Second International Semantic Web Conference

(ISWC 2003)

Posters and Demonstrations

20-23 October, 2003

Sanibel Island, Florida

Conference Organizing Committee

General Chair: Dieter Fensel, Institute for Computer Science (IFI), University of Innsbruck
Program Chairs: Katia Sycara, Robotics Institute, School of Computer Science, Carnegie Mellon University
John Mylopoulos, Department of Computer Science, University of Toronto
Local Chair: Jeff Bradshaw, Institute for the Interdisciplinary Study of Human & Machine Cognition, The University of West Florida
Tutorial Chair: Asunción Gómez-Pérez, Universidad Politecnica de Madrid, Spain
Industrial Program Chair Christoph Bussler, Oracle Corporation, USA
Workshop Chairs: Sheila McIlraith, Knowledge Systems Laboratory, Stanford University, & Dimitris Plexousakis, Department of Computer Science, University of Crete and Institute of Computer Science, Foundation for Research and Technology (FORTH)
Demo Chair: Jeff Heflin, Dept. of Computer Science and Engineering, Lehigh University
Sponsor Chairs: Ying Ding, Institute of Computer Science, University of Innsbruck, & Massimo Paolucci, Robotics Institute, School of Computer Science, Carnegie Mellon University
Metadata Chair: Steffen Staab, AIFB, University of Karlsruhe
Publicity Chair: Mike Dean, BBN Technologies / Verizon
Financial Chair: Jérôme Euzenat, INRIA Rhône-Alpes
Poster Chair: Raphael Malyankar, Arizona State University
Registration Chair: Atanas Kiryakov, Sirma AI, Ltd.

Posters Committee

Nancy Wiegand, University of Wisconsin
Eero Hyvönen, University of Helsinki
Ullas Nambiar, Arizona State University

Conference Sponsors

ISWC 2003 is supported by the Semantic Web Science Association in cooperation with the OntoWeb Network and the DARPA DAML Program

System Demonstrations

Posters

System Demonstrations

Demonstrator: Ontology Integration and Policy Enactment for Managing Rights Metadata

Gavin Barlas[1], Godfrey Rust[1], Matthew Quinlan[2], Martin Dow[3]

[1] Ontologyx Limited, 10 Leake Street, London SE1 7NN
[2] Network Inference Limited, 25 Chapel Street, London NW1 5DH
[3] IOKO365, 17c Curzon Street, Mayfair, London W1J 5HR

1. Introduction

This abstract describes a demonstrator using integrated metadata from Ontologyx, Network Inference's Cerebra Server and the W3C's OWL language [McGuinness et al., 2003], to enable content aggregation and rights management for multi-sourced, any-media content.

An efficient system for managing rights metadata needs to support a domain characterized by dynamism along a number of dimensions, including the changing rights of entities over the course of time, changing legal systems, and differences between jurisdictions [Pitkänen et al., 2000]. This dynamism and the need to integrate disparate syntaxes, standards and semantics suggest an ontological approach [Delgado et al., 2002]. The W3C's OWL language and a Description Logic engine provide a language and platform for metadata integration and querying.

The demonstrator provides metadata integration and dynamic inference of digital rights according to 'policies' (governing rights ownership, permissions, and royalty distribution) defined using OWL.

2. Context

'The Semantic Web is an extension of the current Web in which information is given well-defined meaning, better enabling computers and people to work in cooperation. It is based on the idea of having data on the Web defined and linked such that it can be used for more effective discovery, automation, integration, and reuse across various applications.' [Hendler et al, 2002]

Industry sectors have developed, or are now developing, their own standards and practices for metadata for conducting business electronically, using data that is often highly specialized and granular. The growth of the Web requires that these differing semantics be related. The result is 'silos' of information of varying granularity whose full value cannot be realized without extensive integration efforts.

Emerging W3C standards (RDF, RDFS, OWL) provide a foundation for structuring metadata to incorporate meaning, enabling the expression of descriptive models associated with an organization or industry body. What is needed is a systematic method of bridging the gap between the *specific* meanings of terms in sectoral or local metadata schemes, supported by tools and techniques from the semantic web community.

The demonstrator shows how the gap can be bridged. OntologyX is used to apply identities to granular and diverse meanings – in effect, the equivalent of assigning URIs to meanings – through the implementation of a rich underlying semantic model. The value of both the approach and the metadata integration using Cerebra Server is demonstrated through meeting task-related business goals.

3. Demonstrator Overview

The intended users are institutions compiling and re-publishing existing text/image/audio material – for example, in DVDs, academic coursepacks, broadcast programmes or similar collections. Usage may vary by time period, place, purpose, user group and commercial terms (eg free to students on a specific course, or for general sale). The users need to select from the material according to combinations of subject classification, source journal/book and availability of rights. Content has (a) *RightsStatements* identifying owner/source of specific rights by territory, and (b) *RightsAvailability* indicating the availability of content for specific usage.

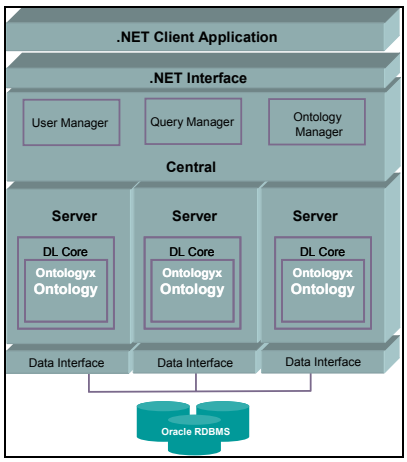


Figure 1: Demonstrator Architecture

The demonstrator references a development of the Copyright Agency Limited (CAL), an Australian company whose primary role is to provide a bridge between creators and users of copyright material. CAL is responding to the increasing demand for integrated academic coursepacks with content drawn from multiple sources by initiating a scheme for the licensing and production of online and printed "coursepacks" for academic institutions and for other training purposes.

4. Demonstrator Use Case

The use case follows the following generic steps:

1. **Find content**: User searches for material by multiple fields including subject/content classification(s), content source, territory, right type, user type, license type. User is presented with a list of results showing specific charges or other terms from the GeneralLicense.

2. **Find availability**: User selects items and, for each, terms under which he wishes to use the material, to determine whether rights may be available, and if any standard license terms are applicable.

Terms vary for different material (eg, all materials require an "Embed" right, some also require an "Adapt" or "Excerpt" right). The user is presented with a list of results showing specific license terms.

3. **Request licenses**: User selects preferred options, generating either (a) a request for license or (b) a notification of intended use for the owner. Requests/notifications are generated for the appropriate rights controller(s).

4. **Payment distribution**: Based on the license requests, payments are distributed to appropriate rights controllers, including situations where payee differs from licensor.

5. About OntologyX

OntologyX is an extensive ontology developed on the <indecs> [Rust et al., 2000] framework "context model" of semantic relationships. This model now underlies the development of a number of standard and proprietary semantic tools including the MPEG21 Rights Data Dictionary and the International DOI Foundation metadata policy.

OntologyX enables the mapping, integration and transformation of multiple ontologies of any level of complexity within a single rich structure. Its initial focus is on any-media and rights metadata, addressing the critical problems of integrating descriptive and rights metadata in complex multi-media local or distributed systems.

OntologyX has its own native class and property hierarchies, but those which are required for this demonstrator are represented in OWL.

6. About Cerebra Server

Cerebra Server is an enterprise platform architected around a commercial inference engine, originally based upon the FaCT reasoner [Horrocks, 2000].

Cerebra Server uses a Description Logic based inference engine with reasoning support for the W3C's candidate recommendation OWL, more specifically for OWL-DL. Cerebra Server is deployed as a web service for ease of integration. Its XQuery API provides a flexible, expressive, easy-to-use querying syntax.

Using Cerebra Server, the demonstrator is able to process data based on semantics without restricting the vocabulary, allowing the identification of available resources across disparate sources, creating a dynamic environment where resources are exchanged to maintain the integrity of the value-chain as new resources become available and existing resources redundant.

7. Summary

Cerebra Server and OntologyX were used to integrate multiple metadata frameworks. They were used to drive a simple end user application for the search and selection of multimedia content.

Cerebra Server was used to infer, according to OWL-defined policies, appropriate rights, notification and payment distribution, according to policies defining complex relationships between content, licensing, rights ownership and territory.

References

[Horrocks, 2000] Horrocks, I. Benchmark Analysis with FaCT. *TABLEAUX 2000*, pages 62-66, 2000

[McGuinness et al., 2003] McGuinness, D. L., van Harmelen, F. *OWL Web Ontology Language Overview*, W3C, August 2003

[Delgado et al., 2002] Delgado, J., Gallego, I., García, R., Gil. R. *An ontology for intellectual property rights: IPROnto*, ISWC Poster, 2002

[Pitkänen et al., 2000] Pitkänen, O., Välimäki, M. *Towards a digital rights management framework*, IEC, 2000

[Rust et al., 2000] Rust, G., Bide, M., *The <indecs> metadata framework: Principles, model and data dictionary*, INDECS White Paper, 2000

[Hendler et al, 2002] Hendler, J., Berners-Lee, T., Miller, E., *Integrating Applications on the Semantic Web*, Journal of the Institute of Electrical Engineers of Japan, Vol 122(10), October, 2002, p. 676-680

OntoLT: A Protégé Plug-In for Ontology Extraction from Text

Paul Buitelaar, Daniel Olejnik, Michael Sintek

DFKI GmbH
Saarbrücken/Kaiserslautern, Germany
{paulb,olejnik,sintek}@dfki.de

1 Motivation

Ontologies are views of the world that tend to evolve rapidly over time and between different applications. Currently, ontologies are often developed in a specific context with a specific goal in mind. However, it is ineffective and costly to build ontologies for each new purpose each time from scratch, which may cause a major barrier for their large-scale use in knowledge markup for the Semantic Web. Creating ambitious Semantic Web applications based on ontological knowledge implies the development of new, highly adaptive and distributed ways of handling and using knowledge that enable existing ontologies to be adaptable to new environments.

As human language is a primary mode of knowledge transfer, a growing integration of language technology tools into ontology development environments is to be expected. Language technology tools will be essential in scaling up the Semantic Web by providing automatic support for ontology monitoring and adaptation. Language technology in combination with approaches in ontology engineering and machine learning provides linguistic analysis and text mining facilities for ontology mapping (between cultures and applications) and ontology learning (for adaptation over time and between applications).

2 Approach

The OntoLT approach provides a plug-in for the widely used Protégé ontology development tool, with which concepts (Protégé classes) and relations (Protégé slots) can be extracted automatically from annotated text collections. For this purpose, the plug-in defines a number of linguistic and/or semantic patterns over the XML-based annotation format that will automatically extract class and slot candidates. Alternatively, the user can define additional rules, either manually or by the integration of a machine learning process.

2.1 Linguistic/Semantic Annotation

The MM annotation format that is used by the OntoLT system integrates multiple levels of linguistic and semantic analysis in a multi-layered DTD, which organizes each level as a separate track with options of reference between them via indices [Vintar et al., 2002]. Linguistic/semantic annotation in the MM format covers: tokenization, part-of-speech tagging (noun, verb, etc.), morphological analysis (inflection, decomposition), shallow parsing (phrases, grammatical functions: subject, object, etc.) and lexical semantic tagging (synonyms) using EuroWordNet [Vossen, 1997].

2.2 Ontology Extraction From Text with OntoLT: An Example

Consider the development of an ontology for the computer science field from a corpus of relevant text documents (i.e., scientific papers). From this corpus we could, for instance, automatically extract and represent the occurring classes of technology (e.g., "web services", "P2P platforms", "RDF parsing"). In fact, this knowledge can be extracted from such sentences as: *…university develops P2P platform…*; *... University is the first group to develop an open source P2P platform…* By selecting the **`Institute-Verb-Obj`** pattern, the system selects all subjects of semantic class **`Institute`** (i.e., *university*) and extracts the corresponding verbs. By selecting one or more appropriate verbs (e.g., *develop, design, implement*), the user is presented with a list of automatically generated Protégé classes corresponding to the extracted objects of these verbs. Additionally, each of these classes will be assigned a slot **`institute`** of class **`Institute`**.

This extraction process is implemented as follows. OntoLT introduces a class called **`Mapping`** where the user can define the structure of the new classes and instances to be extracted. Each **`Mapping`** has **`Conditions`** and **`Operators.`** The **`Conditions`** describe the constraints that have to be fulfilled to be a candidate. The **`Operators`**

describe in which way the ontology should be enlarged if a candidate is found.

3 Related Work

A number of systems have been proposed for ontology extraction from text, e.g.: ASIUM [Faure et al., 1998], TextToOnto [Maedche and Staab, 2000], Ontolearn [Navigli et al., 2003]. Most of these systems depend on shallow text parsing and machine learning algorithms to find potentially interesting concepts and relations between them. The OntoLT approach is most similar to the ASIUM system, but relies even more on linguistic/semantic knowledge through its use of built-in patterns that map possibly complex linguistic (morphological analysis, grammatical functions) and semantic (lexical semantic classes, predicate-argument) structure directly to concepts and relations. A machine learning approach can easily be build on top of this but is not strictly necessary. Additionally, like the TextToOnto system, OntoLT provides a complete integration of ontology extraction from text into an ontology development environment, but selects for this purpose (unlike TextToOnto) the widely used Protégé tool, which allows for efficient handling and exchange of extracted ontologies (e.g., in RDF/S format).

Acknowledgements

This research has in part been supported by EC grants IST-2000-29243 for the OntoWeb project and IST-2000-25045 for the MEMPHIS project.

References

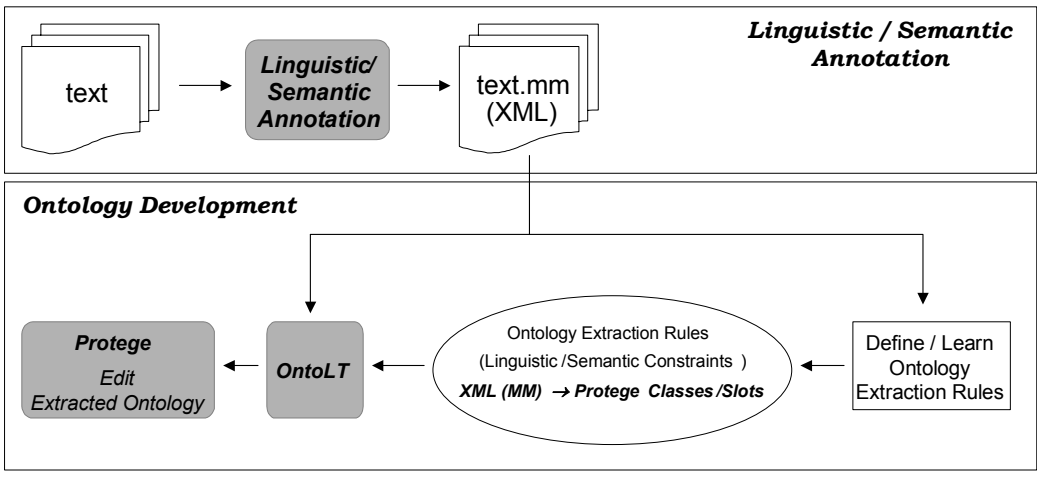
[Faure et al., 1998] Faure D., Nédellec C. and Rouveirol C. *Acquisition of Semantic Knowledge using Machine learning methods: The System ASIUM* Technical report number ICS-TR-88-16, 1998.

