return a set of person(s) by following fr:went-to-dinner-with and fr:went-to-movies in both directions.

```
(defgenerator knows ()
    (bidirectional fr:went-to-dinner fr:went-to-movies))
```

We can use this generator to find, for example, the shortest path between two people. In this case the query will return a list of persons.

```
(select ?x
        (shortest-path knows fr:Person1 fr:Person2 ?x))
```

Or we can use the generator to first create a group of friends and friends of friends in the ego-group predicate, and then we find the importance of each member using the actor-centrality measure. This predicate will start with the most important one first.

```
(select ?x
        (ego-group fr:Person1 knows 2 ?group)
        (actor-centrality-members ?group knows ?x))
```

AllegroGraph is a native, general graph database, written specifically to make graph search faster. However, the bottleneck is still getting triples from disk as fast as possible and having the smartest algorithms and best caching available. For example, many of the centrality measures that are used to compute the importance of an actor in a known group need to compute the shortest path between every actor in the group. We have created special constructors to cache these groups in a transparent way so that most computations can be done without minimal IO.

IV. AN OVERVIEW EXAMPLE

In order to give the reader an impression of the breadth and depth of the query language, we provide a typical example that combines geospatial, temporal, SNA and RDFS reasoning.

```
(select (?x)
  (ego-group person:jans knows ?group 2)
  (actor-centrality-members ?group knows ?x ?num)
  (q ?event fr:actor ?x)
  (qs ?event!rdf:type fr:Meeting)
 (interval-during ?event “2008-12-01” “2008-12-05”)
  (geo-box-around geoname:Berkeley ?event 5 miles)
  !)
```

In English this translates into:

> *Find the group of friends and friends of friends around the person “Jans”. Find within this group the most important person first. Find if this person was part of an event that was of type Meeting and happened in a particular time interval within 5 miles of Berkeley.*

Note that we seamlessly mix Social Network Analysis in the first two clauses, a simple database look up in the third, an RDFS inference about the type of event, and then a temporal and a geospatial constraint. This current example and the examples shown above utilize Prolog. We expect in early 2009 to have a SPARQL engine that will perform this identical query.

The syntax of the SPARQL query will be slightly more contrived due to the fact that SPARQL normally only allows patterns that map directly on triples (see example below). Note that we introduced the non-standard ‘=’ or assignment construct. We are planning to discuss this topic with the SPARQL committees.

```
select ?x where {
  ?group =  ego-group(person:jans knows 2) .
  ?x = actor-centrality-members(?group knows ?x) .
```

```
?event fr:actor ?x ;
        rdf:type fr:Meeting .
FILTER (interval-during ?event '2007-12-01' '2007-12-31')
FILTER (geo-box-around geoname:Berkeley ?event 5miles)
}
```

V. SUMMARY AND FUTURE RESEARCH

In this paper we have discussed how RDF can serve as a basis for an event database where events are defined as 'things' that (1) require RDFS++ reasoning because events have types, (2) require geospatial reasoning because events happen somewhere, (3) require temporal reasoning because events nearly always have a start and duration and (4) require some form of social analysis because most interesting events have one or more actors.

We demonstrated how all of these capabilities can be used in one query language, in this case Prolog. And we expect that in the near future these capabilities will be available in SPARQL as well.

The primary research effort for the current version of the query framework is to enhance query-optimization. Notice that in the example shown above, most clauses are not direct matches against the database but functors that do computations. Some of these functors can act both as generators and as filters (as is common in Prolog). In case a functor acts as generator we need to research better statistical predictions for how many solutions can be expected so that we can do better re-ordering of clauses.

VI. REFERENCES

[1] Aasman, J., Unification of geospatial reasoning, temporal logic, & social network analysis in event-based systems, Distributed Event Based Systems (DEBS 2008) http://portal.acm.org/citation.cfm?id=1386007

[2] 1st Scientific American article on the Semantic Web, http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21&ref=sciam

[3] REWERSE, http://rewerse.net

[4] Scharl, A., Tochtermann, K.: The Geospatial Web: How Geobrowsers, Social Software and the Web 2.0 are Shaping the Network Society. Springer (2007)

[5] W3C Geospatial Incubator Group, http://www.w3.org/2005/Incubator/geo/

[6] Gutierrez, C., Hurtado, C., and Vaisman, A. Temporal RDF. In European Conference on the Semantic Web (ECSW'05) (Best paper award), pages 93–107, 2005, http://www.dcc.uchile.cl/~cgutierr/papers/temporalRDF.pdf

[7] Mika, P.: Social Networks and the Semantic Web. Springer (2007)

[8] Raimond, Y. Abdallay, S., Event Ontology, 2007, http://motools.sourceforge.net/event/event.html

[9] Allen, J.F.: Time and Time Again: The Many Ways to Represent Time. International Journal of Intelligent Systems, Vol. 6, No. 4 (1991)

[10] Wikipedia R-tree data structure, http://en.wikipedia.org/wiki/Rtree

[11] Geospatial tutorial section of Franz Inc.'s AllegroGraph 3.0 documentation, http://agraph.franz.com/support/documentation/current/geospatial-tutorial.html

[12] GeoNames Data Access, http://www.geonames.org/export/

Toward an Open-Source Foundation Ontology Representing the Longman's Defining Vocabulary: The COSMO Ontology OWL Version

Patrick Cassidy
MICRA, Inc., Plainfield, NJ
cassidy@micra.com

Abstract - **The COSMO foundation ontology is being developed to test the hypothesis that there are a relatively small number (under 10,000) of *primitive* ontology elements that are sufficient to serve as the building blocks for any number of more specialized ontology elements representing concepts and terms used in any computer application. Finding evidence for this hypothesis would suggest that a promising tactic to achieve *Semantic Interoperability* among computer applications is to focus effort on the common foundation ontology to that ontology that contains those primitive elements. This will constrain the size of the ontology on which agreement is required, to the minimum that will support accurately relating domain and application ontologies to each other. The rationale, methodology and current status of this project is reported here.**

Index Terms – **Foundation ontology, conceptual primitives, COSMO, semantic interoperability, common ontology, ontology mapping, Longman, defining vocabulary.**

I. Introduction

Information communicated and analyzed by the intelligence community is highly diverse, including technical, social and psychological concepts. The challenge of using automatic techniques for integrating such information will require adoption of an ontology that is capable of unambiguously representing the full range of knowledge that people communicate. There is as yet no consensus on how to structure that ontology. This paper describes one approach to overcome the lack of agreement caused by multiple fundamentally different approaches to foundation ontology development. The proposed approach depends on three factors: (1) to develop a foundation ontology that is effective as a standard of meaning for communication among many applications, it is not necessary to achieve universal agreement among ontology developers about the structure of the foundation ontology; it is only necessary to build a sufficiently large user group that third-party vendors will have incentive to develop utilities making the ontology easier to use, and applications that demonstrate the usefulness of the ontology for practical purposes. (2) by allowing multiple logically compatible views for representing the same entities, and providing translation utilities between them, many of the differing preferences for representing entities can be accommodated in the same ontology. (3) the number of different ontology groups that will accept the ontology can be maximized by keeping the foundation ontology as small as possible without compromising its ability to support logical representation of terms and concepts in any application domain. In the COSMO approach , that could be achieved by discovering the smallest inventory of fundamental ontology elements, representing the minimal essential primitive concepts that are needed to build representations of any more complex concept.

II. Background to the **COSMO** approach

A. The Notion of Conceptual Primitives

The approach proposed here relies on the observation that communication among agents (human or automated) depends on the agents sharing some common set of internally understood concepts, labeled by an agreed set of symbols such as words in human languages, or element names in databases. Wherever a particular community uses concepts not already among the known concepts of other communities, information sharing requires the first community to use a common set of *defining concepts* to construct definitions of the unknown concepts understandable to the other communities. In this manner communicating agents can accurately transfer information on topics familiar or initially unfamiliar to other agents. Information transfer using human languages is facilitated by the existence of a relatively small vocabulary of basic words, representing those commonly understood concepts, that can be used to create linguistic definitions of any specialized concept. Research in Linguistics has explored by experimental techniques the number and identity of the common primitive concepts that are used in linguistic communication among people speaking different languages. Some of that work, summarized by Goddard[1], has suggested that as few as 60 semantic primitives are adequate to construct definitions of a very large number of concepts. A less systematic but more comprehensive demonstration of the power of primitive concepts to suffice for construction of definitions of many words is found in some English-language dictionaries such as

the Longman [2] that use a Defining Vocabulary of basic words with which to define all of the entries in the dictionary. The Longman Defining Vocabulary (hereafter LDV) contains 2148 words, but an investigation [3], [4], [5] has shown that even fewer words are needed to define (recursively) all of the Longman entries. For cases where a proposed definition of a new word uses words not already in the defining vocabulary, the Defining Vocabulary tactic requires that the unrecognized word itself be defined by use of the basic Defining Vocabulary. The answer appears to be that, for the Longman, words recursively defined in such a manner "ground out" using a basic vocabulary of 1433 words representing 3200 word senses.