[Maedche and Staab, 2000] Maedche, A., Staab, S.: *Semi-automatic Engineering of Ontologies from Text.* In: Proceedings of the 12th International Conference on Software Engineering and Knowledge Engineering, 2000.

[Navigli et al., 2003] Navigli R., Velardi P., Gangemi A. *Ontology Learning and its application to automated terminology translation* IEEE Intelligent Systems, vol. 18:1, January/February 2003.

[Vintar et al., 2002] Vintar Š., Buitelaar P., Ripplinger B., Sacaleanu B., Raileanu D., Prescher D. *An Efficient and Flexible Format for Linguistic and Semantic Annotation* In: Proceedings of LREC, 2002.

[Vossen, 1997] Vossen P. *EuroWordNet: a multilingual database for information retrieval*. In: Proc. of the DELOS workshop on Cross-language Information Retrieval, March 5-7, Zürich, Switzerland.



Towards a Semantic Enterprise Information Portal - a Demo

Emanuele Della Valle, Paolo Castagna and **Maurizio Brioschi**
CEFRIEL - Politecnico of Milano
Via Fucini, 2 - 20133 Milano - Italy
{dellava, castagna, brioschi}@cefriel.it

1 Introduction

Knowing what you know is becoming a real problem for many enterprises. Their intranets are full of shared information, their extranet support a flow of data both with suppliers and customers, but they have lost the integrated view of their information. Thus finding information for decision taking is every day harder. A comprehensive solution to this problem should provide at least an answer to the following questions: What information do we have? Where is it? How did it get there? How do I get it? How can I add more? What does it mean?

Portals, in particular Enterprise Information Portals (EIPs), some years ago have been brought into the limelight for their ability to address these questions by giving a unique and structured view of the available resources. However EIPs cannot be considered a final solution, because they do help people in managing the information, but they still require a huge amount of manual work. So, we believe that using state-of-the-art web technologies will not be sufficient in the immediate future, since the lack of formal semantics will make it extremely difficult to make the best use (either manually or automatically) of the massive amount of stored information and available services.

2 The concept

Soon enterprises would be able to build "corporate Semantic Web" represented by services and documents annotated with metadata defined by a corporate ontology. Thus they will need to update their EIPs in order to cope with ontologies and metadata. They will need a *Semantic EIPs*.

The innovative idea, first proposed by [Maedche *et al.*, 2001], is straightforward: can we use metadata defined by ontologies to support the construction of portals? And if so, does it help? Even if it might appear as a radical new departure actually it is not. On the contrary it is the bringing together of existing and well understood technologies: *Web Frameworks* (as Struts, Jetspeed, etc.) that implement Model-View-Controller design pattern, *WWW conceptual models* (as WebML [Ceri *et al.*, 2000]) that are proposals for the conceptual specification (using extended E-R models) and automatic implementation of Web sites, *Ontologies* to model the domain information space, the navigation, the access and the presentation, and *Metadata* to make resource descriptions available to machine in a processable way.

On the one hand, concerning modeling, we have decided to follow an approach similar to those adopted in WWW conceptual modeling. We model separately the domain information space, the navigation and the access. The *domain information model* (in this case the corporate ontology) is a shared understanding of the information present in the corporate semantic web. Its design is completely decoupled from the semantic EIP design. Therefore the semantic EIP cannot assume any "a priori" agreement except the use of a common set of primitives (e.g. OWL). However, if we want to access the corporate semantic web using a semantic EIP we need to define at least some *upper terminology*, known by the semantic EIP, that can be employed in defining both the navigation and the access model. The *navigation models* represent the heterogeneous paths the homogeneous categories of users can adopt in traversing the corporate semantic web. They should be built by *mapping* the corporate ontology terminology to the navigation upper terminology. Finally, the *access models* represents collections of resources not strictly homogeneous, highly variable and sometimes even related to a specific user, a sort of *views*. They can be built via *mapping*, too. But they might require also to explicitly draw some new relationships as well as to add ad-hoc resources .

On the other hand, concerning presentation, we have chosen that, when users retries a resource present in the corporate semantic web, the semantic EIP *insert* it in a *navigation panel* that contains automatically generated links to the related resources. In particular, we propose to place in the navigation panel of a semantic EIP three different kinds of links: *Access point links* that render, using one of the access models, a sort of views to guide the user in accessing the information, *categorized links* that render, using one of the navigation models, a set of boxes populated with links that are the result of a simple property-based query over the metadata describing the retrieved resource, *metadata links* that provide an intuitive navigation from and to the retrieved resource following the metadata used to describe it.

3 An early proof of concept

In order to proof this concept, we have built a first prototype of a semantic EIP (an on-line demo is available at `http://seip.cefriel.it`). It is a servlet-based application that uses Velocity for implementing the model-view-

controller pattern and RACER [Haarslev and Moller, 2001] as reasoner. It "understand" RDF, RDFS and OWL property characteristics (owl:inverseOf, owl:TransitiveProperty, owl:SymmetricProperty). Moreover we assume that it "knows" two simple ontologies whose terms describe both the navigation and the access of a generic portal. The *navigation ontology* defines only a symmetric property, `related`, and two transitive properties, `contains` and its inverse `contained`. The *access ontology* defines a class, `Home`, and four transitive properties: `next`, `down` and theirs respective inverses `prev` and `up`. They represent a first draft of the introduced navigation and access upper terminology. We kept these two ontologies explicitly as simple as possible, but still rich enough to be useful in proofing the concept.

Metadata links
The prototype, "understanding" RDF and RDFS, can process the metadata that describe the retrieved resource, generating links according to the following schema:

```
CEFRIEL[Organisation] hasUnit eTECH[Unit]
Brioschi[HeadOfUnit,Person] worksFor CEFRIEL[Organisation]
```

The former states that CEFRIEL, which is an organisation, has got eTECH as unit and the later that Brioschi, which is a person and a head of unit, works for CEFRIEL. All the words are links that retrieve the resource with the corresponding label.

Categorised links
The propotype has got 3 boxes containing categorized links. A first one is the *contains* box, that shows links to resources conceptually "contained" in the retrieved one. We have chosen to interpret "contained" in a relaxed way including both `rdfs:subclassOf` hierarchies and user defined (via `contains`) hierarchies. A second one is the *contained* box, that shows links to resources that "contains" the retrieved one, thus either the superclasses or the resources related to the retrieved one via `contained`. Finally a third one is the *related* box, that shows links to resources that are associated to the retrieved resource via a `related` property.

As we explain instead of asking to use directly these terms, we expect that corporate terminology is mapped to navigation upper terminology. In particular we choose to map properties using `rdfs:subpropertyOf`. This way the reasoner can easily compute sub-property closure and "understand" that two resources are related (e.g. via `contains`) not only when it is explicitly stated, but also when it is entailed.

Access point links
Finally the prototype has got a global navigational bar and a contextual navigational bar configurable through the access model. The global navigation bar is populated with links to resources of type `Home`, while for the contextual navigation we use an approach similar to the one illustrated for categorised links. So our prototype populates the boxes labeled "prev", "next", "up" and "contextual navigation" with links to resources, that are associated to the retrieved resource, respectively via a `prev`, `next`, `up` and `down` property.

Switching between different models
In order to show how different views, of the same corporate memory, can be generated by combining navigation and access models, we develop also a "management service" (available on-line at `http://seip.cefriel.it/seip/manager.html`) that can be used to switch between a set of available corporate memories mounting different navigation and access models.

Related works
The approach that shows more similarities with ours is COHSE [Carr *et al.*, 2001]. Its main concern is in linkage and navigation aspects between web pages, but it doesn't model explicitly *views* using navigation and access models. Another similar approach is SEAL [Maedche *et al.*, 2001] and its recent evolution SEAL-II, but they both uses pre-semantic web technologies.

4 Conclusion

The described approach for semantic EIPs brings many innovation in EIP development. It imposes no restriction but the use of RDF, RDF Schema and OWL in building the corporate ontology. It doesn't require the information carried by the metadata to be coded in any particular way, thus this information is reusable. It enables both resources and metadata management in a distributed and autonomous way as long as resources are network retrievable. Yet, it offers a homogeneous navigation experience over a corporate semantic web through mapping of corporate terminology to the portal terminology.

So, a semantic EIP, built using the proposed approach, will give a unified view of the information present in the corporate semantic web, while the enterprise can keep developing distributed and autonomous systems on an ad-hoc basis and singular enterprise departments can keep their degree of autonomy in managing such systems.

Acknowledgements
We thank our student Lara Marinelli and we report that the implementation of the prototype has been partially founded by Engineering as part of CEFRIEL XV Master IT

References

[Carr *et al.*, 2001] Les Carr, Wendy Hall, Sean Bechhofer, and Carole A. Goble. Conceptual linking: ontology-based open hypermedia. In *World Wide Web*, pages 334–342, 2001.

[Ceri *et al.*, 2000] Stefano Ceri, Piero Fraternali, and Aldo Bongio. Web Modeling Language (WebML): a modeling language for designing Web sites. *Computer Networks (Amsterdam, Netherlands: 1999)*, 33(1–6):137–157, 2000.

[Haarslev and Moller, 2001] Volker Haarslev and Ralf Moller. High performance reasoning with very large knowledge bases: A practical case study. In *IJCAI*, pages 161–168, 2001.

[Maedche *et al.*, 2001] Alexander Maedche, Steffen Staab, Nenad Stojanovic, Rudi Studer, and York Sure. SEAL – A framework for developing SEmantic Web PortALs. *Lecture Notes in Computer Science*, 2097:1–7, 2001.

Lucy and Pete deal with Mom – implementing the Scientific American Scenario

James Hendler and **Bijan Parsia** and **Evren Sirin**
{hendler, evren}@cs.umd.edu,{bparsia}@isr.umd.edu

Maryland Information and Network Dynamics Laboratory
Semantic Web and Agents Project (MINDSWAP)
University of Maryland,
College Park MD 20742, USA

Two years ago this May, *The Semantic Web* [Berners-Lee *et al.*, 2001] article appeared in Scientific American. The authors started the article with a futuristic scenario of what could be done when Semantic Web technologies would come of age. At this point in time, two years after publication, the technologies have reached the point where a prototype of all the pieces can be shown and integrated, as we will show in this demonstration, using currently available, open-source, Semantic Web tools developed at our lab or elsewhere. We will also demonstrate the tools individually and discuss how the demonstration was accomplished.

The first part of the scenario describes the interaction between devices where one device is able to discover the other devices in the environment, find out their capabilities and control their functionality. We designed an architecture where devices describe their functionality through web service descriptions written in the DAML-S language [DAML Services Coalition, 2002], these descriptions are made available for discovery using Universal Plug and Play (UPnP) technology. We extended the DAML-S groundings to include UPnP groundings and to directly invoke Web Service Description Language (WSDL) groundings. Therefore, using a functionality of the device is same as invoking a web service. The scenario requires a small device such as a telephone have the processing power to achieve these goals. This is achieved by assigning a simple computer, actually a PDA, to handle these responsibilities.

Following this in the scenario are a number of agents that operate on semantic web to do the tasks for a user. We represent some of the actions defined in the scenario as web services, e.g. there will be one web service returning available appointment times for the doctor. The markup of these web services with DAML-S language allows us to make discovery, composition and execution by linking the descriptions of services to ontologies written in the Web Ontology Language, OWL on the Semantic Web. We have developed a service composition tool [Sirin *et al.*, 2003] to compose DAML-S descriptions and execute them using the WSDL and UPnP groundings.

Besides the ability to process web services, the user agent also needs to have a planning capability not only to arrange a meeting time between different people's schedules but also to find the correct order of appropriate services to get the information in order to accomplish the goal. We are using the Simple Hierarchical Ordered Planner (SHOP) [Nau *et al.*, 2003] for composing services. SHOP is a domain independent HTN planner that can solve classical AI planning problems. We developed a way [Wu *et al.*, 2003] to map the web service composition task to a planning problem defined for SHOP. By translating DAML-S services to methods and operators in SHOP, we can solve the problem of finding a set of services that will achieve some specified goal.

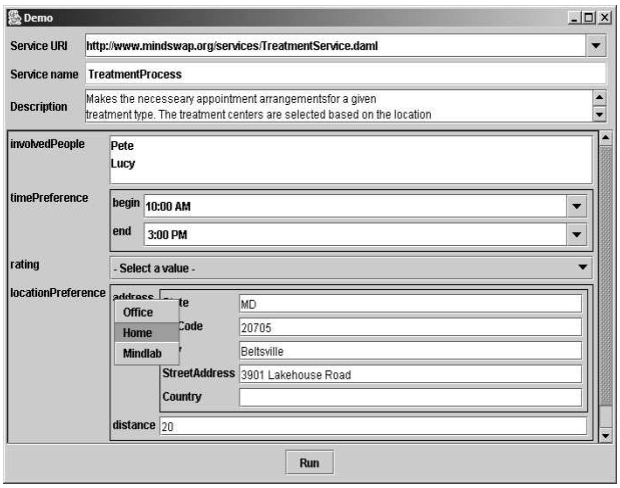


Figure 1: User interface that creates data entry forms to easily fill the paramaters for DAML-S services

Another important aspect of the scenario is that ontologies are distributed at different sources and not always directly compatible with each other. We will show a demo of OntoLink 2 a software which is used to define semantic mappings between concepts that are defined at different ontologies through a simple user interface. We will show how some of these mapping tasks are automated by using some heuristics and how the user can extend these mappings by defining ad-hoc transformations between the concepts. Same tool is also used to generate the semantic service descriptions from existing WSDL descriptions.

The scenario requires the agents of Lucy and Pete share information with each other based on the fact that they have a pre-defined trust relation. To accomplish this task, agents first

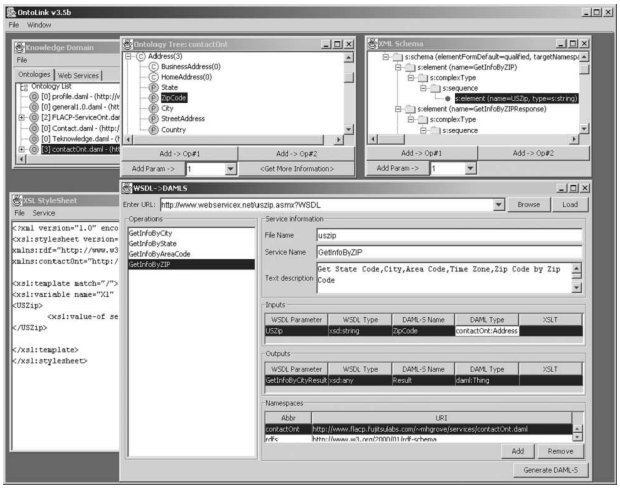


Figure 2: A tool to automate translation from WSDL descriptions to DAML-S and define mappings between ontologies

need to authenticate and then decide how much information can be shared with the other party based on their trust relationship. We demonstrate a simple rule-based authentication (substituting for an eventual public key or other such more robust system). After authentication takes place, one agent must also decide if the other agent is trusted enough to share the requested information. For this purpose, we have developed a distributed trust system [Golbeck *et al.*, 2003] using social network analysis. Everybody assigns a trust value to the people they know and using graph theory trust relationship can be deduced between nodes who did not explicitly state any trust level to each other but can be linked through people they trust.

Another feature described in the scenario is people who are not computer experts such as the clinic's office manager can generate the semantic markups. The demo of RDF/RDFS/OWL-Driven Mindswap Semantic Web Site [Mindswap Semantic Web Site, 2003] will show how users can view, query and modify the semantic data at the web site. The various different technologies used for storing the data (e.g. Redland toolkit), querying the triplestore (e.g. several different scripting languages), generating user viewable web pages (e.g. XSLT) and interfaces that lets the user interactively edit the content will be shown.

References

[Berners-Lee *et al.*, 2001] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American*, May 2001.

[DAML Services Coalition, 2002] DAML Services Coalition. DAML-S: Web Service Description for the Semantic Web. In *The First International Semantic Web Conference (ISWC)*, June 2002.

[Golbeck *et al.*, 2003] Jennifer Golbeck, Bijan Parsia, and James Hendler. Trust networks on the semantic web. In *Proceedings of Cooperative Intelligent Agents 2003*, Helsinki, Finland, August 2003.

[Mindswap Semantic Web Site, 2003] Mindswap Semantic Web Site. http://owl.mindswap.org, 2003.

[Nau *et al.*, 2003] Dana Nau, Tsz-Chiu Au, Okhtay Ilghami, Ugur Kuter, William Murdock, Dan Wu, and Fusun Yaman. SHOP2: An HTN planning system. *Journal of Artificial Intelligence Research*, 2003.

[Sirin *et al.*, 2003] Evren Sirin, James Hendler, and Bijan Parsia. Semi-automatic composition of web services using semantic descriptions. In *Web Services: Modeling, Architecture and Infrastructure workshop in ICEIS*, Angers, France, April 2003.

[Wu *et al.*, 2003] Dan Wu, Bijan Parsia, Evren Sirin, James Hendler, and Dana Nau. Automating DAML-S web services composition using SHOP2. In *Proceedings of 2nd International Semantic Web Conference (ISWC2003)*, Sanibel Island, Florida, October 2003.

Querying Real World Services through the Semantic Web

Kaoru Hiramatsu Jun-ichi Akahani Tetsuji Satoh
NTT Communication Science Laboratories
Nippon Telegraph and Telephone Corporation
2-4, Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0237 Japan
{hiramatu,akahani}@cslab.kecl.ntt.co.jp satoh.tetsuji@lab.ntt.co.jp

Abstract

We propose a framework for querying information of real world services, such as store locations/hours and routes of public transportation, through the Semantic Web. In this framework, a natural language query of real world services is accepted by a user interface module and translated into extended-SQL. The translated query is processed by service coordinator agents that find appropriate Web services according to each service description in DAML-S. The results are visualized in various styles such as digital maps and structural tree views. While processing, the query is revised to obtain an adequate number of search results based on spatial-temporal ontologies, and next available options are provided for jumping to advanced and associated topics. In this demonstration, we present a prototype system based on this framework and show how it works through searching for real world services in Kyoto, Japan.

1 Introduction

Information of real world services, such as store locations/hours and routes of public transportation, are popular and frequently used contents on the Internet. For example, Web pages of stores in a city provide their locations and hours, and route planners are also available through many sites of online map services and transport facilities. However, they are only associated with each other by hyperlinks and the search engines provide mere pointers to the result based on indices created from scraped keywords of Web pages. This simple framework restricts users and information providers to conducting related information and cascading Web services flexibly.

To remove such restrictions, the Semantic Web [Berners-Lee *et al.*, 2001] is expected to play an important role as an extension of the current Web. After the Semantic Web starts functioning, annotation data of Web contents based on standard formats (e.g., RDF), vocabularies, and ontologies will be published online. Its processing framework will not only enhance the search engines but also enable us to access various neighbor information based on semantic relations and find personal optimal services according to each service descriptions in DAML-S [Coaliation, 2002].

Moreover, these meta data enable adaptable interaction between users and systems for searching the real world services. With the Semantic Web, the system supports users to find preferable information by query modification based on semantic relations and enhance the initial query into next available options for jumping to advanced and associated topics according to annotation data of the Web pages based on spatial-temporal ontologies. These meta data are also applicable to handling a natural language query with natural language processing and visualizing the search result in suitable styles for the data types of the result.

In this demonstration, we present a prototype system based on the above framework and show how it works through searching for real world services in Kyoto, Japan.

2 System Overview

The prototype system consists of user agents, service coordinator agents, and Web services. We implemented these modules using Java, Jena (a Java API for manipulating RDF models), Jun for Java (a 3D graphics class library), and PostgresSQL (an open source Object-Relational DBMS).

For the purpose of this demonstration, we prepared a test data set that is extended from the original data created for the Digital City Kyoto prototype [Ishida *et al.*, 1999]. We collected Web pages in Kyoto, Japan from the Internet and described their meta data based on spatial-temporal ontologies. These data are accessed via a Semantic Web search service that we prepared. We also collected Web services of real world services. These Web services are accessed via the service coordinator agents.

The prototype system works as follows.

The user agent accepts natural language queries and translates them into extended-SQL [Hiramatsu and Ishida, 2001]. The translated query includes the conditions of information attributes and relationships among information.

According to the translated query, the service coordinator agents find appropriate information of the real world services. In this prototype system, each agent advertises its service descriptions in DAML-S. These descriptions enable the service coordinator agents to find and coordinate appropriate Web services.

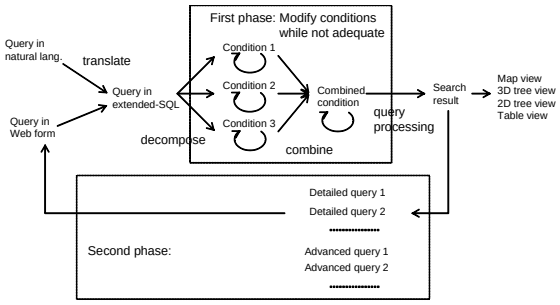


Figure 1: Two-phase query modification

While query processing, the query is refined into an appropriate one for getting an adequate search result according to the number of intermediate search results. The query is also editable through a Web query form after query processing. In addition, the search results are visualized in various styles, such as digital maps and tables, according to data types, so that users are able to have a good understanding of the relational structures among the search results.

3 Two-phase Query Modification

We employ two-phase query modification [Hiramatsu *et al.*, 2003] into our prototype system for conducting users to interactive query evolution. This query modification is divided into two phases:

1. Revising ambiguous conditions into appropriate ones for getting an adequate number of search results, and
2. Providing next available options to enable users to jump to advanced and associated topics.

The first phase is processed automatically during query processing to avoid outputting a zero search result or a huge result list. The second phase is invoked after query processing and requires the user's selection based on a visualized result. Both phases are processed tightly coupled with query processing in accordance with semantic relations derived from meta data, thesauri, and gazetteers that are based on spatio-temporal ontologies.

4 Coordinating Real-World Services

There are various services available on networks in the real world. It is necessary to find adequate services for queries. We therefore introduce service coordinator agents into our framework. In our framework, one service coordinator agent performs one or both of the following roles.

1. Service provider agents that provide services. Each service provider agent advertises its service description in DAML-S.
2. Mediator agents that forward queries to adequate service provider agents based on the service descriptions of the service provider agents.

Moreover, the service provider agents are categorized into the following two types.

1. Service wrapper agent that wraps Web services.
2. Service integrator agent that integrates services provided by other service provider agents. Each service integrator agent advertises a composite service description.

In the prototype system, we implemented two types of Web services: a Semantic Web search service and a route finding service. These Web services are wrapped by the service wrapper agents. We also implemented a service integrator agent that integrates these two services. For example, consider a query, "find a route to Kyoto station and a bank on the way to Kyoto station." The user agent translates the query and asks a mediator agent. The mediator agent forwards the translated query to the service integrator agent based on the service descriptions. The service integrator agent first asks a service wrapper agent that provides a route finding service about the route. Then, the service integrator agent asks a service wrapper agent that provides a Semantic Web search service about a bank along the route.

5 Conclusion

In this demonstration, we showed how the prototype system works through searching for real world services in Kyoto, Japan. We assume enlargement of the Semantic Web will lead to a close relation between the Internet and the real world services. To accelerate such evolution, we are planning to refine the framework and the prototype system along with meta data and ontologies.

Acknowledgment

Thanks are due to content holders for the permission to use their Web pages concerned with Kyoto, and to Yuji Nagato and Yoshikazu Furukawa of NTT Comware Corporation for their great contributions to the demonstration system.

References

[Berners-Lee *et al.*, 2001] Tim Berners-Lee, James Hendler, and Ora Lassila. The Semantic Web. *Scientific America*, May 2001.

[Coaliation, 2002] The DAML Service Coaliation. DAML-S: Web Service Description for the Semantic Web. In *The First International Semantic Web Conference (ISWC)*, 2002.

[Hiramatsu and Ishida, 2001] Kaoru Hiramatsu and Toru Ishida. An Augmented Web Space for Digital Cities. In *The 2001 Symposium on Application and the Internet (SAINT2001)*, pages 105–112, 2001.

[Hiramatsu *et al.*, 2003] Kaoru Hiramatsu, Jun-ichi Akahani, and Tetsuji Satoh. Two-phase Query Modification using Semantic Relations based on Ontologies. In *Proceedings of IJCAI-03 Workshop on Information Integration on the Web (IIWeb-03)*, pages 155–158, 2003.

[Ishida *et al.*, 1999] Toru Ishida, Jun-ichi Akahani, Kaoru Hiramatsu, Katherine Isbister, Stefan Lisowski, Hideyuki Nakanishi, Masayuki Okamoto, Yasuhiko Miyazaki, and Ken Tsutsuguchi. Digital City Kyoto: Towards A Social Information Infrastructure. In *Cooperative Information Agents III*, volume 1652 of *Lecture Notes in Computer Science*, pages 34–46, 1999.

Application Scenario for Semantic Annotation of Image Collections

Laura Hollink[1] and **Guus Schreiber**[1] and **Jan Wielemaker**[2] and **Bob Wielinga**[2]

[1]Free University Amsterdam, Computer Science, e-mail {laurah,schreiber}@cs.vu.nl
[2]University of Amsterdam, Social Science Informatics, e-mail {jan,wielinga}@swi.psy.uva.nl

1 Overview

In this demo we show how ontologies can be used to support annotation and search in image collections. Figure 1 shows the general architecture we used within this study. For this study we used four ontologies (AAT, WordNet, ULAN, Iconclass) which were represented in RDF Schema. The resulting RDF Schema files are read into the tool with help of the SWI-Prolog RDF parser[1]. The tool subsequently generates a user interface for annotation and search based on the RDF Schema specification. The tool supports loading images and image collections, creating annotations, storing annotations in a RDF file, and two types of image search facilities.

For this study we used four thesauri, which are relevant for the art-image domain:

1. The Art and Architecture Thesaurus (AAT) is a large thesaurus containing some 125,000 terms relevant for the art domain. The terms are organized in a single hierarchy.
2. WordNet is a general lexical database in which nouns, verbs, adjectives and adverbs are organized into synonym sets, each representing one underlying lexical concept. WordNet concepts (i.e. "synsets") are typically used to describe the content of the image. In this study we used WordNet version 1.5, limited to hyponym relations.
3. Iconclass is an iconographic classification system, providing a hierarchally organized set of concepts for describing the content of visual resources. We used a subset of Iconclass.
4. The Union list of Artist Names (ULAN)contains information about around 220,000 artists. A subset of 30,000 artists, representing painters, is incorporated in the tool.

For annotation and search purposes the tool provides the user with a description template derived from the VRA 3.0 Core Categories. The VRA template is defined as a specialization of the Dublin Core set of metadata elements, tailored to the needs of art images. The VRA Core Categories follow the "dumb-down" principle, i.e., a tool can interpret the VRA data elements as Dublin Core data elements. The subject of the image is described with a collection of statements of the form "*agent action object recipient*". Each statement should at least have an agent (e.g. a portrait) or an object (e.g. a still life). The terms used in the sentences are selected from terms in the various thesauri.

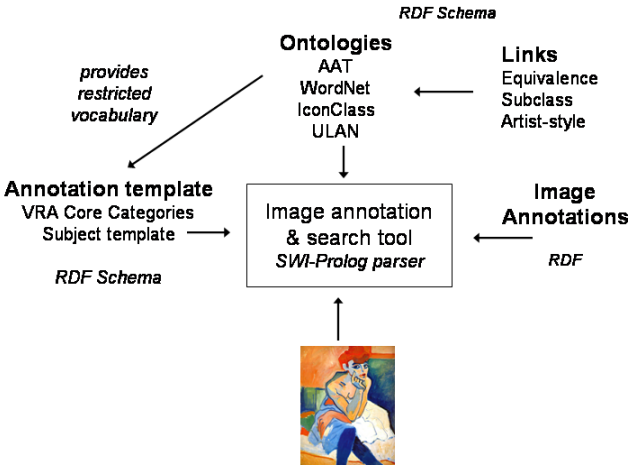


Figure 1: Tool architecture.

Where possible, a slot in the annotation template is bound to one or more relevant subtrees of the ontologies. For example, the VRA slot style/period is bound to two subtrees in AAT containing the appropriate style and period concepts.

The four ontologies contain many terms that are in some way related. For example, WordNet contains the concept `wife`, which is in fact equal to the AAT concept `wives`. We added three types of ontology links: (1) equivalence relations, (2) subclass relations, and (3) domain-specific relations: e.g., artist to style.

2 Demo Excerpts[2]

2.1 Annotating art-historic features

Figure 2 shows a screenshot of the annotation interface. In this scenario the user is annotating an image of a painting

[1]For more information see: J. Wielemaker *et al.* (2003) Prolog-based infrastructure for RDF: performance and scalability. *Proceedings ISWC'03*

[2]Other functionality includes transforming existing annotations and annotating image content.

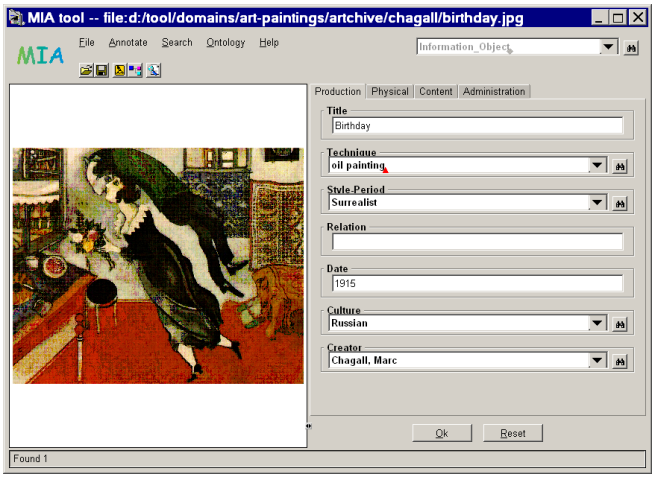


Figure 2: Screenshot of the annotation interface.