The success of the linguistic defining vocabulary for dictionaries suggests that a similar tactic could be effective for automated information transfer among computer systems. For automated systems, the "Defining Vocabulary" would take the form of a ***foundation ontology*** having an inventory of basic concept representations that is sufficient to create representations of any new concept, by combinations of the basic elements. Communities using such a "***Conceptual Defining Vocabulary (CDV)***" (i.e. a common foundation ontology) would be able to pursue their own interests using any local terminology or ontology that suits their purposes, and still communicate their information accurately in a form suitable for automated inferencing, by translating the local information into the terminology of the *common foundation ontology*. Limiting the core foundation ontology to the elements needed for a CDV will minimize the effort required to perform the translations, while ensuring that accurate translations are possible. The question remains whether the linguistic Defining Vocabulary examples can be adapted to the more precise requirements of representing terms and concepts in a logical format, suitable for automated reasoning.

The essential principle of such a tactic for Semantic Interoperability is that, when the separately developed ontologies of two different systems both use the same CDV to specify the structures of their ontology elements, then accurate information sharing can be achieved, even if the two systems each have some separately-defined ontology elements not in the other, by *sharing* the specifications of the ontology elements of each that are not in the other. Since the ontology elements of each system are built from the same primitive elements of the CDV, they will be properly and accurately interpretable in both systems. The combination of the ontologies of the two systems in effect creates a single merged ontology common to both systems. In that situation, the same input data in both systems will produce the same inferences. Different data in the two systems will create some different inferences, but those will not be logically inconsistent if the data is not inconsistent. For a proper automated merger of the two ontologies, it will be necessary to have utilities that can automatically recognize identical elements created in the two separate local ontologies, and to detect inconsistencies if they exist. But this tactic for interoperability avoids the impossible task of automatically interpreting information in an external ontology that is based on fundamentally different (usually undocumented) assumptions about how to represent the same intended meanings of terms and concepts.

B. The Current Absence of a Conceptual Standard

To function as a conceptual standard that will enable **semantic interoperability**, i.e. permit computers to reason accurately and automatically with transferred information, the syntactic format for a common standard must have at least the expressivity of First-Order Logic (FOL), so as to permit logical inference using rules expressing domain knowledge. Several foundation ontologies, such as OpenCyc[6], SUMO[7], DOLCE[8], and BFO[9], have been developed that have this technical capability. Other knowledge classifications such as NIEM[10] and the DoD Core Taxonomy[11] have less expressiveness. None of these projects has adopted the tactic of creating a CDV, and none has been recognized as a default standard for application builders concerned with specific topics and indifferent to the nuances of representation at the abstract levels. The reasons for lack of wide adoption vary. The complexity of each of the existing foundation ontologies presents a steep learning curve which requires a strong motivation to impel potential users to spend the required time. In the case of Cyc, much of the content (such as the over 1000 specialized reasoning modules) is still proprietary and cannot be part of an open-source project that could include desired components from many non-Cycorp sources. Development of an effective open-source natural-language interface to the ontology is also desirable, to make learning and use convenient. None of the existing foundation ontologies has such an interface. Without publicly available examples showing the benefits of using a complex ontology, a specialized application developer without a need to interoperate

outside the local community is strongly tempted to develop a specialized ontology that is not linked to a foundation ontology. As a result, specialized ontologies with no linkages to any of the major foundation ontologies have proliferated.

The above considerations suggest the following desiderata for a foundation ontology that can be adopted and used by a large enough community to serve as a *de facto* standard of meaning:

- the core set of concept representations required to use the ontology effectively should be as small as possible, but sufficient to support specification of any specialized concept meaning
- the ontology should be fully public and developed by an open procedure, so as to permit alternative logically compatible views of entities; it should be maintained by an open process and allow additions as needed to represent new topics;
- there should be a powerful intuitive natural language interface, capable of determining whether (1) representations of specific concepts are already present in the core foundation ontology or in some public extension, or (2) if not, to list the elements in the ontology closest in meaning
- the ontology format should have the expressiveness of at least FOL
- there should be several open-source substantive applications demonstrating the usefulness of the ontology
- extensions to the core, with logical specifications of concepts based on combinations of the core concept representations, should be maintained and freely available, in the manner of Java library packages, to minimize the need for creating new definitions.

In order have a *de facto* standard of meaning, it is not necessary to have universal agreement to use only ***one*** foundation ontology; it is only necessary that ***some*** foundation ontology have a user community large enough for third-party vendors to have incentive to develop utilities that make the standard easier to use, and to develop applications that demonstrate its utility. It should also have a sufficiently wide community of users that research groups will have an incentive to use it as the standard of meaning through which they can transfer information from diverse separate applications, each using different forms of intelligent information processing.

III. The COSMO project

A. Origin

The COSMO ontology [12] is currently being developed to serve as a fully public foundation ontology that contains representations of all of the 2100 words in the LDV, with the intention of serving as a broadly acceptable CDV. COSMO (COmmon Semantic MOdel) was initiated in 2005 [13] as a project of the Ontology and Taxonomy Coordinating Working Group [14], a working group of the Federal Semantic Interoperability Community of Practice. The origin of COSMO is discussed in more detail in [15]. In early 2008 the project adopted the current goal of representing the LDV. Developing the ontology as a CDV promises to furnish a foundation ontology that has all of the elements (types, relations) needed to build representations of any concept of interest in any application, yet be small enough to be usable without an extended learning period. The goal in effect is to identify the *smallest* foundation ontology that is sufficient to serve as the basis for broad semantic interoperability. Such a foundation ontology will contain representations of the essential units of meaning that can be combined to represent any specialized term or concept of interest in applications.

B. Project phasing

COSMO is proceeding in several phases. The first phase, expected to be complete within 3 months, is to create a representation of all of the words in the LDV, in an OWL format [16]. The expressiveness of at least pseudo-second-order logic (a FOL in which variables can represent relations or assertions) is required for some applications such as Natural Language understanding. The plan is therefore to maintain an OWL version, but convert it automatically to a Common-Logic (CL) compliant language such as KIF or IKL. This will require representing rules, functions, and higher-arity relations in the OWL format.

When the COSMO ontology has the full set of LDV words represented, it will be tested for its ability to serve as a CDV, by creating representations of several sets of specialized concepts and discovering how many new fundamental concept representations need to be added to the foundation ontology. It is estimated that this first version will contain over 7500 types (OWL classes), over 700 relations, and over 1000 restrictions that constrain the meanings of the elements.

The COSMO itself is not expected to be adopted without change as a common foundation ontology. The main purpose of this project is to demonstrate the feasibility a Conceptual Defining Vocabulary as an effective basis for semantic interoperability. A CDV that is widely accepted is likely to arise only from a collaborative effort by a broad consortium of ontology builders and users, as well as developers of other knowledge representation constructs such as the NIEM. More than one CDV may eventually find wide use, but the number of such ontologies is likely to be smaller than the number of operating systems, because the greater number and complexity of primitive data structures required for a CDV is larger than those manipulated by operating systems.

C. Criterion for Success

The criterion for determining whether the COSMO can serve as a starting CDV will be based on the number of new primitive ontology elements that must be added to the COSMO in order to represent groups of new terms or concepts from additional specialized topics. It is expected that ***some*** additional primitive elements (types, relations) will be need to be added to the COSMO as knowledge in diverse fields is represented. To function as an effective CDV, what is required is that the number of such new primitives added to the ontology will decrease asymptotically as each successive block (e.g. of 500) of new terms is represented using the foundation ontology. Such statistical evidence that there is ***some*** limit to the number of new terms that must be added will help answer the two questions, of whether there is ***any*** limit to the number of basic elements required for the CDV, and if so, approximately what is that number.

D. Allowance for Multiple Viewpoints

Essential to its role in enabling semantic interoperability is that COSMO must be inclusive of all logically compatible views, so as to permit translations among all of the representations used in applications. This means that wherever different ontologists prefer different means of representing a concept, both alternatives are included, with a translation rule (e.g. "bridging axioms") that automatically converts from one view to the other. An example would be the concept of "mother" which is represented in some ontologies only as a relation ('isTheMotherOf'), and in others as the type (class) 'Mother'. The COSMO OWL version can include both representations, but the automatic conversion of such alternative views will often require that rules be used, and will be possible only in the more expressive common-logic format.