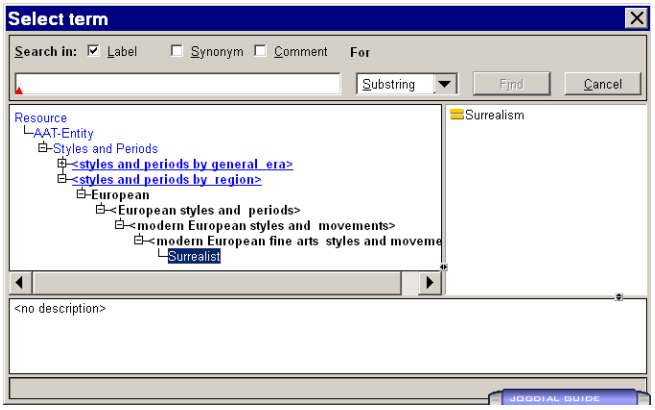


Figure 3: Browser window for values of style/period.

by Chagall. The figure shows the tab for production-related VRA data elements. The four elements with a "binoculars" icon are linked to subtrees in the ontologies, i.e., AAT and ULAN. For example, if we would click on the "binoculars" for style/period the window shown in Figure 3 would pop up, showing the place in the hierarchy of the concept `Surrealist`. We see that it is a concept from AAT. The top-level concepts of the AAT subtrees from which we can select a value for style/period are shown with an underlined bold font (i.e., <`styles and periods by general era`> and <`styles and periods by region`>).

2.2 Searching for an image

The tool provides two types of semantic search. With the first search option the user can search for concepts at a random place in the image annotation. Figure 4 shows an example of this. Suppose the user wants to search for images associated with the concept `Aphrodite`. Because the ontologies contain an equivalence relation between `Venus` (as a Roman deity, not the planet nor the tennis player) and `Aphrodite`, the search tool is able to retrieve images for which there is no syntactic match. For example, if we would look at the annotation of the first hit in the right-hand part of Figure 4, we

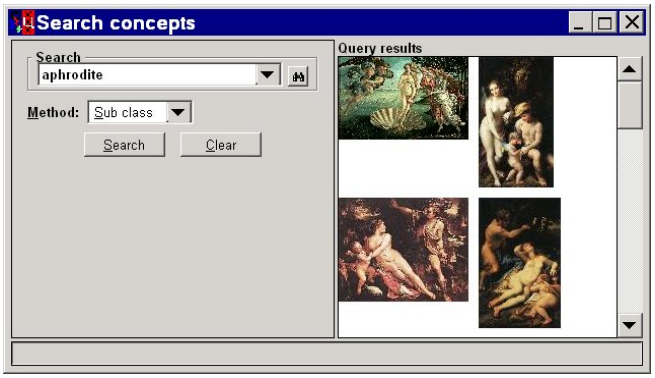


Figure 4: Example of concept search.

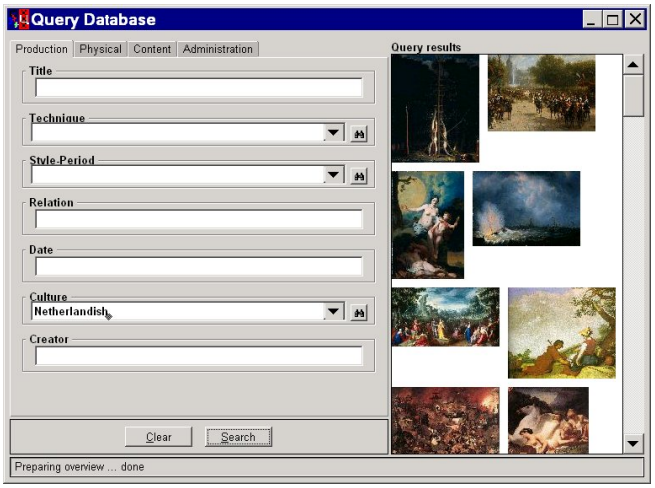


Figure 5: Search using the annotation template.

would find "Venus" in the title ("Birth of Venus" by Botticelli) and in the subject-matter description (`Venus (a Roman deity) standing seashell`). The word "Venus" in the title can only be used for syntactic marches (we do not have an ontology for titles), but the concept in the subject description can be used for semantic matches, thus satisfying the "Aphrodite" query.

General concept search retrieves images which match the query in some part of the annotation. The second search option allows the user to exploit the annotation template for search proposes. An example of this is shown in Figure 5. Here, the user is searching for images in which the slot culture matches `Netherlandish`. This query retrieves all images with a semantic match for this slot. This includes images of `Dutch` and `Flemish` paintings, as these are subconcepts of `Netherlandish`.

Acknowledgments This work was supported by the IOP Project "Interactive discolore of Multimedia Information and Knowledg" and the ICES-KIS project "Multimedia Information Analysis", both funded by the Dutch Ministry of Economic Affairs. We gratefully acknowledge the contributions of Marcel Worring, Giang Nguyen and Maurice de Mare.

Hozo: Treatment of "Role", "Relationship" and Dependency Management *

Kouji Kozaki*, Eiichi Sunagawa*, Yoshinobu Kitamura*, and Riichiro Mizoguchi*
*The Institute of Scientific and Industrial Research, Osaka University
8-1 Mihogaoka, Ibaraki, Osaka, 567 -0047 Japan
{kozaki,sunagawa,kita,miz}@ei.sanken.osaka-u.ac.jp

Abstract

We have developed an environment for building/using ontologies, named Hozo. Since Hozo is based on an ontological theory of a role-concept, it can distinguish concepts dependent on particular contexts from so-called basic concepts and contribute to building reusable ontologies. We present an outline of the features of Hozo and demonstrate its functionality.

1 Introduction

Building an ontology requires a clear understanding of what can be concepts with what relations to others. Although several tools for building ontologies have been developed to date, few of them were based on enough consideration of an ontological theory. We argue that a fundamental consideration of these ontological theories is needed to develop an environment for developing an ontology [Sowa, 1995; Guarino, 1998]. We have developed an environment for building/using ontologies, named Hozo, based on both of a fundamental consideration of an ontological theory and a methodology of building an ontology. The features of Hozo are: 1) it can distinguish concepts dependent on particular contexts from so-called basic concept, 2) it can manage the correspondence between a wholeness concept and a relation concept, 3) it supports distributed ontology development based on dependency management between component ontologies. We present an outline of the features of Hozo and demonstrate its functionality.

2 Hozo

2.1 The architecture of Hozo

We have developed an integrated ontology engineering environment, named "Hozo ", for building/using task ontology and domain ontology based on fundamental ontological theories[Kozaki et al., 2000; 2002]. "Hozo" is composed of "Ontology Editor", "Onto-Studio" and "Ontology Server" (Figure.1). Ontology Editor provides users with a graphical interface, through which they can browse and modify ontologies by simple mouse operations (Figure.2). Onto-Studio is based on a method of building ontologies, named AFM (Activity-First Method) [Mizoguchi et al., 1995]. The building process of ontologies using Onto-Studio consists of 12 steps and it helps users design an ontology from technical documents. Ontology Server manages ontologies and models which are built in Hozo. The ontology and the resulting model are available in different formats (Lisp, Text, XML/DTD, DAML+OIL) that make it portable and reusable.

2.2 The features of Hozo

Hozo has been designed based on a fundamental consideration of ontological theories, and it has following remarkable features:

1. Clear discrimination among a role-concept (e.g. teacher role), a role-holder (e.g. teacher) and a basic concept (e.g. man) is done to treat "Role" properly.
2. Management of the correspondence between a wholeness concept (e.g. brothers) and a relation concept (e.g. brotherhood).
3. Distributed ontology development based on dependency management between component ontologies.

What is a role? : Basic concept, role concept and role holder

When an ontology is seriously used to model the real world by generating instances and then connecting them, users have to be careful not to confuse the Role such as teacher, mother, front wheel, fuel, etc. with other basic concepts such as human, water, oil, etc. The former is a role played by the latter. To deal with the concept of role appropriately, we identified three categories for a concept. That is, a basic concept, a role-concept, and a role holder.

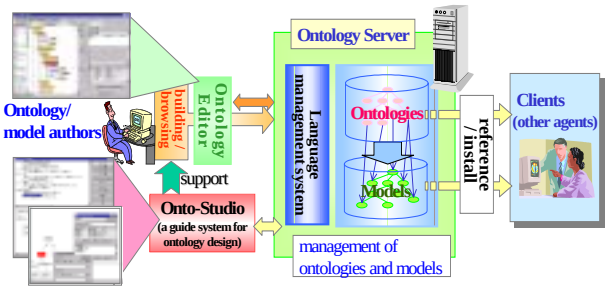


Figure.1 The architecture of Hozo

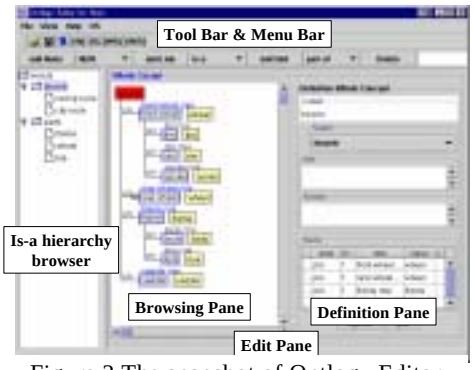


Figure.2 The snapshot of Ontlogy Editor

A role-concept represents a role which a thing plays in a specific context and it is defined with other concepts. On the other hand, a basic concept does not need other concepts for being defined. An entity of the basic concept that plays a role such as teacher role or wife role is called a role holder. A basic concept is used as the class constraint. Then an instance that satisfies the class constraint plays the role and becomes a role holder. For example, when a man plays a role as a teacher ("a teacher role") in a school which is defined as a role-concept, he is called "a teacher" which is role holders. Hozo supports to define such a role concept as well as a basic concept.

Wholeness concept and relation concept

There are two ways of conceptualizing a thing. Consider a "brothers" and a "brotherhood". "The Smith brothers" is a conceptualization as concept, on the other hand "brotherhood between Bob and Tom" is conceptualized as a relation. On the basis of the observations that most of the things are composed of parts and that those parts are connected by a specific relation to form the whole, we introduced "wholeness concept" and "relation concept". The former is a conceptualization of the whole and the latter is that of the relation. In the above example, the "brothers" is a wholeness concept and the "brotherhood" is a relation concept. Because a wholeness concept and a relation concept are different conceptualizations derived from the same thing, they correspond to each other. Theoretically, every thing that is a composite of parts can be conceptualized in both perspectives as a wholeness concept and a relation concept. Hozo can manage the correspondence between these two concepts.

Distributed ontology development based on dependency management

Hozo supports development of an ontology in a distributed manner. By a distributed manner, we mean an ontology is divided into several component ontologies, which are developed by different developers in a distributed environment. The target ontology is obtained by compiling the component ontologies. To support such a way of ontology development, Ontology Editor allows users to divide an ontology into several component ontologies and manages the dependency between them to enable distributed development of an ontology. We introduced two dependencies: super-sub relation (is-a relation) and referred-to relation (class constraint). The system observes every change in each component ontologies and notifies it to the appropriate users who are editing the ontology which might be influenced by the change. The notification is done based on the 16 patterns of influence propagation analyzed beforehand. The notified users can select the countermeasure among the three alternatives: (1)to adapt his/her ontology to the change, (2)not to do adapt to the change but stay compliant with the last version of the changed ontology and (3)neglect the change by copying the last version into his/her ontology[Sunagawa et al., 2003].

3 Conclusion and Future work

We outlined our ontology development system, Hozo. The system has been implemented in Java and its ontology editor has been used for 6 years not only by our lab members but also by some researchers outside [Mizoguchi et al., 2000; Kitamura et al., 2003]. We have identified some room to improve Hozo through its extensive use. The following is the summary of the extension:

- Ontological organization of various role-concepts.
- Augmentation of the axiom definition and the language.
- Gradable support functions according to a user's level of skill.

References

[Guarino, 1998] N. Guarino, Some Ontological Principles for Designing Upper Level Lexical Resources. Proc. of the First International Conference on Lexical Resources and Evaluation, Granada, Spain, 28-30, May 1998.

[Kitamura et al., 2003] Y. Kitamura and R. Mizoguchi, Ontology-based description of functional design knowledge and its use in a functional way server, Expert Systems with Applications, Vol.24, pp.153-166, 2003.

[Kozaki et al., 2000] K. Kozaki, et al., Development of an Environment for Building Ontologies which is based on a Fundamental Consideration of "Relationship" and "Role": Proc. of PKAW2000, pp.205-221, Sydney, Australia, December, 2000

[Kozaki et al., 2002] K. Kozaki, et al., Hozo: An Environment for Building/Using Ontologies Based on a Fundamental Consideration of "Role" and "Relationship", Proc. of EKAW2002, pp.213-218, Sigüenza, Spain, October 1-4, 2002

[Mizoguchi et al., 1995] R. Mizoguchi, M. Ikeda, K. Seta, et al., Ontology for Modeling the World from Problem Solving Perspectives, Proc. of IJCAI-95, pp. 1-12, 1995.

[Mizoguchi et al., 2000] R. Mizoguchi, et al., Construction and Deployment of a Plant, Proc. of EKAW200, Juan-les-Pins, French Riviera, October, 2000.

[Sowa, 1995] John F. Sowa, Top-level ontological categories, International Journal of Human and Computer Studies, 43, pp.669-685, 1995

[Sunagawa et al., 2003] E. Sunagawa, K. Kozaki, et al., An Environment for Distributed Ontology Development Based on Dependency Management, Proc. of ISWC2003, Florida, USA, October 20-23, 2003

Task Computing

Yannis Labrou and Ryusuke Masuoka
Fujitsu Laboratories of America, Inc.
8400 Baltimore Avenue, Suite 302
College Park, MD 20740-2496, U.S.A
{yannis,rmasuoka}@fla.fujitsu.com

Description

This demo complements the paper, "Task Computing – the Semantic Web meets Pervasive Computing," which has been accepted for ISWC2003 (Industrial Track #202).

Task computing is a new paradigm for how users interact with devices and services that emphasizes the tasks that users want to accomplish while using computing devices rather than how to accomplish them. Task computing fills the gap between what users want to do and the devices and/or services that might be available in their environments. Task computing presents substantial advantages over traditional approaches, such as the current personal computing paradigm, namely, it is more adequate for non-expert computer users, it is a time-saver for all types of users and is particularly suited for the emerging pervasive computing type of computing environments.

We call "Task Computing Environments (TCE)," a framework that support task computing, by providing support for its workflows, semantic service descriptions, and service management for end-users.

Our Task Computing Environment (TCE) consists of Task Computing Clients (TCC), which we call STEER (Semantic Task Execution EditoR), multiple Semantically Described Services (SDS's), Semantic Service Discovery Mechanisms (SSDM's), and Service Controls.

We base our technology on standards as much as possible. For example, we use a web client for STEER's user interface, UPnP [1] for SSDM, DAML-S [2] for semantic service descriptions, UPnP and Web services for service invocations. By combining these existing technologies in a framework that enables user-driven discovery, composition and execution of complex tasks, in real-time (as opposed to design time) task computing provides a totally different level of interoperability between devices and services, along with a novel user experience.

In the demo, for example, the user can display her slides from her own computer or the remote web service result on any display in the environment or use the environment to share information with other users (even after the first user left the environment!). Such a universal and flexible task computing framework proves, we believe, to be very useful and powerful in environments like hospitals, offices, and homes where the end-user can integrate and manipulate seamlessly functionalities on her own computer, devices around her, and remote web services, enabling her to easily define, execute and monitor complex tasks, in ways that can only be accomplished today by painstaking, design-time integration.