Using an ontology representing multiple views could lead to inference that is less efficient than with a more restrictive representation. However, it is expected that multiple alternative representations will be needed only for interoperability among applications, and individual local applications will not use the full ontology, but will select out only those elements required for the local application. In this way, full semantic interoperability can be achieved among applications, without sacrifice of efficiency.

REFERENCES

[1] Cliff Goddard, Bad Arguments Against Semantic Primitives, *Theoretical Linguistics*, Vol. 24 (1998), No. 2-3: 129-156. (Available online at: http://www.une.edu.au/bcss/linguistics/nsm/pdfs/bad-arguments5.pdf)

[2] Longman Dictionary of Contemporary English, Longman Group, Essex, England (New Edition,1987)

[3] Guo, Cheng-ming (1989) *Constructing a machine-tractable dictionary from "Longman Dictionary of Contemporary English"* (Ph. D. Thesis), New Mexico State University.

[4] Guo, Cheng-ming (editor) *Machine Tractable Dictionaries: Design and Construction*, Ablex Publishing Co., Norwood NJ (1995).

[5] Yorick Wilks, Brian Slator, and Louise Guthrie, *Electric Words: Dictionaries, Computers, and Meanings*, MIT Press, Cambridge Mass (1996).

[6] OpenCyc: http://opencyc.org/

[7] http://www.ontologyportal.org/

[8] See: http://www.loa-cnr.it/DOLCE.html

[9] Pierre Grenon, *BFO in a Nutshell: A Bi-categorial Axiomatization of BFO and Comparison with DOLCE*, IFOMIS report 06/2003 (2003). Available at: http://www.ifomis.uni-saarland.de/Research/IFOMISReports/IFOMIS%20Report%2006_2003.pdf.
See also : http://www.ifomis.uni-saarland.de/bfo/

[10] See: http://www.niem.gov/

[11] DoD Core Taxonomy: http://www.dtic.mil/dtic/annualconf/conf05-Dickert.ppt

[12] http://micra.com/COSMO/COSMO.owl

[13] http://semanticommunity.wik.is/Federal_Semantic_Interoperability_Community_of_Practice/Work_Group_Status/Ontology_and_Taxonomy_Coordination/COSMO_Common_Semantic_Model

[14] http://semanticommunity.wik.is/Federal_Semantic_Interoperability_Community_of_Practice/Work_Group_Status/Ontology_and_Taxonomy_Coordination

[15] http://micra.com/COSMO/COSMOoverview.doc

[16] The OWL Web Ontology Language Reference: http://www.w3.org/TR/owl-ref/

ICD Wiki – Framework for Enabling Semantic Web Service Definition and Orchestration

Dean Brown, Dominick Profico
Lockheed Martin, IS&GS, Valley Forge, PA

Abstract – As Net-Centric enterprises grow, the desire to rapidly define and build reusable services and create new business processes through the combination of services and workflow will grow. Semantic Web Services is one approach to facilitate automated mediation between services based on semantic understanding of the services. Lockheed Martin is investigating the use of a wiki with an underlying RDF data model to provide a collaborative framework to define services, document services, manage ontology models, and quickly build composite services.

I. Net-Centricity, SOA, and Web Services

> ***Net-Centric:*** *Participating as a part of a continuously-evolving, complex community of people, devices, information and services interconnected by a communications network to achieve optimal benefit of resources and better synchronization of events and their consequences.*[i]

A. Net-Centricity

The following two principles are what makes Net-Centric different from how we usually build systems:

- Openness. Information systems can communicate across traditional system and enterprise boundaries in an open-ended ways.
- Dynamic Interaction. The capability to dynamically change the interaction and organizational scope at run-time versus at system development time.[ii]

Service-oriented architectures (SOAs) implemented with web services provides an open, standards-based approach to implementing capabilities that can be dynamically linked together to implement a business process. The movement towards SOA and web services allows service providers to provide high-value capabilities and services without necessarily knowing the service consumer. To optimize the value of these services, service providers need to design and build services that are reusable (as agnostic as possible to a specific implementation) and as stateless as possible (scalable and more independent).

B. Composite Web Services

As the enterprise inventory of reusable web services grows, the desire to build Composite Web Services that leverage these services to quickly support new business processes and user-desired functionality grows. Composite Services logically chain multiple web services together, ideally using an execution language like WS-BPEL or Google Mashup that can execute the service without requiring software compilation by software developers.

However, based on the heterogeneous nature of web services, linking web services together where data formats, names, units, and message formats are different requires an integrator knowledgeable about the specifics of each service; which is usually a software developer.

II. Semantic Web Services

> *The vision of Semantic Web services is to describe and annotate the various aspects of a Web service using explicit, machine-understandable semantics, enabling the automatic location, combination, and use of Web services.*[iii]

A. Semantic Web Service Overview

The goal of our research on the IntegrationWare IRAD project is to make service orchestration more in the spirit of the net-centric and Web 2.0 paradigm by allowing Composite Services to be built quickly and easily by end-users in a familiar environment in an intuitive, drag-and-drop user interface. Semantic Web Services provide the foundation to performing automated data-level mediation (matching dissimilar data names, formats, units) between services.

This is done by requiring the web service providers to perform a one-time mapping of their web service to a set of ontology models, as well as documenting additional information regarding the functionality of their services. The ontology models either pre-exist (developed specifically for a particular domain), or are modified as needed to support the web services. Once the web service-to-ontology model mappings are complete, the mapping is converted into a machine-readable format that will be used to facilitate the discovery of services and automated data mediation between the services.

B. Ontology Models

In order to create semantic web services, several different ontology models need to be created: at least one domain model, a units model, a transformation model, and a schema model.

The domain model(s) documents the entities, their relationships, and their properties that are relevant to a particular domain. For example, for the intelligence community domain model, some entities could include ISR assets, sensors, products, reports, tasking, geospatial locations, collection plans, observations,… Ideally, the domain model is built prior to performing the web service mapping to help identify the desired set of web services to be built or obtained. However, as new web services are used that don't map to existing entities and properties, then the domain ontology model will grow.

The units model documents units for values such as distance, area, volume, mass, temperature,… and how to transform between them.

The transformation model documents known "generic" transformations between different data types that aren't units of measurement and aren't associated with a particular entity. For example, a transformation service that can transform from lat/long coordinates to MGRS coordinates would show a relationship between lat/long and MGRS.

The schema model documents the message syntaxes to support message transformations between services.

C. Web Service Mapping to Ontology Models

Our primary short-term goal in developing Semantic Web Services is to support data-level mediation between web services. Because web services can be developed by a wide range of producers that don't build their services with a common data interface model in mind, many services that reference the same data can have different data formats (strings, ints, doubles,…), different data names (asset_id versus AssetId), different data units (meters versus feet, MGRS versus lat/long), and different data structures or groupings of data.

The mapping of web service interface elements (data inputs and outputs) to ontology model object properties unambiguously "defines" that data element in terms of "what" it is (domain model), the data format (schema model), and data unit (unit model) in a machine readable format (see Figure 1). This facilitates automated data transformations to address all these data matching issues.

Once the mapping has been completed, the mapped relationships are converted into a machine-readable format.

D. Design-Time Service Composition

When multiple web service data interface elements are mapped to the same ontology model object property, they are declared to be semantically "equivalent"; meaning a mapped web service output element can be mapped to an equivalent web service input element regardless of data type, name, unit, or data structure if the appropriate

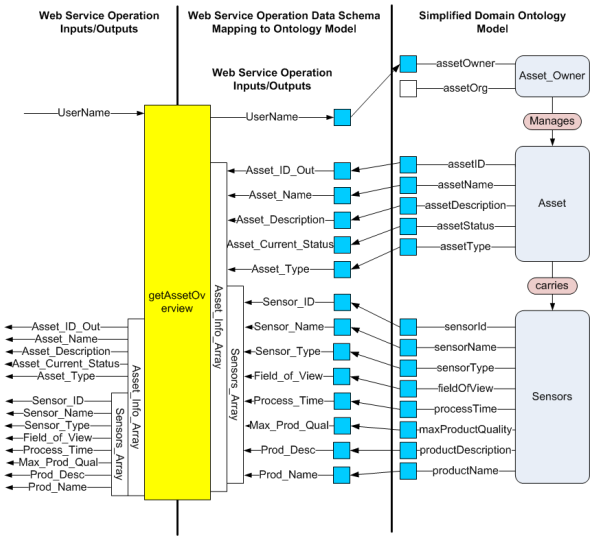


Figure 1. Web Service to Domain Ontology Mapping Example

mapping services are supported (ex: we know how to transform meters to feet).

For example, suppose a user wanted to create a composite service that computes how far an ISR asset is off plan from it's current position. This might require the composition of two specific web services like GetAssetInfo (to get position of identified asset in lat/long/elev) and GetISR_AssetPositionOffset (to take the position of the identified asset (in MGRS coordinates and elev) and a plan ID) to compute the offset (see Figure 2).

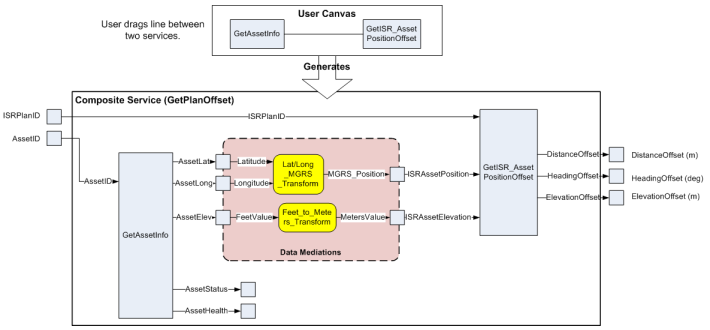


Figure 2. Example Composite Service Generation

If both services are mapped to a domain model (which identifies AssetElev and ISRAssetElev as equivalent), a units model (which knows how to convert feet to meters), and a transformation model (which knows how to convert lat/long to MGRS), then the user can simply drag each service to the canvas and connect them together. The underlying system will use the ontology model mappings to determine what outputs from the first service map to

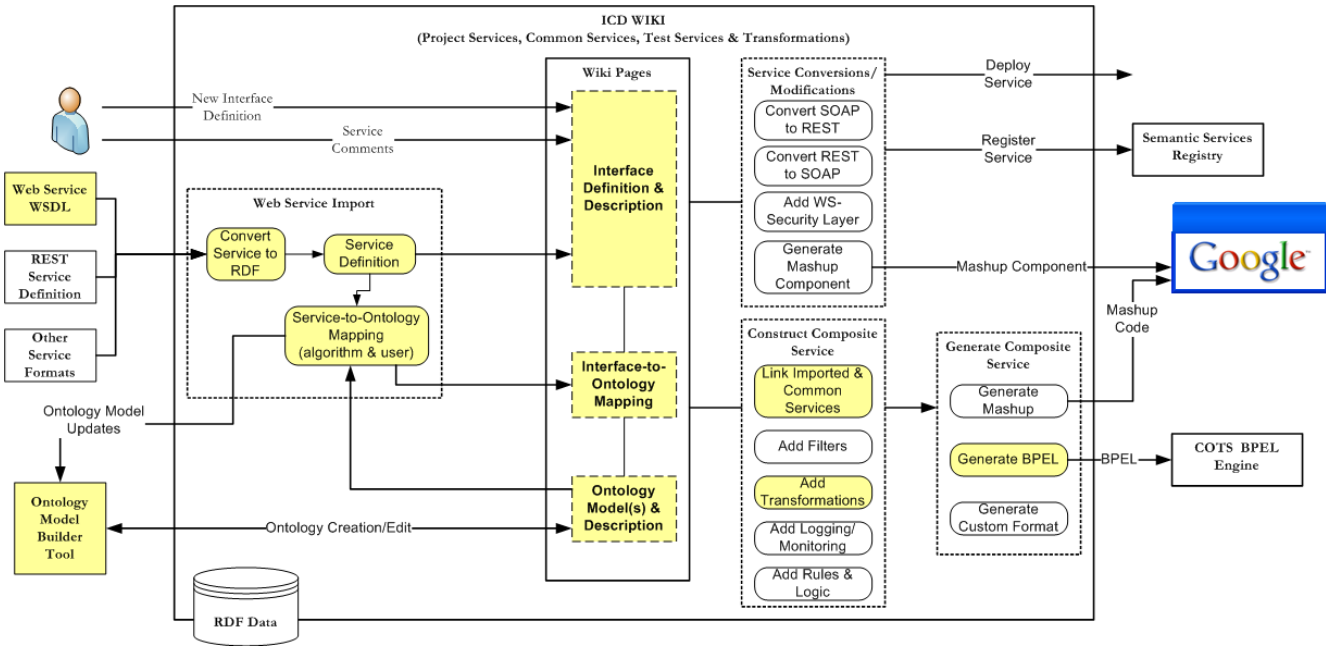


Figure 3. ICD Wiki Functional Diagram

inputs to second service (equivalance). If data transformations are required, then the appropriate transformation services are automatically identified and inserted between the user linked services.

III. ICD Wiki

A. Service Design, Discovery & Composition Framework

We provide a framework to build semantic web services that is intuitive for users by using the Wiki paradigm. The ICD wiki framework allows users to import web services, perform the service-to-ontology model mappings, and generate/convert/modify services for export. It also allows users to "define" desired web services and to provide feedback regarding the usability of existing web services.

See Figure 3 for a functional diagram of the ICD wiki framework vision. The yellow shaded boxes show where we have done development so far. The following sections provide more detail regarding the implemention of the ICD Wiki.

B. Service Import Process

Importing existing service definitions is a key component of increasing the usefulness and adoption of the ICD Wiki system. The wiki provides a user interface designed to compliment existing common user interfaces on the Web. Through this interface, a user is able to select a Web Service Description Language (WSDL) file to be imported into the ICD Wiki Semantic Store. This WSDL file can be either located locally on the user's workstation or any network reachable URL.

The import process places copies of all WSDL and related schema files onto a "Resource Bus", making all files available via a standard URL reference. Co-locating each of the required files simplifies the task of inspecting the interface files and validating references.

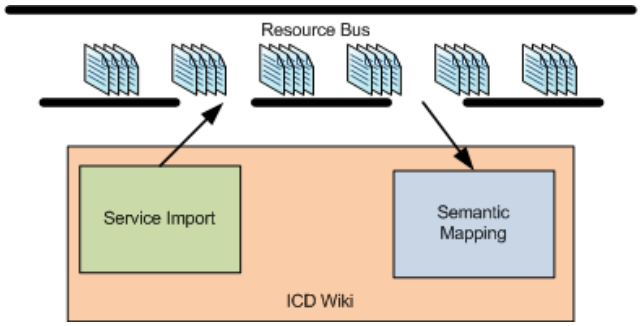


Figure 4. ICD Wiki / Resource Bus Interaction

After replicating the necessary files to the Resource Bus, the primary file is inspected to determine the format and version. WSDL v1.1 files are converted to WSDL v2.0 in order to facilitate the transformation and mapping stages. Conversion of the WSDL takes place via an Extensible Stylesheet Language Transformation (XSLT). The XSL file utilized to perform this transformation was acquired from the W3Civ. WSDL v2.0 specifications require no additional conversion before being transformed into semantic representations.

C. Semantic Service Transformations

There are multiple aspects to converting a service specification into a semantic representation suitable for storage in the ICD Wiki Semantic Store. The OWL-S definition allows for a service specification to be represented semantically, but it does include the ability to represent the underlying syntactic structure of the messages passed into and out of the service's respective operations. Semantic representation of the syntactic structure is required in order to accurately determine how service interfaces can be mapped to one-another. In order to support this level of detail we developed a small model to represent XML Schemas.

This model represents each of the most commonly used XSD elements as semantic entities. We are then able to create individuals of those class elements which represent the imported schema. Once the full schema for the service has been re-represented semantically, that semantic data is inserted into our semantic data store. In addition to automatically transforming the syntactic schema into a semantic one, the system, at this stage, provides the user with the ability to assign "units" to entities in the semantic representation of the service schema. Early in our design process, it was determined that support for unit conversion among service operations would be an integral part of enabling service composition. A simple model was built to represent the abstract concept of units and unit transformations, both simple and complex. If a user chooses to include unit designations for various elements in the schema representation, those units will be taken into consideration during any future composition sessions and used to facilitate additional transformations where appropriate.

At this stage, additional user input is required in order to fully understand the relationship between the imported schema and the known domain-specific models in the ICD Wiki thus far. Automated determination of this relationship is not available at this time in the prototype, so we present the user with an interface to allow point-and-click mapping of the imported elements to one or more domain models. A single entity can be mapped to multiple entities across multiple domain models, further enhancing the intrinsic knowledge within the semantic data store.