1. Universal Plug and Play, http://www.upnp.org/
2. DAML Services, http://www.daml.org/services/

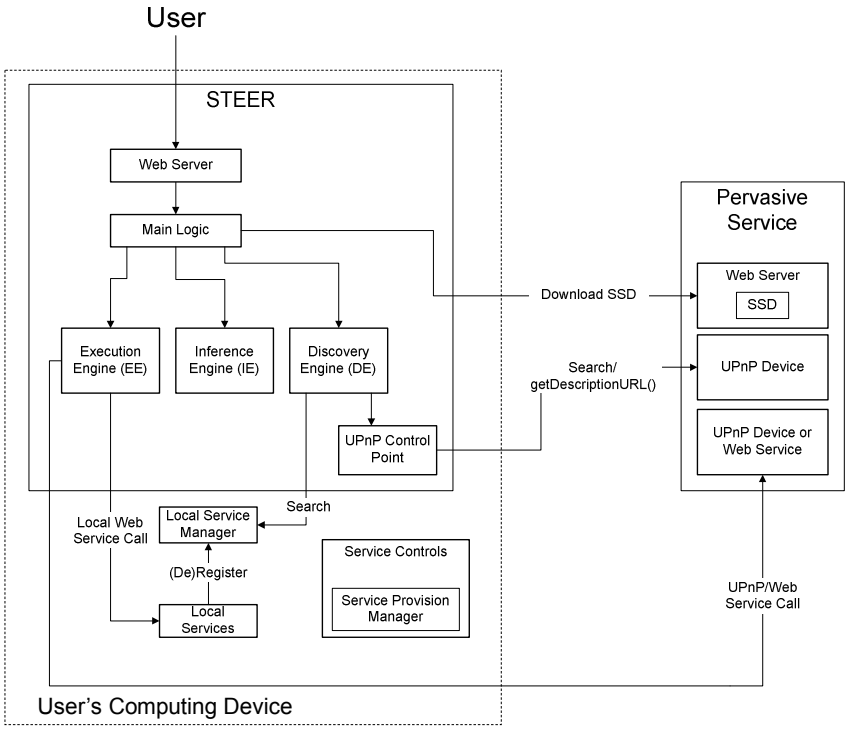


Fig. 1. Architecture of Task Computing Environment:

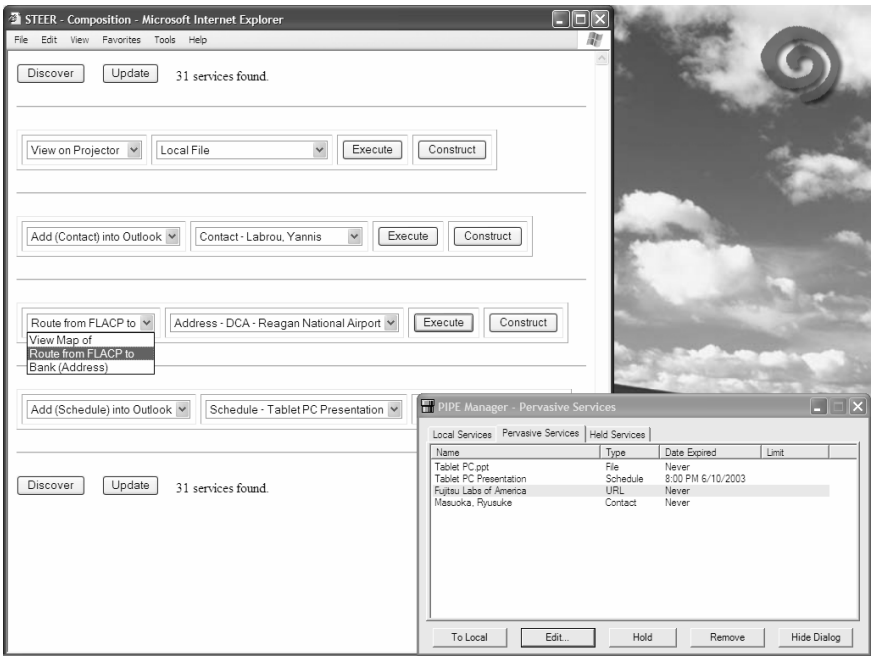


Fig. 2. Screenshot of Task Computing Environment (TCE) Client Desktop

Demonstrator: Ontologies and Inference in Delivering Policy-Driven Automotive Supply Chain Automation

Gary Ng, Henrik Pettersen, Matthew Quinlan, Azad Uddin

Network Inference Limited, 25 Chapel Street, London NW1 5DH

1. Introduction

This abstract describes a demonstrator using Network Inference's Construct and Cerebra Server and the W3C's OWL language [McGuinness et al., 2003] to integrate multiple databases which use different schemas and vocabularies in different corporate domains, and use inference to provide adaptive policy-driven behavior to a supply chain application in the automotive industry.

2. Database Integration

The demonstrator uses a Java client to load and query ontologies using Cerebra Server's standard API over SOAP. Cerebra Server manages database access through its data interface (see Figure 1).

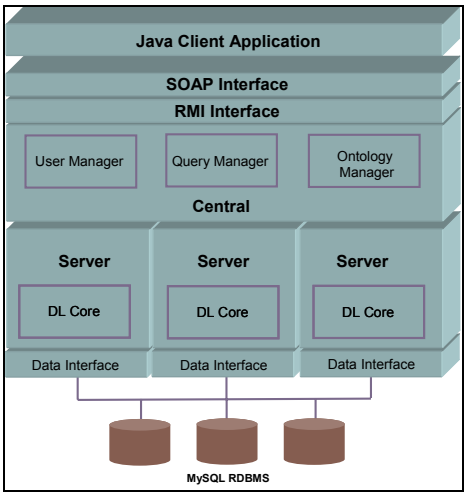


Figure 1: Demonstrator Architecture

The demonstrator starts by loading an ontology whose concepts and properties have been mapped into tables and columns in a single database schema using Construct, a graphical ontology modeling tool in MS Visio (Figure 2). The database schema defines components, their manufacturers and attributes for car models defined within an ERP system. The database is queried via the Cerebra Server query API.

The demonstrator shows the use of ontologies and inferencing to resolve data schema inconsistencies at run-time without recoding at the application level, database changes or other conversion procedures.

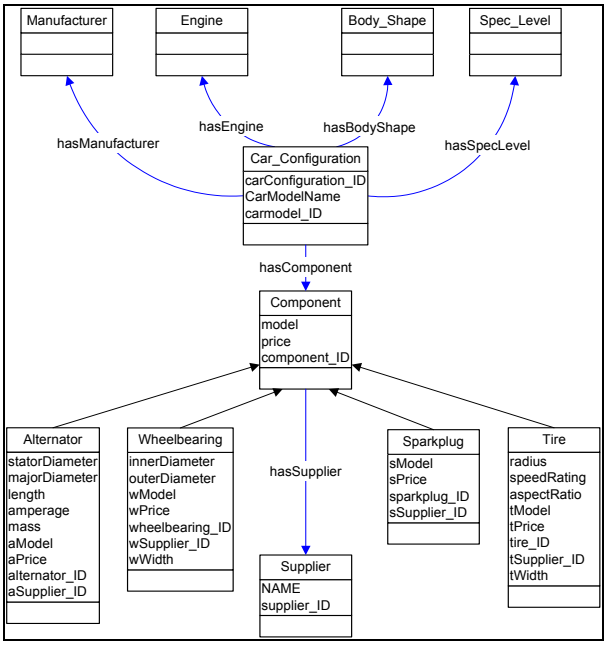


Figure 2: Database Oriented Ontology

The demonstrator loads new ontologies which describe additional databases with different schemas. Further ontologies define a small number of logical statements linking objects in any two of the database ontologies (Figure 3). Cerebra Server dynamically loads, classifies and checks consistency of the 'federated' set of ontologies. At query time, the client application issues a single unchanged query to Cerebra Server which infers the databases, tables and columns required for data retrieval and issues multiple SQL commands.

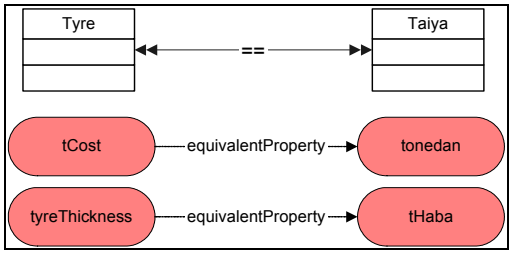


Figure 3: Logical statements linking database ontologies

The addition of a new database requires its association with only **one** of the existing ontologies. The approach proves to be extremely scalable and flexible for enterprise information integration.

3. Policy-driven Supply Chain Management

Section 2 focused on a basic data-oriented ontology to integrate disparate data for querying.

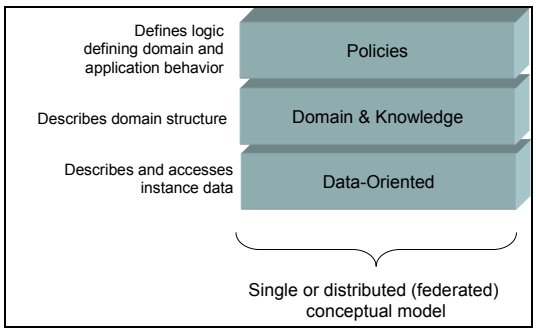


Figure 4: Multiple 'layers' within an ontology architecture

The demonstrator also uses an abstract domain structure (a supply chain ontology) to describe the relations between suppliers and customers, regions, routes, components and products. It is linked to the data-oriented ontology.

The demonstrator introduces an additional ontological definition of supply chain interruptions – localized events which potentially disrupt the supply chain - and associated generic 'policies' (Figure 5).

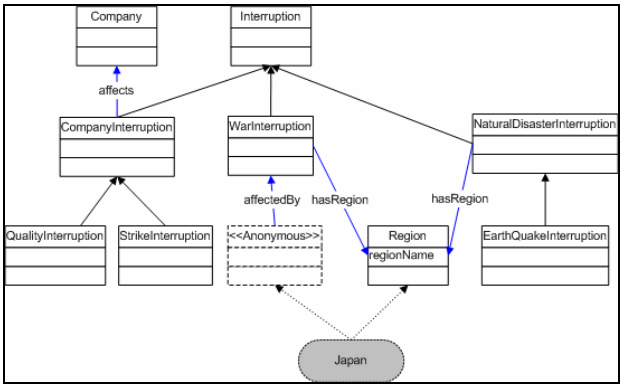


Figure 5: Excerpt of Supply Chain Ontology

The demonstrator allows a simple interruption (defined by type and location) to be dynamically added via the UI, eg a Natural Disaster in Japan. Cerebra Server infers affected car models through their components, suppliers, facilities and delivery routes. The application behavior 'adapts' to the changed state of the supply chain without the need to recode or provide knowledge of the event and its impacts explicitly to users or applications.

The demonstrator uses inference to identify equivalent components or suppliers which are unaffected by the interruption and are valid alternatives to minimize the supply chain impact.

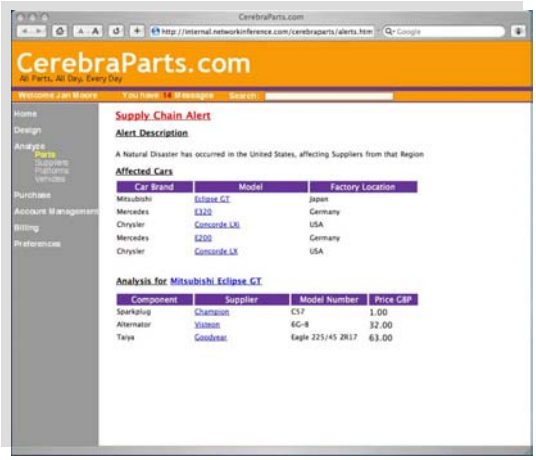


Figure 6: Inferred impacts of supply chain interruption

4. Using Cerebra Server and Construct

Cerebra Server is an enterprise platform, deploying a Description Logic-based inference engine which supports the W3C's OWL-DL. Cerebra Server is deployed as a web service for ease of integration. Its XQuery API provides a flexible, expressive and easy-to-use querying syntax.

Construct enables users to create and edit ontologies, and extend simple structures to describe complex logical expressions according to the OWL specification using graphical symbols and reasoning.

Construct is used with Cerebra Server to minimize complexity and the number of direct relationships needed to represent the business and data models. Cerebra Server is used to resolve data schema inconsistencies at run-time through inference using database mappings defined using Construct. Cerebra Server ensures logical consistency across multiple ontologies.

5. Summary

Cerebra Server and Construct were used to integrate inconsistent databases and provide adaptive behavior to systems through inference using logical 'policies'.

Cerebra Server classifies supply chain interruptions and infers affected production line models. The demonstrator application adapts dynamically to the event without recoding, limiting the event description to defining its direct attributes within the ontology.

References

[McGuinness et al., 2003] McGuinness, D. L., van Harmelen, F., OWL Web Ontology Language Overview, *W3C*, August 2003

SEAN: A System for Semantic Annotation of Web Documents

Amarjeet Singh **Saikat Mukherjee** **Guizhen Yang**
I.V. Ramakrishnan **Zarana Shah** Department of Computer Science and Engineering
Department of Computer Science University at Buffalo
University at Stony Brook Buffalo, NY 14260
Stony Brook, NY 11794

Semantic Web documents use metadata to express the meaning of the content encapsulated within them. Although RDF/XML has been widely recognized as the standard vehicle for describing metadata, an enormous amount of semantic data is still being encoded in HTML documents that are designed primarily for human consumption. Tools such as those pioneered by SHOE [Heflin *et al.*, 2003] and OntoBroker [Fensel *et al.*, 1998] facilitate manual annotation of HTML documents with semantic markups.

Figure 1: New York Times front page

In this demo we will present SEAN, a system for *automatically annotating* HTML documents. It is based on the idea that well-organized HTML documents, especially those that are machine generated from templates, contain rich data denoting semantic concepts (e.g. "News Taxonomy" and "Major Headline News") and concept instances. These kinds of documents are increasingly common nowadays since most Web sites (e.g., news, portals, product portals, etc.) are typically maintained using content management software that creates HTML documents by populating templates from backend databases. For example observe in Fig 1 that it has a news taxonomy (on the left in the figure), which does not change, and a template for major headline news items. Each of these items begins with a hyperlink labeled with the news headline (e.g. "White House..."), followed by the news source (e.g. "By REUTERS..."), followed by a timestamp and a text summary of the article (e.g. "The White House today...") and (optionally) a couple of pointers to related news. These concepts and concept instances can be organized into a semantic partition tree (such as the one shown in Fig 2, which represents the "semantics" of the HTML document.

In a semantic partition tree each partition (subtree) consists of items related to a semantic concept. For example, in Fig 2 all the major headline news items are grouped under the subtree labeled "Major Headline News".

There are two main tasks underlying the creation of a semantic partition tree from a HTML document: (i) identify segments of the document that correspond to semantic concepts; and (ii) assign labels to these segments. Informally, we say that several items are semantically related if they all belong to the same concept.

SEAN automatically transforms well-structured HTML documents into their semantic partition trees by exploiting two key observations. The first observation is that *semantically related items exhibit consistency in presentation style*. For example, observe the presentation styles of the items in the news taxonomy on the left in Figure 1. The main taxonomic items "NEWS", "OPINION", "FEATURES", etc., are all presented in bold font. All the subtaxonomic items (e.g. "International", "National", "Washington", etc.) under the main taxonomic item (e.g. "NEWS") are hyperlinks. A similar observation can also be made on all the major headline news items in the figure. The second observation is that *semantically related items exhibit spatial locality*. For example, when rendered in a browser, all the taxonomic items are placed in close vicinity occupying the left portion of the page. Specifically, in the DOM tree corresponding to the HTML document in Fig 1 all the items in the news taxonomy will be grouped together under one single subtree.

The first observation leads to the idea of associating a type with every leaf node in the DOM tree. The type of a leaf node consists of the root-to-leaf path of this node in the DOM tree and captures the notion of consistency in presentation style. The second observation gives rise to the idea of propagating types bottom-up in the DOM tree and discovering structural

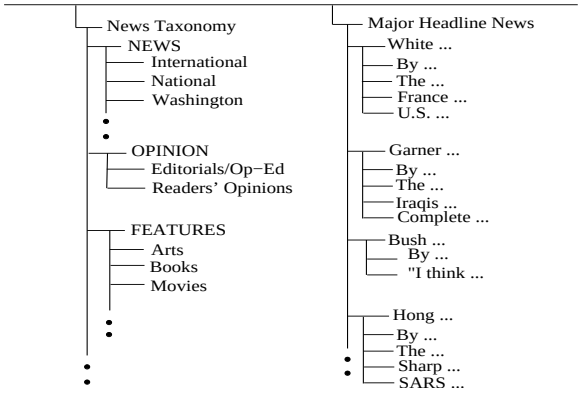


Figure 2: Partition tree of the New York Times front page

recurrence patterns for semantically related items at the root of a subtree. Based on the idea of types and type propagation, SEAN does structural analysis of the HTML document for automatically partitioning it into semantic structures. In the process it also discovers semantic labels and associates them with partitions when they are present in the document (e.g. "NATIONAL", "INTERNATIONAL", etc. appearing in the third column in Fig 1.

SEAN augments structural analysis with semantic analysis to factor in structural variations in concept instances (e.g., the absence of the pointers to related news in the third major headline news item in Fig 1 in contrast to others). Semantic analysis makes lexical associations via WordNet to more accurately put the pieces of a concept instance together. To assign informative labels that are not present in a HTML document (e.g. "Major Headline News" in Fig 1) to partitions semantic analysis makes concept associations by classifying the content of a partition using an ontology encoding domain knowledge.

Thus SEAN uniquely combines structural and semantic analysis to automatically discover and label concept instances in content-rich template-based HTML documents w.r.t. a domain ontology. Details appear in [Mukherjee *et al.*, 2003]. The demo will illustrate how SEAN is used to assign semantic labels to HTML documents. For semantic analysis SEAN provides a very simple editor for creating/editing ontologies for domains of interest. The generated semantic partitions are assigned concept labels by either matching keywords in the partition's content to those associated with concepts in the ontology or by applying concept classification rules to features extracted from the content. The keywords as well as the rules used for classification can both be edited. We point out that there has been extensive work on ontology tools and classifiers and in the future we plan on designing a plug-in architecture for SEAN that will support the use of any sophisticated ontology editing tools such as Protege [Protege, 2000], Shoe [Heflin *et al.*, 2003], OntoBroker [Fensel *et al.*, 1998], etc. and powerful statistical and rule-based classifiers such as Naive Bayes and decision trees [Mitchell, 1997] for doing semantic analysis.

In terms of related work, although a number of works partition a HTML page based on structural analysis, tools based on combining it with domain ontologies for semantic annotation are described in [Dill *et al.*, 2003; Handschuh and Staab, 2002; Handschuh *et al.*, 2003; Heflin *et al.*, 2003]. In [Handschuh and Staab, 2002; Handschuh *et al.*, 2003; Heflin *et al.*, 2003] powerful ontology management systems form the backbone that supports interactive annotation of HTML documents. The observation that semantically related items exhibit spatial locality in the DOM tree of the HTML document is not exploited in [Dill *et al.*, 2003]. As a result, their partitioning algorithm may fail to identify proper concept instances in template generated HTML pages.

References

[Dill *et al.*, 2003] Stephen Dill, Nadav Eiron, Daniel Gibson, Daniel Gruhl, R. Guha, Anant Jhingran, Tapas Kanungo, Sridhar Rajagopalan, Andrew Tomkins, John Tomlin, and Jason Yien. SemTag and Seeker: Bootstrapping the semantic web via automated semantic annotation. In *International World Wide Web Conference*, 2003.

[Fensel *et al.*, 1998] Dieter Fensel, Stefan Decker, Michael Erdmann, and Rudi Studer. Ontobroker: Or how to enable intelligent access to the WWW. In *11th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*, Banff, Canada, 1998.

[Handschuh and Staab, 2002] Siegfried Handschuh and Steffen Staab. Authoring and annotation of web pages in CREAM. In *International World Wide Web Conference*, 2002.

[Handschuh *et al.*, 2003] Siegfried Handschuh, Steffen Staab, and Raphael Volz. On deep annotation. In *International World Wide Web Conference*, 2003.

[Heflin *et al.*, 2003] Jeff Heflin, James A. Hendler, and Sean Luke. SHOE: A blueprint for the semantic web. In Dieter Fensel, James A. Hendler, Henry Lieberman, and Wolfgang Wahlster, editors, *Spinning the Semantic Web*, pages 29–63. MIT Press, 2003.

[Mitchell, 1997] Tom M. Mitchell. *Machine Learning*. McGraw Hill, 1997.

[Mukherjee *et al.*, 2003] Saikat Mukherjee, Guizhen Yang, and IV Ramakrishnan. Annotating content-rich web documents: Structural and semantic analysis. In *International Semantic Web Conference (ISWC)*, 2003.

[Protege, 2000] Protege, 2000. http://protege.stanford.edu.

Building an integrated Ontology within the SEWASIE project: the Ontology Builder tool

D. Beneventano, S. Bergamaschi, A. Fergnani, D. Miselli, M. Vincini
Department of Informatics Engineering
University of Modena and Reggio Emilia
Via Vignolese 905 - 41100 Modena - Italy
Email: lastname@dbgroup.unimo.it

1 Introduction

SEWASIE (SEmantic Webs and AgentS in Integrated Economies) (IST-2001-34825) is a research project founded by EU on action line Semantic Web (May 2002/April 2005) (http://www.sewasie.org/). The goal of the SEWASIE project is to design and implement an advanced search engine enabling intelligent access to heterogeneous data sources on the web via semantic enrichment to provide the basis of structured secure web-based communication. A SEWASIE user has at his disposal a search client with an easy-to-use query interface able to extract the required information from the Internet and to show it in an easily enjoyable format. In this paper we focus on the Ontology Builder component of the SEWASIE system, that is a framework for information extraction and integration of heterogeneous structured and semi-structured information sources, built upon the MOMIS (Mediator envirOnment for Multiple Information Sources) [Bergamaschi *et al.*, 2001] system.

The Ontology Builder implements a semi-automatic methodology for data integration that follows the Global as View (GAV) approach [Lenzerini, 2002]. The result of the integration process is a global schema which provides a reconciled, integrated and virtual view of the underlying sources, GVV (Global Virtual View). The GVV is composed of a set of (global) classes that represent the information contained in the sources being used and the mappings establishing the connection among the elements of the global schema and those of the source schemata. A GVV, thus, may be thought of as a domain ontology [Guarino, 1998] for the integrated sources. Furthermore, our approach "builds" a domain ontology as the synthesis of the integration process, while the usual approach in the Semantic Web is based on "a priori" existence of an ontology (or a list of different versions of an ontology). The obtained conceptualization is a domain ontology composed of the following elements (see figure 1):

- local schemata of the sources: formal explicit descriptions with a common language, ODL_{I3} [Bergamaschi *et al.*, 2001], of concepts (classes), properties of each concept (attributes), and restrictions on instances of classes (integrity constraints).
- annotations of the local sources schemata: each element (class or attribute) is annotated with its meanings according to lexical ontology (we use WordNet [Miller, 1995]).
- a Common Thesaurus: is a set of intensional and extensional relationships, describing intra and inter-schema knowledge about elements of sources schemata. The kind of relationships are SYN (synonym of), BT (broader term / hypernymy), NT (narrower term / hyponymy) and RT (related term/relationship).
- a Global Virtual View (GVV): it consists of a set of global classes and the mappings between the GVV and the local schemata. In our approach, each Global Class represents a concept of the domain and each Global Attribute of a Global Class a specification of the concept. It is possible to define ISA relationships between Global Classes and to use a Global Class as domain of a Global Attribute.
- annotations of the GVV: the GVV elements (classes and attributes) meanings are semi-automatically generated from the annotated local sources.

With reference to the Semantic Web area, where generally the annotation process consists of providing a web page with semantic markups according to an ontology, in our approach we firstly markup the local metadata descriptions and then we produce the annotation of the GVV elements.

2 The Ontology Integration phases

1. **Ontology source extraction**
   The first step is the construction of a representation of the information sources, i.e. the conceptual schema of the sources, by means of the common data language ODLI3. To accomplish this task, the tool encapsulates each source with a wrapper that logically converts the underlying data structure into the ODLI3 information model. For conventional structured information sources (e.g. relational databases, object-oriented databases), schema description is always available and can be directly translated. In order to manage a semi-structured source we developed a wrapper for XML/DTDs files. By using that wrapper, DTD elements are translated into semi-structured objects in the same way as OEM objects [Papakonstantinou *et al.*, 1995].
2. **Annotation of the local sources**
   The designer has to manually choose the appropriate WordNet meaning for each element of local schemata.

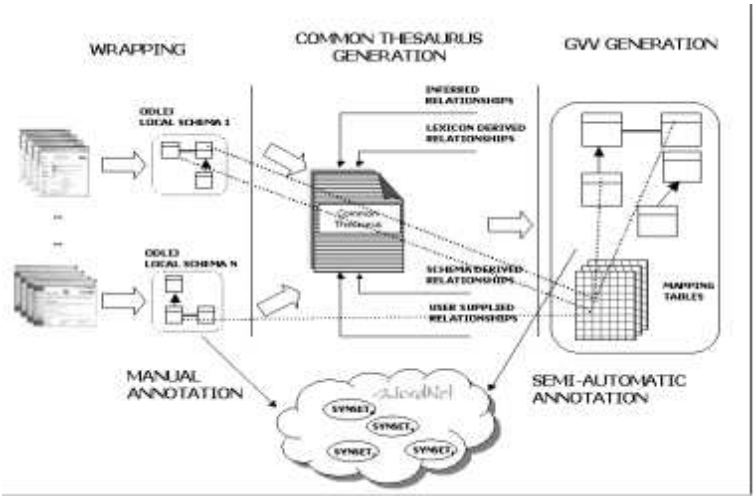


Figure 1: The Ontology Integration phases

First, the WordNet morphologic processor aids the designer by suggesting a word form corresponding to the given term, then the designer can choose to map an element on zero, one or more senses. If a source description element has no correspondent in WordNet, the designer may add a new meaning and proper relationships to the existing meanings.

3. **Common Thesaurus generation**
   The relationships of the Common Thesaurus are automatically extracted by analyzing local schemata description (for example in XML data files, ID and IDREF generate a BT/NT relationship and nested elements RT relationships), from the lexicon, on the basis of source annotation and of semantic relationships between meanings provided by WordNet, and inferred by using description logic inference techniques provided by ODB-Tools [Beneventano *et al.*, 1997].

4. **Affinity analysis of classes**
   Relationships in the Common Thesaurus are used to evaluate the level of affinity between classes intra and inter sources. The concept of affinity is introduced to formalize the kind of relationships that can occur between classes from the integration point of view. The affinity of two classes is established by means of affinity coefficients based on class names, class structures and relationships in Common Thesaurus.

5. **Clustering classes**
   Classes with affinity are grouped together in clusters using hierarchical clustering techniques. The goal is to identify the classes that have to be integrated since describing the same or semantically related information.

6. **Generation of the mediated schema (GVV)**
   For each cluster C, composed of a set S of local classes, a Global Class GC and mappings between global and local attributes are automatically defined. In particular, attributes of local classes in S related by SYN and BT/NT relationships in the Common Thesaurus are grouped and mapped into a single global attribute of GC.

7. **Annotation of the GVV**
   GVV elements (classes and attributes) meanings are semi-automatically generated from the annotated local sources. For a Global Class, the annotation is performed by considering the set of all its "broadest" local classes w.r.t. the relationships included in the Common Thesaurus. In particular the union of the meanings of the local class names in are proposed to the designer as meanings of the GVV and the designer may change this set, by removing some meanings or by adding other ones. For a Global Attribute, we use the same method starting from the set of local attributes which are mapped into it.

References

[Bergamaschi *et al.*, 2001] Sonia Bergamaschi, Silvana Castano, Domenico Beneventano and Maurizio Vincini. Semantic Integration of Heterogeneous Information Sources. DKE, Vol. 34, Num. 1, pages 215–249, Elsevier Science B.V., 2001.

[Lenzerini, 2002] Maurizio Lenzerini. Data Integration: A Theoretical Perspective. PODS, pages 233–246, 2002.

[Guarino, 1998] Nicola Guarino. Formal Ontology in Information Systems. In *N. Guarino (ed.)*, FOIS'98, 1998.

[Miller, 1995] A. G. Miller. A lexical database for English. Communications of the ACM, 38(11):39:41, 1995.

[Papakonstantinou *et al.*, 1995] Y. Papakonstantinou, H. Garcia-Molina, J. Widom. Object exchange across heterogeneous information sources. In *Proceedings of ICDE '95*, 1995.

[Beneventano *et al.*, 1997] Domenico Beneventano, Sonia Bergamaschi, Claudio Sartori and Maurizio Vincini. ODB–QOPTIMIZER: A tool for semantic query optimization in oodb. In *Proceedings of ICDE '97*, 1997.

Posters

Semantic Web Technologies for Economic and Financial Information Management*

J. L. Alonso[1], C. Carranza[2], P. Castells[2], B. Foncillas[1], R. Lara[3], M. Rico[2]

[1] Tecnología Información y Finanzas
C/ Español eto 19, 28010 Madrid
{jalonso, bfoncillas}@afi.es

[2] Universidad Autónoma de Madrid
Ctra. de Colmenar Viejo km. 15, 28049 Madrid
{cesar.carranza, pablo.castells, mariano.rico}@uam.es

[3] Universität Innsbruck, Institut für Informatik (IFI)
Techniker Strasse 13, 6020 Innsbruck, Austria
ruben.lara@uibk.ac.at

Abstract

We present an ontology-based platform for economic and financial content management, search and delivery. Our goals include a) the development of an ontology for the domain of economic and financial information, b) the integration of contents and semantics in a knowledge base that provides a conceptual view on low-level contents, c) an adaptive hypermedia-based knowledge visualization and navigation system, and d) semantic search facilities.

1 Introduction

The field of economy and finance is a conceptually rich domain where information is complex, huge in volume, and a highly valuable business product by itself. A massive amount of valuable information is produced worldwide every day, but no one is able to process it all. Efficient filtering, search, and browsing mechanisms are needed by information consumers to access the contents that are most relevant for their business profile, and run through them in an effective way.

The finance community is a major spender in information technology. The web has created new channels for distributing contents, to which more and more activity and information flow has been shifting for more than a decade. The new web technologies are enabling a trend away from monolithic documents, towards the emergence of new content products that consist of flexible combinations of smaller content pieces, fitting different purposes and consumers, and procuring a more efficient capitalization and reuse of produced contents.