Mapping from complex objects in the syntactic schema to entities and entity –types with in the domain models facilitates the systems ability at later stages to determine "assignability" between two service interfaces.

"Assignability" is a determination made by applying semantic rules to the ICD Wiki service domain model and other domain models to generate an entailment. This entailment is used to ensure that an output message for a service's operation is "assignable" to the input message of another service, during service composition. Entities are assignable in many ways, and can be chained together to support the integration of services without those services supporting the exact same mapping.

D. Service Composition

Composition of services into larger, more robust services is one of the primary drivers of the ICD Wiki concept. Utilizing an off-the-shelf product called mxGraphv, we have a built a canvas-style, drag and drop interface for service composition. The available set of services are retrieved from the semantic store for inclusion in the new composite service. The composition tool makes use of common Web 2.0 tenets to allow a user to drag a service from the available service pool and place it on the canvas. Once on the canvas, a user can draw linkages between service inputs and outputs. During this process, the underlying architecture will continually check for assignability between the services linked.

Assignability is the determination "IF" two services can be linked together. During composition, this is enough information to allow the user to connect two services without being burdened with type and meaning mapping. The necessary transformations and conversions are added during the composed service generation phase.

After a user has completed laying out the composed service as desired, the canvas will be examined and the generation of the necessary work flow will be begin. The current system supports work flow generation as Business Process Execution Language files. Built in to the generated workflow code is all of the necessary unit and type transformations to combine the services. The data is not transformed into a semantic format during execution of the workflow, but rather the semantic data is used to determine what values can be assigned to what parameters, so a direct assignment is done between parameters inside the workflow. Multiple assignments and transformations may be necessary to progress from one service parameter to another, but these complexities are completely hidden from the user. These composed services are made available within the ICD Wiki for further composition and integration as needed by additional users.

E. Wiki Page Generation

Generation of pages inside the ICD Wiki takes place during the service and model import capabilities. During an import of either a model definition file or a service definition file, the necessary wiki pages are automatically created to support the human-readable aspect of the ICD Wiki concept.

Every wiki page in the ICD Wiki is capable of displaying a side-bar style component which shows all of the known semantic relationships between the entity represented on the current page and other entities in the semantic store. Using the sidebar, users are able to

explorer related entities in a traditional point-and-click, web format.

For imported services, a page is created to represent the service document as a whole, as well as pages for each

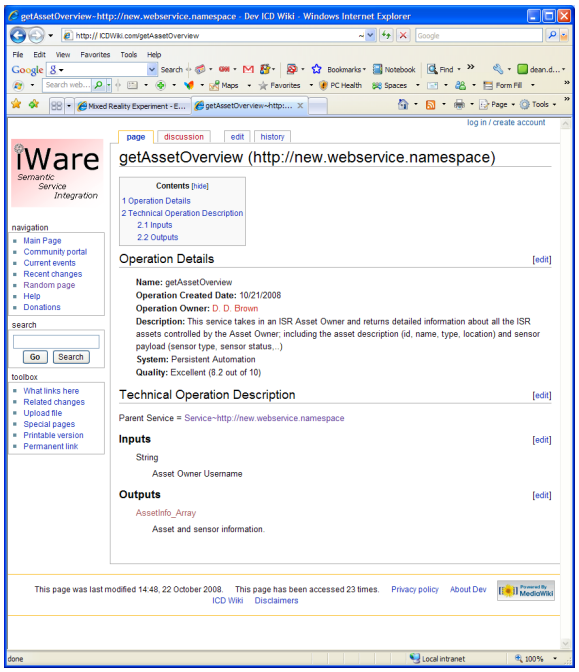


Figure 5. Operation Wiki Page Example

operation and input/output for those operations. The created pages contain little textual content, but instead there are custom wiki tags injected into the page text in order to support dynamic generation of various page sections, including the cross-linking of operations and types belonging to the service.

In addition to the semantic side bar outlined above, every imported model has a page from which a user can start exploring an imported model. This page provides access to the Domain Model Explorer.

F. Domain Model Explorer

The Domain Model Explorer is a tool built directly into the ICD Wiki system which supports a user in their exploration of the available domain models (see Figure 6). On each domain model's starting page, a "web" of semantic entities is displayed, allowing the user to find relationships amongst the various entities in the model. Further, each entity is accessible as a drill down point in order to find further relationships with in the model.

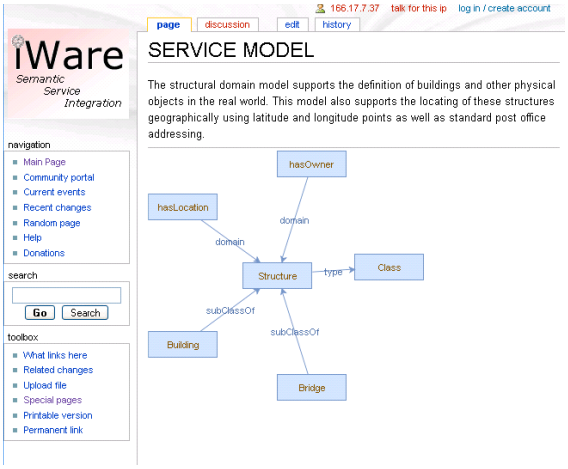


Figure 6. Domain Model Start Page

IV. Conclusion

We believe that the ICD wiki framework and semantic web services will allow all users to better leverage the growing list of available web services, intuitively define the services they want built, and provide feedback on the usability of all services..

V. References

[i] 'Net-Centric', *Wikipedia, The Free Encyclopedia*, http://en.wikipedia.org/wiki/Net-centric, (accessed 10/2/08)

[ii] The Essence of Net-Centricity, Hans Polzer

[iii] Dieter Fensel, Holger Lausen, Axel Polleres, Jos de Bruijn, Michael Stollberg, Dumitru Roman, John Domingue. *Enabling Semantic Web Services*. Berlin Heidelberg: Springer-Verlag, 2007

[iv] http://dev.w3.org/2006/wsdlConverter/wsdl11to20.xsl

[v] http://www.mxgraph.com/

Intelligence Analysis and the Semantic Web: an Alternative Semantic Paradigm

Brock Stitts

Abstract— **Intelligence analysis involves gathering data from multiple and diverse sources. The Internet provides a monstrously large set of diverse sources. It is so large and diverse, in fact, that the project of manually gathering data from all the potentially useful sources is not feasible. This is where the Semantic Web comes into play. With the Semantic Web, web pages are given a machine understandable content such that web agents can search the internet and perform tasks autonomously. A key property of this machine understandable content is that it must provide for semantic interoperability between the various web pages. The Semantic Web, as its chief advocate, Sir Tim Berners-Lee, admits remains "largely unrealized." The thesis presented here is that by going back to the foundations of semantics, we can generate a new hypothesis as to how the Semantic Web can be realized. In particular, centering on activities (or services) instead of a trying to build a global upper ontology will more effectively cope with semantic interoperability issues and thus will help realize the Semantic Web.**

Index Terms—**intelligence analysis, semantic web, ontology, semantics**

I. Introduction

Applied Systems Intelligence, Inc. (ASI) has developed a methodology for intelligence analysis which involves evaluation of a threat via a parameterized Bayesian belief network (BBN). "Feeding" this BBN to build a threat analysis involves actively seeking evidence to confirm or deny parameterized hypotheses. An outstanding data source for this analysis would be the Semantic Web. With it, web pages are given a machine understandable content so that web agents can search the internet and perform tasks, such as retrieving evidence, autonomously. A key property of this machine understandable content must be to provide for semantic interoperability between the various web pages. The Semantic Web, as its chief advocate, Sir Tim Berners-Lee, admits remains this "largely unrealized."[1] The thesis presented here is that by going back to the foundations of semantics, we can generate a new hypothesis as to how the Semantic Web can be realized. First, we begin with a brief discussion of semantics.