Along this line, a number of XML standards for financial contents and business have been defined during the last few years, like FpML, XBRL, RIXML, ebXML, NewsML, IFX, OFX, MarketsML, ISO 15022, swiftML, MDDL, to name a few [Coates, 2001]. Most of them are concerned with describing business processes and transactions. Some, like XBRL, RIXML and NewsML, do focus on content structure and provide a rich vocabulary of terms for content classification. Our assessment is that these vocabularies need significant extensions when faced to the actual needs of content managers that deal with advanced financial information. More insightful semantics and a sharper level of representation are required to describe and exploit complex information corpora.

The purpose of our work is to achieve an improvement in current Internet-based economic information management practice by adopting Semantic Web technologies and standards in a real setting. We have undertaken a joint project involving a content provider in this field, and two academic institutions, aiming at the development of an ontology-based platform for economic and financial content management, search and delivery. The specific technical goals of this project are:

- Define an ontology for the economic and financial information domain.
- Develop ontology-aware tools for content provision and management.
- Develop a hypermedia-based module for content visualization and semantic navigation in web portals.
- Support semantic search in terms of the economic and financial information ontology.
- Include a user modeling component to be used in navigation and search.

2 Financial and Economic Information Providers

Tecnología, Información y Finanzas (TIF), is part of a company corporation that generates high-quality economic information (equity research notes, newsletters,

* This work is funded by the Spanish Ministry of Science and Technology, grants FIT-150500-2003-309, TIC2002-1948.

analysis, sector reports, recommendations), and provides technologic solutions for information consumers to access, manage, integrate and publish this information in web portals and company intranets.

The consumer profile of this information is diverse, including financial institutions, banks, SMEs that use the information in decision making and foreign trade activity, and distributors who publish the information in first-rank printed and digital media about Spanish economic activity. Adequating the information and delivery procedures to such heterogeneous customer needs, interests, and output channels, is quite a challenge.

A large group of professionals and domain experts in the company is in charge of generating daily economic, market, bank, and financial analyses, commercial fair reports, import/export offers, news, manuals, etc. This information is introduced in the company database, which feeds the automatic delivery systems and web sites. Contents are organized and processed on the basis of a conceptual model (in expert's mind), a vocabulary for information structures and classification terms, which is driven by market needs and reflects the view of the company on the information products it deals with. This model is present somehow in the current TIF software system for information management: it is implicit in the design of the database. As a consequence the possibilities to reason about it are fairly limited.

3 A Semantic Knowledge Base for Economic and Financial Information

Our first endeavor in this project is to wrap the current databases where contents are stored into a knowledge base that provides a conceptual ontology-based view of the information space, above the low level content storage system.

We have built an ontology where the conceptual model of TIF is explicitly represented. It includes concepts like MutualFund, IndustrySector, CommercialFair, EconomicIndicator, CompanyReport, TechnicalAnalysis, FinancialAnalyst, Publisher, Association, and BusinessOpportunity, relations between such concepts, and several classification hierarchies for subject topics, industry sectors, intended audience, and other content fields. In this ontology, the old data model has been transformed and augmented with explicit semantics, and enriched with collected domain expertise from TIF. We have integrated the RIXML classification schemes as well, extending and adapting them to support the TIF concepts, terminology, and views. We have defined a mapping from our ontology to RIXML and NewsML formats. The conversion from our ontology to these standards implies a (meta)information loss, in exchange for a wider potential dissemination.

The knowledge base can be queried and browsed directly in terms of the conceptual view. Meaningful queries can be expressed in terms of the vocabulary provided by the ontology, improving current keyword-based search. The database from which actual contents and data are retrieved has not been redesigned, which would have implied a major cost and a disruption for a critical service that needs to keep going. Instead, we have developed a gateway that dynamically maps ontology instances to (combinations of) database records and fields.

The tools for inputting contents have been adapted to allow defining richer semantics in terms of the ontology. Content managers themselves are users of a highly expressive version of the search and browsing facilities. Efficiency and precision in locating the right contents, and ease of navigation through them, are essential for authors who classify and link pieces together to define global information structures.

The explicit ontology allows more meaningful and precise user profiles, which can express preferences on specific topics, content classes, or even abstract content characterizations. User profiles are taken into account by the adaptive hypermedia-based visualization and navigation module, which is based on our previous work on Pegasus [Castells, 2001]. It uses an explicit presentation model, defined in a fairly simple language, where parts of a semantic network can be easily referenced, and conditions over the user model can be expressed. Presentation models are associated to ontology classes, and define what parts (attributes and relations) of a class instance must be included in its presentation, their visual appearance and layout.

5 Conclusions

The development of a significant corpus of actual Semantic Web applications has been acknowledged as a necessary achievement for the Semantic Web to reach critical mass [Haustein, 2002]. The project presented here intends to be a contribution in this direction. It takes up our previous research work on Semantic Web user interfaces and adaptive navigation systems [Castells, 2001], and will provide a testing ground for our past and future research.

The system is currently under active development. Protégé RDF(S), Jena 2, and RDQ are used to build, represent, parse, and query the ontology. A full implementation of the system is scheduled to be released by the beginning of 2004.

References

[Castells, 2001] P. Castells and J. A. Macías. An Adaptive Hypermedia Presentation Modeling System for Custom Knowledge Representations. *World Conference on the WWW and Internet (WebNet 2001)*. Orlando, 2001.

[Coates, 2001] A. B. Coates. The Role of XML in Finance. *XML Conference & Exposition 2001*. Orlando, Florida, December 2001.

[Haustein, 2002] S. Haustein and J. Pleumann. Is Participation in the Semantic Web too Difficult? *International Semantic Web Conference (ISWC 2002)*. Sardinia, Italy, 2002.

MIKSI: A semantic and service oriented integration platform for cultural institutions

Aleksandar Balaban, Alexander Wahler, Bernhard Schreder, René Androsch, Klaus Niederacher
NIWA WEB Solutions
A-1070 Wien, Kirchengasse 13/1A
{balaban, wahler, schreder, androsch, niederacher}@niwa.at

1 Motivation

The emergence of Web Service technology and the evolution towards the Semantic Web offer new opportunities to automate e-business and to provide new value added services. The MIKSI project addresses the business of cultural institutions and focuses on Web Services for digital administration of members, cooperation-partners, sponsors, journalists and event-data and interactive services for journalists, cultural workers and their customers. The goal of the MIKSI project is to define and implement a service oriented integration platform, which provides pluggable and reusable components, defined as atomic services, XML based semantic description of business processes, ongoing tasks controlled by a process flow composition engine and a possibility to perform dynamic service discovery and composition based on user-defined goals and a knowledge base (KB). The knowledge base contains semantic descriptions about the capabilities of registered atomic services.

2 Goals and proposed Solutions

The MIKSI integration platform will be developed to support several key requirements for effective service finding, process composition and integration with third party applications:

- *basic components realized as simple well defined internal objects or external web services.*
- *business processes described in high level manner as XML documents.*
- *high level processes performed through an engine that provides run time composition based on semantic process description.*
- *integration of dynamic sub-processes in a static described process flow depending on pre defined goals and user interactions.*
- *efficient knowledge base about capabilities of atomic services that supports dynamic service finding and process composition.*

Building on Business Process Execution Language for Web Services (BPEL4WS [2], [8]) MIKSI uses a novel approach to integrate Semantic Web technology and dynamic composition of services into process flows. MIKSI benefits from using the advantages of the process definition in BPEL4WS and the process execution by BPWS4J engine [4] by IBM used for the development of MIKSI (sessions managing, concurrency, error handling, automatically publishing as Web service) and extends the functionality by adding possibilities which provide on demand dynamic composition embedded into the static description of the BPEL process.

BPWS4J engine performs composition of atomic (web) services based on descriptions which must be defined and hard coded at the development time. BPEL4WS has a rich set of statements and controls to define business process flows with sequences, flows, loops, branching, concurrency, transactions and error handling. But dynamic service composition at run time is supported neither from BPEL4WS specification nor from BPWS4J engine and it will be a MIKSI specific extension of BPWS4J implementation.

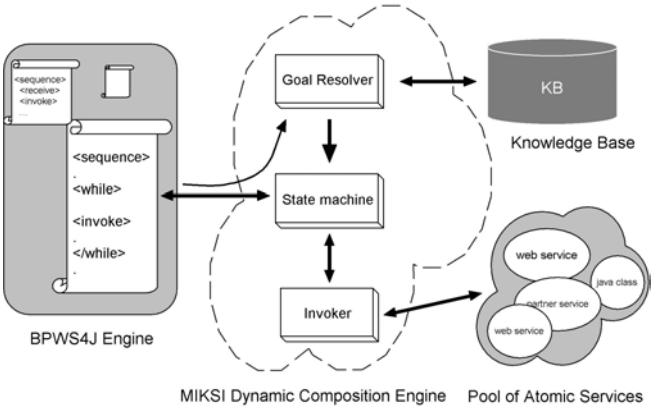


Figure 1 Components of MIKSI Platform with Composition Engine and its main components**.**

The dynamic service composition, performed by the MIKSI Composition Engine, is embedded into the BPEL document and invoked at run time through passing the goal as arguments to the engine.

MIKSI Composition Engine will support different phases of dynamic composition. Main parts of the MIKSI Composition Engine are:

- ***Goal resolver**, which translates the goal into a sequence of atomic web services.*
- ***State machine,** which manages the advancement towards the goal during the dynamically called invocation steps and signals the fulfilment of the goal back to the BPWS4j engine (see loop in control flow diagram below).*
- ***Invoker,** that invokes an atomic service in one composition step.*

A prototype of the MIKSI Composition Engine will be realized in Java.

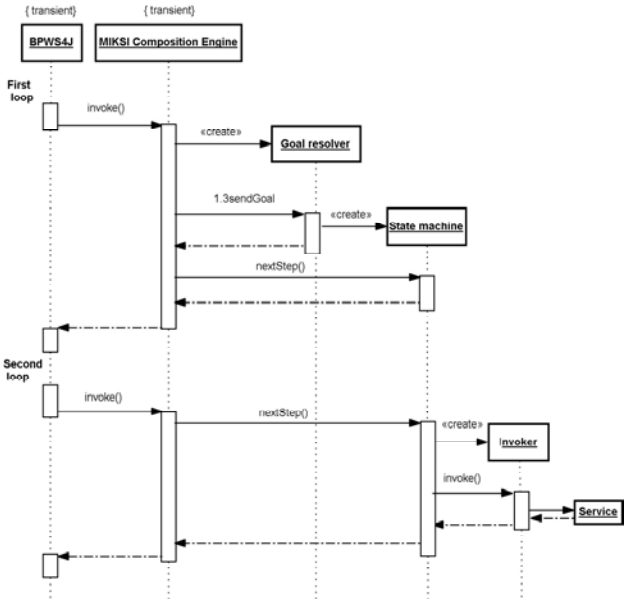


Figure 2: Control flow diagram of begin of dynamic composition process**.**

The goal resolver uses semantic descriptions of Web Services involved in the service composition. Description of services are modeled as ontology which provide understanding of what one atomic service can provide and how to use its functionality in correlation to other services in composition scenarios. This solution offers the possibility to deploy new services by simple describing the capabilities in the knowledge base. It provides a fast extensible environment of the MIKSI service-platform without extra programming effort! As a first example a "press release service", which support the composition of newsletters, folders, etc out of different heterogeneous data-sources (e.g. address database, event-database) is modeled and implemented in the MIKSI platform.

3 Conclusions

The MIKSI platform will be a solution for services oriented applications using well defined processes with mixed static and dynamic service definitions. Building on BPEL4WS the MIKSI Composition Engine enables a dynamic service composition using semantic descriptions, which are mapped to ontologies.

More information: www.miksi.org

4 References

[1] Berners-Lee T., Hendler J. and Lassila O. *The semantic Web*. Scientific American (May2001).
[2] Business Process Execution Language for Web Services Version 1.1. http://www.siebel.com/bpel
[3] Daniel J. Mandell and Sheila A. McIlraith, Knowledge Systems Lab, Stanford University . *Adapting BPEL4WS for the Semantic Web with a Semantic Discovery Service*. http://ksl.stanford.edu/sds.
[4] The BPWS4J platform http://www.alphaworks.ibm.com/tech/bpws4j
[5] DAML Services Coalition. DAML-S version 0.5,0.6 and 0.7. http://www.daml.org/services/
[6] D. Roller, M.-T. Schmidt, F. Leymann, *Web services and business process management*, IBM SYSTEMS JOURNAL, VOL 41, NO 2, 2002
[7] T. Andrews, F. Curbera, H- Dholokia, Y. Goland, J. Klein, F. Leymann, K. Liu, D. Roller, D. Smith, S. Thatte,Trickovic and S. Weerawarana. *Business Process Execution Language for Web Services, Version 1*.1, May 2003.
[8] Curbera, F., Goland, Y., Klein, J., Leyman, F., Soller, D., Thatte, S., Weerawarana, *S., Business Process Execution Language for Web Services,* BEA Systems & IBM Corporation & Microsoft Corporation, 2002, http://www-106.ibm.com/developerworks/library/ws-bpelwp

Semantic Web Search Engines: the SEWASIE approach

D. Beneventano[1,2]**, S. Bergamaschi**[1,2]**, D. Montanari**[3,1] **and L. Ottaviani**[1]
[1] Dipartimento di Ingegneria dell'Informazione - Università di Modena e Reggio Emilia
Via Vignolese 905, 41100 Modena, Italy
[2] IEIIT-BO Bologna - V.le Risorgimento 2, 40136 Bologna, Italy
[3] EniData, Viale Aldo Moro 38, 40127 Bologna, Italy
{ domenico.beneventano,sonia.bergamaschi,laura.ottaviani}@unimo.it, Daniele.Montanari@enidata.it

1 Introduction

Search engines were born with the web, to provide tools for finding information. However, they showed two major drawbacks very early on, namely

- (*coverage*) they could not keep the pace with the growth of the web, so that their coverage was limited to a fraction of the available information and continued getting worse
- (*expressiveness of query language*) the keyword–based retrieval mechanism is a token–level pattern matching scheme possibly augmented with logical operators to express logical conjunction, disjunction, *et cetera*, with no semantic dimension being considered, so that possible answers for all the possible meanings of a keywords have to be harvested, giving the user a very long list of responses to wade through

Even the better contemporary search engines still face these issues. In the second half of the 90's several ideas were developed to attack these problems. In particular

- (*distribuited architecture*) the centralized architecture of the current search engines is a severe limit to their function; distributed architectures like the internet and the web naturally call for tools which are distributed, autonomous, and adapted to local needs and opportunities
- (*agent–based approach*) agent features may provide further adaptability, robustness, and enforcement of general policies
- (*semantic dimension*) the terms used by the content providers to label information and by the content seekers to formulate queries should be enriched with meaning, taking the respective context into account both when the information is readied for presentation (provider side) and when the user query is expressed (seeker side)

The SEWASIE project (*SEmantic Webs and AgentS in Integrated Economies*, IST-2001-34825) [The SEWASIE Consortium, 2002] is a 3–year research and development project partially sponsored by the European Union to design and develop an advanced search engine and an integrated user environment for the exploitation of semantically enriched data. The partners (Università di Modena e Reggio Emilia, CNA Servizi Modena, Rheinisch–Westfaelischen Technischen Hochschule Aachen, Università di Roma "La Sapienza", Libera Università di Bolzano, Thinking Networks AG, IBM Italia, and Fraunhofer FIT) have joined their efforts by leveraging their experience in the fields of mediator systems, agent architectures, ontologies, query management, user interfaces, negotiation support and OLAP tools, and integrating their technical expertise with direct user need identification, result evaluation in the field, and support of the exploitation of the technological results.