II. Two Views on Semantics

- Meaning is denotation: words are defined by reference to the objects or things which they designate in the external world or by the thoughts, ideas, or mental representations that one might associate with them
- Meaning is use: words are defined by how they are used in effective, ordinary communication.[2]

If one inquires as to how the denotation gets set up between a word and its object, one finds that the answer is that it is by virtue of using the word in particular contexts that it receives its denotation. In other words, communication happens within the context of some human activity. It is this activity that gives words their meaning. The philosopher Ludwig Wittgenstein considers the following simple scenario (the so-called "builder's language" introduced in section two of the Philosophical Investigations):

"The language is meant to serve for communication between a builder A and an assistant B. A is building with building-stones: there are blocks, pillars, slabs and beams. B has to pass the stones, in the order in which A needs them. For this purpose they use a language consisting of the words "block", "pillar" "slab", "beam". A calls them out; — B brings the stone which he has learnt to bring at such-and-such a call."[3]

This is a simple illustration of the basic functioning of language. The words are used as "moves" in a kind of "game." Wittgenstein coined the term "language game" based on this and other examples. In general, the meaning of the parts (the words and objects of the activity) is derived from the whole (the activity). Likewise, the activity is defined in terms of its parts. This circle is referred to as the "hermeneutic circle." Another way of saying this is:

> "It (the hermeneutic circle) refers to the idea that one's understanding of the text as a whole is established by reference to the individual parts and one's understanding of each individual part by reference to the whole."[4]

Instead of seeing words as the "semantic atoms" out of which sentences are built, the semantic unit is a language game (or activity). Much further argumentation can be provided to support this view, but providing this support is the topic of

another paper. Instead, we assume it to be accurate, and generate a new approach to building the Semantic Web based on it.

III. An Alternative Semantic Paradigm

Underlying the approaches of much symbolic artificial intelligence (AI) is the use of set theoretic concepts. In such approaches, the world consists of a set of individuals. These individuals have properties. For an individual to have a property corresponds to its being a member of some set. With such a viewpoint, assertions about individuals are not relative to some context. For the approach presented here, individuals and their properties are relative. In particular, they are relative to an activity. The individuals and their properties are components of an activity. While these individuals and properties may be used in other activities, there is no guarantee of synonymy across them. It is the hypothesis here that the assumption of synonymy across language games leads to much erroneous reasoning. In general, the long chains of inference found in some traditional AI systems will be problematic because they will cut across multiple activities and so will contain invalid inferences. Metaphorically, they will be using apples to infer things about oranges.

A. Application to the Semantic Web

As noted above, semantic interoperability between web services (or agents) is a prerequisite of the Semantic Web. The general idea on how to do this is to create metadata that accompanies web pages. This metadata would contain the semantic contents of the web page. The representation of the metadata would use the web ontology language (OWL). The assumption by Berners-Lee is that the web agents would use an inference engine to reason about this semantic content.[5] The approach here reverses the implicit denotational semantics of Berners-Lee's approach; instead, a web agent knows the meaning of the name and parameters of a service if it knows how to use the service. The semantics of a language game are contained in the game itself. With the Semantic Web, however, different language games must interact. The problem of creating the Semantic Web is then essentially a matching problem. A web agent would try to find an appropriate web service to accomplish whatever task it needed to perform. To do this, it must match up its service request with a web service that can fulfill that request. This matching problem is difficult because any solution must also solve the semantic interoperability problem. This problem comes about in two ways. First, the requester and provider may use different symbols that mean the same thing. The second, and more difficult problem, occurs when they use the same symbol but mean different things by that symbol. To make matters worse, both problems can occur with a single match.

This matching problem has no easy solution. What we outline here are a proposed set of techniques to solve it.

- Use Google-style page ranking as part of the matching algorithm. This is clearly effective to some degree, but one need only attempt using Google to perform Berners-Lee's example of the Semantic Web in action[6] to see why Google only is not sufficient. The goal of this step is really just to generate a set of candidate agents.
- Use case based reasoning (CBR) methods. If one thinks of a web service as a "solution" and a web agent as having a "problem" it is trying to solve, we see that there is a strong analogy between CBR and the matching problem.[7]
- Perform verification. If a web agent has an "answer key" for selected "problems," it can use this key to verify that it has used a web service appropriately. Likewise, if the web service provides a sample usage set, this can also be used for verification. The importance of this step cannot be understated. This is a key part of cognition and scientific reasoning. In cognition, the subject generates expectations based on his or her understanding of a situation. If these expectations are met, that understanding is verified.
- Rather than just providing a service's name, input parameters, and output parameters, provide for instructions (in the form of metadata) on how, why, when, and who should use the service. Although these "instructions" would be prone to ambiguity just as all symbols are, they provide a richer data set to use in matching.

Just as the Web gradually grew as content providers built more content, the approach here would lead to a gradual growth of the Semantic Web. In fact, every piece of this solution would evolve over time. Clearly much work needs to be done to flesh out the details. ASI is currently at work doing this so as to extend its intelligence analysis capabilities.

IV. Conclusion

If the thesis approach presented here is correct, much of the work in deriving an upper ontology will not be all that productive. With the IEEE suggested upper merged ontology (SUMO), for example, there are bound to be numerous cases where its logical axioms are ambiguous; they apply in some contexts but not others. Rather than solving the problem of how to keep chains of reasoning consistent, the approach here is not to perform them. The Semantic Web has two components: the Web and semantics. Semantics for natural languages are captured in dictionaries. However, dictionaries are descriptive. Neologisms are generated when new situations arise that call for them, and are created by a wide variety of language users. Likewise, the web is built "bottom up" by its numerous content providers. Having a committee to define language syntax is workable, but this does not hold for semantics. The semantics of a language is the set of uses of

that language. How to use and grow that language is best left to the users of the language.

[1] Nigel Shadbolt, Wendy Hall, Tim Berners-Lee (2006). "The Semantic Web Revisited". *IEEE Intelligent Systems*. Retrieved on 2007-04-13.

[2] See http://en.wikipedia.org/wiki/Philosophical_Investigations
[3] See http://en.wikipedia.org/wiki/Language-game
[4] See http://en.wikipedia.org/wiki/Hermeneutic_circle
[5] See http://www.sciam.com/article.cfm?id=the-semantic-web&print=true
[6] See http://www.sciam.com/article.cfm?id=the-semantic-web&print=true
[7] See http://en.wikipedia.org/wiki/Case-based_reasoning

Model Driven Ontology:
A New Methodology for Ontology Development

Mohamed Keshk
Sally Chambless
Raytheon Company
Largo, Florida
Mohamed.Keshk@raytheon.com
Sally.Chambless@raytheon.com

Abstract

Semantic technology is becoming a preferable alternative for enterprise-wide applications intertwined with interoperable information sharing, due to the distributed nature of this technology. Ontology is the cornerstone of semantic technology; therefore, a major challenge for the project team is to build a complete and consistent ontology data model that represents the correct business domain. Effective collaboration among customer and team members is essential for the creation of the correct ontology model. Equally necessary is a mechanism to automatically transform this model into ontology script.

Within today's leading organizations using semantic technology, a significant factor in business success rests solely in the hands of the ontologists. It is they alone who are responsible for building the correct ontology data model. Having no other members on the project team capable of verifying and validating the created ontologies may put the entire business at risk.

This paper describes a new methodology, "Model Driven Ontology," in which using a standard modeling activity as a key process for building ontology would effectively and efficiently enhance collaborations between different parties of the project team. This would lead to a consistent ontology model validated and approved by all members of the project team (business experts, intel analysts, DB admins, architects, ontologists, etc.).

Model Driven Ontology uses a UML object model artifact as a starting point to build an ontology data model. This model as a common ground for all team members is then systematically transformed to a formal ontology, facilitating the development of enterprise-wide information exchange and sharing, which can be uniformly developed, centrally maintained, and efficiently reused [6]. This would lead to more efficient and inexpensive information sharing between different information systems, cost effective development and deployment of information systems, and better quality decision making as a result of more timely, accurate, and complete information.

Introduction

The development of large-scale enterprise applications has become increasingly complex due to the massive growth of enterprise data and the constant changing of requirements. Semantic technology has been seen as a crucial alternative for managing this complexity by providing a solid and flexible infrastructure for information exchange, retrieval, sharing, and discovery.

As ontologies play a central role in facilitating semantic technology solutions, it is essential for business to standardize the ways ontologies are developed. The phases of ontology development include analysis, design, coding, validation, execution, and maintenance. Moreover, it is vital for businesses to keep all key players (business

experts, intel analysts, architects, DB admins, ontologists, etc.) closely involved in the development phases.
Organizations using semantic technology, including those in both governmental and private sectors, frequently hand ontology development tasks solely to the ontologists. In most cases, the consequence is a dilemma, since no other team member is capable of validating the ontology script created by the ontologists, and the business might be at great risk if the script does not reflect the correct business model. Therefore, a methodology to standardize the way ontology is developed is badly needed.

This paper sheds light on the value of modeling in the context of ontology development for enterprise applications. It shows how modeling can be an effective way to manage the complexity of ontology development [5], as it fosters better communication by overlooking implementation details that are not relevant to the overall system, and delivers robust design and assessment of requirements and architectures. Despite these virtues, mainstream ontologists have yet to take advantage of modeling in everyday practice [8].

Our approach uses a Unified Modeling Language (UML) object model as the common means for expressing ontology models. As an industry standard, UML graphic models provide a common ground for team members to better understand the business data models and elevate the level of collaboration. The result is a consistent data model, validated and approved by all team members, which leads to a more accurate ontology script.

In general, there is no one correct methodology for developing ontologies, since there is no one correct way to model a domain [2]. Ontology itself is a data model based on formal logic and greatly overlaps with a UML object model, as both share many basic concepts. While a UML object model has the concepts of classes, properties, associations, constraints, and instances, ontology has the same concepts named classes, datatype properties, object properties, restrictions, and individuals, respectively. Providing a single data model for all parties of project team will increasingly eliminate design ambiguity, reduce the complexity of the enterprise data model, and speed up the overall development.

Therefore, a UML object model can be seen as a common model for ontologists and software architects, as it enhances communication between both camps and brings other parties to the table. It also aligns the effort of building a consistent data model that is accessible and usable not only by ontologists, but also by other team members.

Model Driven Ontology Methodology

In this section, we will discuss in detail the Model Driven Ontology approach with a simple, yet complete, example [1]. The following diagram (Fig. 1) shows a UML class diagram of a purchase order example in terms of classes, attributes, enumerations, and relationships including inheritance, composition, aggregation and associations with constraints represented as cardinalities.

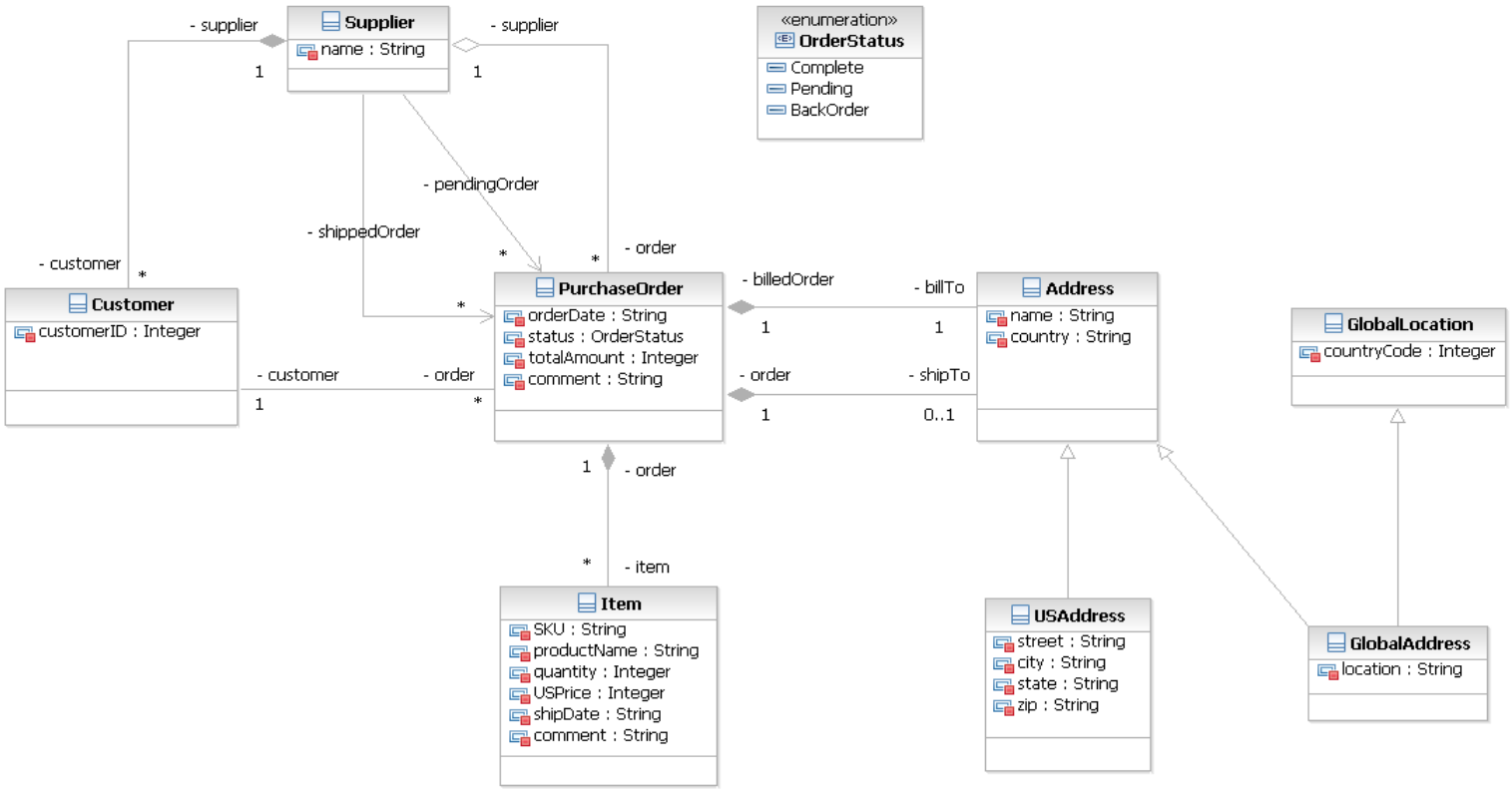


Fig-1

Our approach starts with the UML class diagram which represents the data model of a particular domain. Once the team comes up with the correct UML data model, validated and approved by all parties, we generate a UML version that is encoded in XMI by exporting the model using a standard UML tool such as RSA. We then apply transformation rules by parsing the XMI file into ontology script.

This parsing is done through Eclipse Modeling Framework (EMF) API provided by UML2 plugin [3], which is an EMF-based implementation of the UML 2.x metamodel for the Eclipse platform. The objective of this plugin is to provide a useable implementation of the UML metamodel to support the development of model processing tools, a common XMI schema to facilitate interchange of semantic models, test cases as a means of validating the specification, and validation rules as a means of defining and enforcing levels of compliance [3].

Our transformation platform is EMF which is part of the Model Driven Architecture (MDA) and is the implementation of a subset of the MDA in Eclipse platform [1]. An EMF model is essentially the class diagram subset of UML. EMF is originally based on MOF (Meta Object Facility) by OMG (Object Management Group). EMF uses XMI (XML Metadata Interchange) as its canonical form of a model definition. EMF has its own meta-metamodel called Ecore. Ecore is considered the metamodel for UML in addition to some other metamodels, such as XSD, WSDL, BPEL, etc. Ecore is located at the M3 layer of MDA paradigm and defines all kinds of metamodels located at M2, including UML. Ecore, itself, is very similar to EMOF (Essential MOF), but has Eclipse as a runtime environment.

EMF lets you define a data model in one of three formats: Java interface, XML schema, or UML class diagram, then

allows you to generate the other two formats. The most-likely scenario is to start with a UML model and generate the corresponding Java interfaces and XML schema. Our approach extends this capability by generating RDF/OWL script from the same UML model.

Building transformation rules is a joint effort between the architecture team, the ontology team and business domain experts. The expertise of these teams helps generate the correct script corresponding to the data model. For the purpose of illustrating the transformation mechanism, we have isolated a subset of the diagram (Fig. 2). The complete generated OWL script is too lengthy to include in this paper.

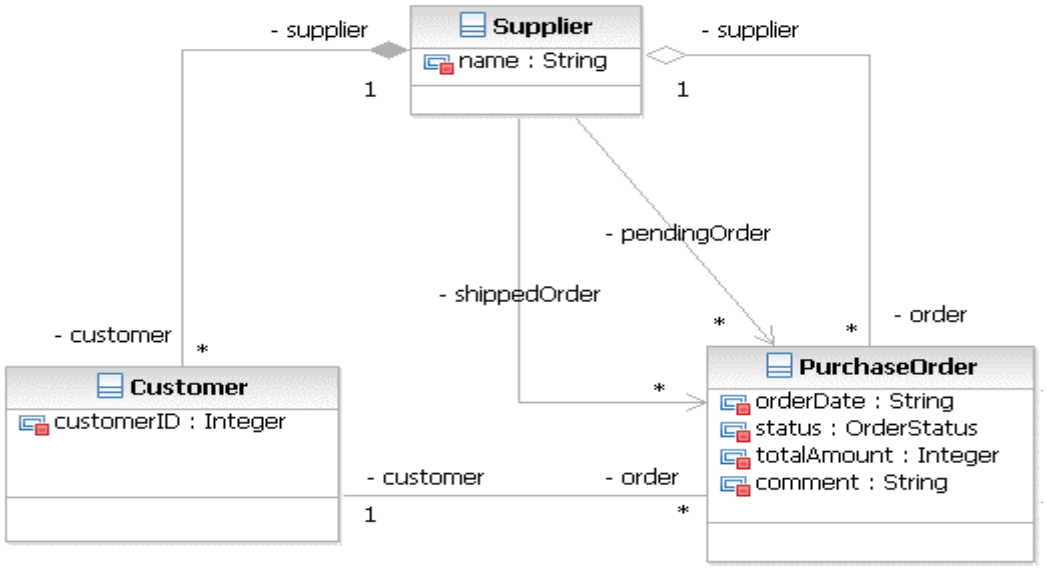


Fig-2

The following is XMI script for “Supplier” class:

```
<packagedElement xmi:type="uml:Class" xmi:id="_maCsFE3GEd2Y9cy9X2GvMA" name="Supplier">
 <ownedAttribute xmi:id="_maCsFU3GEd2Y9cy9X2GvMA" name="name" visibility="private">
  <type xmi:type="uml:PrimitiveType" href="pathmap://UML_LIBRARIES/UMLPrimitiveTypes.library.uml#String"/>
 </ownedAttribute>
 <ownedAttribute xmi:id="_maCsFk3GEd2Y9cy9X2GvMA" name="customer" visibility="private" type="_maCsDk3GEd2Y9cy9X2GvMA"
     aggregation="composite" association="_maCsKU3GEd2Y9cy9X2GvMA">
  <upperValue xmi:type="uml:LiteralUnlimitedNatural" xmi:id="_maCsF03GEd2Y9cy9X2GvMA" value="*"/>
  <lowerValue xmi:type="uml:LiteralInteger" xmi:id="_maCsGE3GEd2Y9cy9X2GvMA"/>
 </ownedAttribute>
 <ownedAttribute xmi:id="_maCsGU3GEd2Y9cy9X2GvMA" name="pendingOrder" visibility="private"
     type="_maCr5U3GEd2Y9cy9X2GvMA" association="_maCsKk3GEd2Y9cy9X2GvMA">
  <upperValue xmi:type="uml:LiteralUnlimitedNatural" xmi:id="_maCsGk3GEd2Y9cy9X2GvMA" value="*"/>
  <lowerValue xmi:type="uml:LiteralInteger" xmi:id="_maCsG03GEd2Y9cy9X2GvMA"/>
 </ownedAttribute>
 <ownedAttribute xmi:id="_maCsHE3GEd2Y9cy9X2GvMA" name="shippedOrder" visibility="private"
     type="_maCr5U3GEd2Y9cy9X2GvMA" association="_maCsLk3GEd2Y9cy9X2GvMA">
  <upperValue xmi:type="uml:LiteralUnlimitedNatural" xmi:id="_maCsHU3GEd2Y9cy9X2GvMA" value="*"/>
  <lowerValue xmi:type="uml:LiteralInteger" xmi:id="_maCsHk3GEd2Y9cy9X2GvMA"/>
 </ownedAttribute>
 <ownedAttribute xmi:id="_maCsH03GEd2Y9cy9X2GvMA" name="order" visibility="private" type="_maCr5U3GEd2Y9cy9X2GvMA"
     aggregation="shared" association="_maCsM03GEd2Y9cy9X2GvMA">
```

```
    <upperValue xmi:type="uml:LiteralUnlimitedNatural" xmi:id="_maCsIE3GEd2Y9cy9X2GvMA" value="*"/>
    <lowerValue xmi:type="uml:LiteralInteger" xmi:id="_maCsIU3GEd2Y9cy9X2GvMA"/>
  </ownedAttribute>
</packagedElement>
```

And its corresponding OWL script is the following:

```
<owl:Class rdf:about="&PO;Supplier">
  <rdfs:label>Supplier</rdfs:label>
</owl:Class>

<owl:DatatypeProperty rdf:about="&Supplier;name">
  <rdfs:domain rdf:resource="&PO;Supplier"/>
  <rdfs:range rdf:resource="&xsd;string"/>
  <rdf:type rdf:resource="&owl;FunctionalProperty"/>
</owl:DatatypeProperty>

<owl:ObjectProperty rdf:about="&Supplier;order">
  <rdfs:domain rdf:resource="&PO;Supplier"/>
  <rdfs:range rdf:resource="&PO;PurchaseOrder"/>
  <owl:inverseOf rdf:resource="&PurchaseOrder;supplier"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="&Supplier;customer">
  <rdfs:domain rdf:resource="&PO;Supplier"/>
  <rdfs:range rdf:resource="&PO;Customer"/>
  <owl:inverseOf rdf:resource="&Customer;supplier"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="&Supplier;shippedOrder">
  <rdfs:domain rdf:resource="&PO;Supplier"/>
  <rdfs:range rdf:resource="&PO;PurchaseOrder"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="&Supplier;pendingOrder">
  <rdfs:domain rdf:resource="&PO;Supplier"/>
  <rdfs:range rdf:resource="&PO;PurchaseOrder"/>
</owl:ObjectProperty>
```

Conclusion

This paper explains the benefits and values that modeling practice can offer for ontology-based applications, by treating modeling as a first class artifact, rather than documentation. In addition to presenting a single common data model that all team members can share, a mechanism is presented to generate the ontology script directly from the UML model once it is validated and proofed. In this case, the model is used not only as a diagram or a blueprint, but also as a primary artifact from which efficient script is generated by applying transformation rules.

We argue that the use of Model Driven Ontology would increasingly boost productivity, eliminate mistakes due to human misunderstanding, break the monopoly of ontologists over ontology development, and save a significant amount of development effort.

References

[1] Frank Budinsky. Dave Steinberg, Ed Merks, Ray Ellersick, and Timothy J. Grose, "Eclipse Modeling Framework", Addison-Wesley Professional, August 2003.

[2] Dragan Gasevic, Dragan Djuric, and Vladan Devedzic, "Model Driven Architecture and Ontology Development", Springer, 1st edition, July 2006.

[3] Eclipse UML2 project, http://www.eclipse.org/modeling/mdt/?project=uml2

[4] D. S. Frankel, Model Driven architecture: Applying MDA to Enterprise Computing, OMG Press, ISBN: 0471319201, January 2003.

[5] Nešić, S., Jazayeri, M., Jovanović, J., Gašević, D., "Ontology-based content model for scalable content reuse", In Proceedings of the 4th ACM International Conference on Knowledge Capture, Whistler, BC, Canada, 2007, pp. 195-196.

[6] "National Information Exchange Model - NIEM", http://www.niem.gov/

[7] dos Santos, E.S., Ralha, C.G., Carvalho, H.S., Gašević, D., "MDA-based Ontology Development: A Study Case," In Proceedings of the 19th International Conference on Software Engineering and Knowledge Engineering, Boston, USA, 2007.

[8] Stephen Cranefield, Jin Pan, "Bridging the Gap Between the Model-Driven Architecture and Ontology Engineering", The Information Science Discussion Paper Series, Number 2005/12, December 2005, ISSN 1172-6024.

Acknowledgements

CONFERENCE CO-CHAIRS:

Kathryn Blackmond Laskey, George Mason University, Fairfax, VA, USA
Duminda Wijesekera, George Mason University, Fairfax, VA, USA

SCIENTIFIC COMMITTEE:

Bill Andersen, Ontology Works, Incorporated
Selmer Bringsjord. Rensselaer Polytechnic Institute
Dennis Buede. Innovative Decisions, Inc.
Werner Ceusters, University at Buffalo
Randall Dipert, University at Buffalo
Katherine Goodier, NCI, Inc.
Kathleen Stewart Hornsby, University of Iowa
Terry Janssen, Lockheed Martin Corporation
Kathryn Blackmond Laskey, George Mason University, co-chair
Nancy Lawler, US Department of Defense
Kevin Lynch, CIA
Dan Maxwell, Innovative Decisions, Incorporated
Fabian Neuhaus, NIST
Leo Obrst, MITRE Corporation
Steven Robertshaw, UK Defence Science and Technology Laboratory
Barry Smith, University at Buffalo
Duminda Wijesekera, George Mason University, co-chair

SPONSORS:

Center of Excellence in Command, Control, Communications, Computing and Intelligence (C4I Center), George Mason University, USA

Innovative Decisions, Incorporated, Vienna, VA

Saab, Stockholm, Sweden

National Center for Ontological Research (NCOR), University at Buffalo, USA

We wish to thank the authors of all submitted papers for their time and interest in this event. An electronic version of the OIC-2008 proceedings id available at CEUR-WS (http://CEUR-WS.org).