The following sections describe the architecture of the SEWASIE system, its major components, and the current status of the project.

2 Architecture and strategic goals

In the context described above the SEWASIE vision springs up from the following specific points

- a basic architecture should comprise *information providers*, *intermediaries*, and *information seekers*; each actor should be as autonomous as possible;
- providers and seekers must be able to express available information and needs/questions in the most natural way; in particular, multi–lingual issues have to be addressed;
- queries to the system are handled by *query agents*, which are responsible of supporting the query management in the large;
- intermediaries (*brokering agents*) must support the match of requests and available information; this matching is supported by collecting semantic information from information providers, exchanging it among intermediaries, and connecting it in a (partial) global view mapping concepts among locally istantiated ontologies and remotely istantiated ones.

The architecture is expected to be able to support two different scenarios, namely the *narrow–deep* scenario (relatively few nodes, limited domains, and strong central control), which is expected to be more limited in scope and diffusion but characterised by a well–defined and controlled semantic domain, and the *wide–shallow* scenario (many nodes, unlimited domains, and no central control), where scope and diffusion are wider while the number and variety of involved semantic domains is higher and leads to lighter mappings of

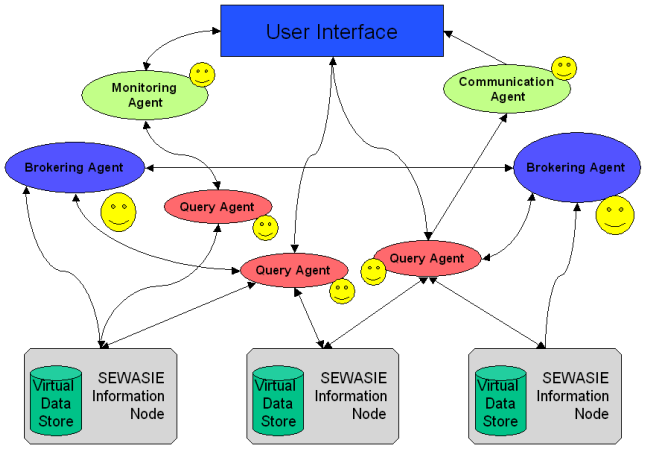


Figure 1: General Overview of the architecture

ontologies in the definition of a local view of the global system by the single Brokering Agent.

Some features, and most notably the security ones, will not be researched by the project but will be kept under scrutiny to make sure that they are always taken into account since the early design and development phases of the project.

Finally, the resulting system must be *exploitable*, i.e. its adoption should be smooth and progressive, allowing for return of investment proportionate to the correpsonding effort, and with a reasonably easy and enticing entry point.

SEWASIE Information Node (SINode)

A SEWASIE INformation Node is a basic integrated information providing node. This element may be defined by administrative convenience and alignment with organizational constraints, while the most relevant feature is the complete integration of the ontologies being involved in the semantic enrichment of the sources comprising the node. Tools are defined here to support the ontologies definition and integration. Each node publishes the resulting integrated ontology to a Brokering Agent, which will map it within a wider ontological context, including those of the underlying nodes and the references to those maintained by other Brokering Agents. The SINodes also support the query management within their scope.

Brokering Agents

The Brokering Agents define the ”semantic routing” structure of the system. These agents maintain mappings among the *underlying SINodes*, namely the SINodes which export their ontological information directly and fully to the Brokering Agent, and the (less detailed) ontological information exchanged by the Brokering Agents. The Brokering Agents may work in an autonomous mode, establishing mostly basic mappings, or in a human–supported mode where the support of the human expert may introduce further enrichment.

It is expected that Brokering Agents will be established by entities managing SINodes, but also by third parties which may have no underlying node and rather specialise in a specific domain, where they plan to act as *pure informants* and develop an autonomous expertise.

Query Agents

Query Agents are in charge of managing the overall query management, outside the SINodes. Each Query Agent is defined by a user interface with a user query on board, it addresses a Brokering Agent with the query, receives directions to the appropriate SINodes or other Brokering Agents, addresses the SINodes, receives any returns, and reconciles the (partial) answers into a coherent user relevant whole.

Communication Agents

The Communication Agent is responsible for finding and contacting potential business partners, asking for initial offers, and ranking these. Human negotiator can then decide and choose the best offer to begin negotiating with support by the communication tool.

Monitoring Agents

The Monitoring Agents are responsible for monitoring information sources according to user profiles. At regular time intervals the Monitoring Agent issues new query agents to get the desired information. It then filters monitored information with respect to user profiles and may also display a “difference view” concerning the history of information that has been seen by the user previously.

User Interface

The user interface comprises all the user services which may be available in any given environment. Users (mostly in narrow–deep environments) may have special instruments available to use the results of queries, e.g. for analytical processing and negotiation purposes in an economic environment. The most relevant service provided is the ability to disambiguate the user query by annotating the same with respect to user ontologies, and to translate the initial (local language) formulation into a neutral intermediate one in the process.

3 Current status of the project and future plans

The project has reached the end of its first year of activity. The architecture has been completely defined and the first prototypes (global virtual view definition tool at the SINode level, and query management within the SINode) have been developed and demonstrated. More prototypes of relevant components are under development (Brokering Agents and User Interface) and will be readied by the end of the second year of activities. Integration with other initiatives (European and worldwide) will also be sought to exploit synergies, and to contrast different approaches.

References

[The SEWASIE Consortium, 2002] The SEWASIE Consortium. Project website *http://www.sewasie.org/*, 2002-2003.

Incremental Formalization of Document Annotations

Jim Blythe and Yolanda Gil
USC Information Sciences Institute,
4676 Admiralty Way, Marina del Rey, CA 90292
{blythe,gil}@isi.edu

Abstract

Manual semantic markup of documents will only be ubiquitous if users can express annotations that conform to ontologies (or schemas) that have shared meaning. But any given user is unlikely to be intimately familiar with the details of that ontology. We describe an implemented approach to help users create semi-structured semantic annotations for a document according to an extensible schema or ontology. In our approach, users enter a short sentence in free text to describe all or part of a document, then the system presents a set of potential reformulations of the sentence that are generated from valid expressions in the ontology and the user chooses the closest match. We use a combination of off-the-shelf parsing tools and breadth-first search of expressions in the ontology to help users create valid annotations starting from free text. The user can also define new terms to augment the ontology, so the potential matches can improve over time. The annotations should ideally follow the ontology as closely as possible, while allowing users who may not know the terms in the ontology to make statements easily and deviate from the formal representation of the ontology if they so desire.

Introduction

Semantic annotations of documents are useful to qualify their contents, enable search and retrieval, and to support collaboration. In some approaches, these annotations can be extracted automatically from the document. In other approaches, the annotations are manually created by users. Handcrafted annotations may be more accurate but more importantly they enable users to reflect their opinions or their own analysis of the document. However, expressing these annotations formally is difficult for web users at large and is a challenge that must be addressed if semantic annotation tools are to become widely accessible.

Our approach is to enable users to express annotations in concise free text statements and then help them formalize the statement partially or totally by mapping it to an existing schema or ontology. Given a free text statement, the annotation system creates plausible paraphrases of the sentence generated using the ontology and presents them to the user as possible canonical forms of their original statement. If new terms appear in the statement, the system will suggest to the user possible extensions to the ontology that incorporate the new terms. To generate the plausible paraphrases, the system makes use of a parser and a beam search of expressions within the ontology.

Our work extends the TRELLIS annotation tool that enables users to express their analysis of possibly contradictory information sources [Gil and Ratnakar, 2002]. In TRELLIS, each statement in the analysis is formulated in free text, and linked to other statements through a set of domain-independent formal constructs for argumentation, expressed in a semantic markup language. The Canonicalizer tool, described in this paper, extends TRELLIS by helping users to incrementally formalize the text statements according to a domain ontology in OWL.

We illustrate the approach with a scenario drawn from professional sports where teams sign players amidst much controversy and rumors, causing many press articles with dissenting views as well as many on-line discussions of opinionated fans. Here, a user may want to annotate a certain news item, for example with his conclusion reached after reading it that a certain team is very likely to sign a certain player. Consider a conclusion, for example, that a particular football club, West Ham, wishes to sign attacking players who are currently playing in the top league in that country, the English Premier League (EPL). Two users may express this same conclusion using two very different statements, for example ”West Ham are targeting strikers from the EPL” and ”WHU prefer forwards who play in the Premier League”. It is not our aim to match such pairs of phrases in all cases -such a task would require a deep understanding of the sentences that is beyond the state of the art. However, even partial reformulations of the sentences would be useful if they help expose their similar meanings. This will improve the likelihood that a search engine would detect the similarities of both analyses. Thus, the task of the Canonicalizer is to suggest reformulations of a concise text statement that conform as much as possible with the desired ontology or schema.

The Canonicalizer brings together three techniques to help with this task. First, it performs a substring match on the sentence against the terms defined in the ontology and suggests re-writing specific terms with their canonical values. For ex-

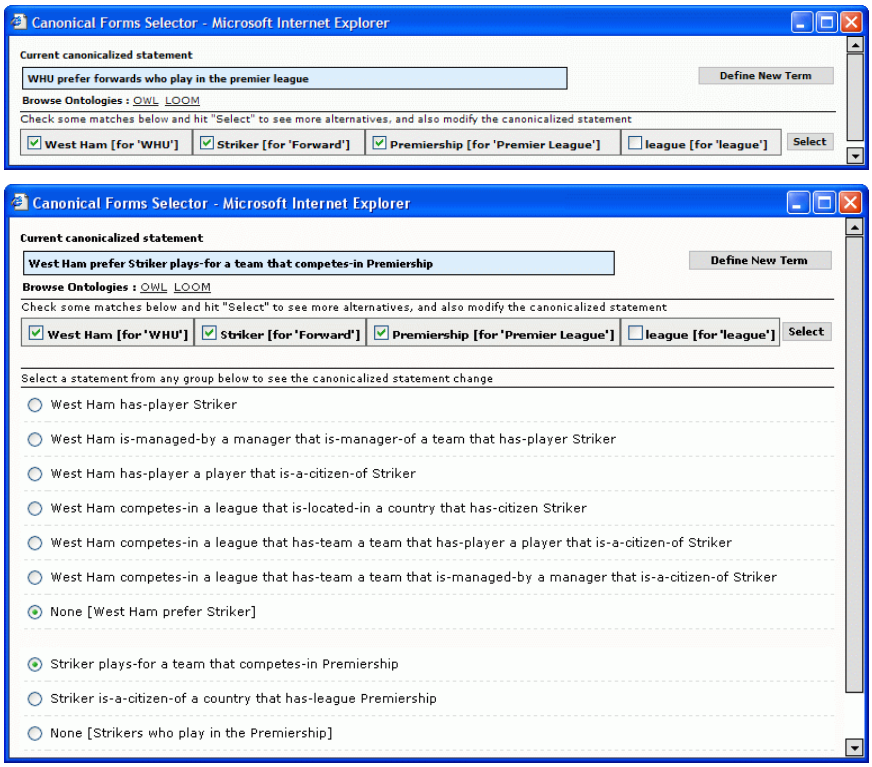


Figure 1: The Canonicalizer suggests reformalizations of the original text.

ample, in the second sentence above, the tool might suggest to replace ”forwards” with ”strikers” based on the known synonyms of that class. Second, it uses an off-the-shelf parser to generate information about the sentence that can help simplify it, for example to find determiners or passive verbs. The use of the parser is robust in the sense that reformulations can be suggested even if the tool fails to parse the sentence or returns an incorrect parse. Finally, we make use of the ontology again to search for plausible compositions of relations and classes that can link the matched terms. At each step, we make suggestions to the user rather than reformulate the sentence automatically. This process may be partial, leaving part of the sentence unconverted and generating an annotation that includes some text as well as some expressions generated in the markup language.

Figure 1 shows at the top the suggestions that result from the first step, and at the bottom the suggestions that result from the latter steps. In the first step, synonyms for simple terms in the ontology are replaced using a sub-string match. On its own this step clearly contributes to putting the sentence in a regular form, but another purpose is to confirm with the user some of the known entities in the domain. Next, the tool uses the Link Grammar Parser [Sleator and Temperly, 1993] to identify words that should be ignored during the final composition step, such as cardinals (’They want 2 strikers’) or negative particles (”Liverpool did not sign Ronaldo’).

Finally we search for plausible compositions of relations and terms in the ontology that match terms and other words found in the user’s sentence. A forward beam search is made through the space of valid compositions of expressions, made up of relations, classes, instances and event templates. The search returns the shortest expressions that include a set of requested words, possibly including synonyms for the terms. It then generates a sentence encoding the expression for the user to consider. If no expressions match all the requested words, paths are used that match are subset of the words, weighted according to how many words are matched and whether synonyms are used. This approach was originally applied to help users build complex expressions of problem-solving knowledge, as described in [Blythe, 2001]. Notice that the system is disambiguating the text. For example, the phrase ”players from the Premier League” might refer to players who play in the premier league now, or who have been transferred from there, or who were born in the same country.

Users can also add terms to the ontology by selecting a portion of the statement and choosing where the new term should be inserted in the class hierarchy. Currently only classes are added, but other terms will be included in future versions.

References

[Blythe, 2001] Blythe, J. 2001. Integrating expectations from different sources to help end users acquire procedural knowledge. Proc. IJCAI-01.

[Gil and Ratnakar, 2002] Gil, Y. and Ratnakar, V. Trusting Information Sources One Citizen at a Time. In Proc. EKAW-02.

[Sleator and Temperley, 1993] Sleator, D. and Termperley D., Parsing English with a link grammar, Proc International Workshop on Parsing Technologies, 1993

Implementing DISCourse-driven hypermedia Presentations

Stefano Bocconi, Joost Geurts and Jacco van Ossenbruggen
CWI, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands,
Firstname.Lastname@cwi.nl

1 Introduction

DISC [Geurts *et al.*, 2003] is an automatic hypermedia presentation generation engine. DISC generates presentations based on a user-specified subject. The domains that we have handled to date are the musea for fine arts, specifically the Rijksmuseum in Amsterdam.

Presentations are created using two types of knowledge: discourse and narrative knowledge and subject domain knowledge. The former allows selection of appropriate presentation genres and creation of narrative structures, while the latter is used to select the content of the presentations.

We designed the system to be applicable to different domains, consequently, to avoid domain dependency, domain knowledge is not used directly but through an internal ontology. The internal ontology encodes the discourse and narrative knowledge, making explicit all the knowledge the system uses.

2 The Technical Framework

DISC [1] is currently being implemented using the Apache Cocoon framework. This framework provides an XSLT [Clark, 1999] transformation engine and active server pages (eXtensible Server Pages, XSP [The Apache Software Foundation., 1999]) for dynamic XML content. DISC generates SMIL [W3C, 2001] output (using the library developed for Cuypers [van Ossenbruggen *et al.*, 2001]).

The instances of the domain ontology form a semantic graph, i.e. a graph whose nodes are all the annotated information elements that can be selected for a presentation and whose edges are the semantic relations relating those information items. Both the domain ontology, the internal ontology and their instances are RDF(S)-encoded and stored in Sesame, an Open Source RDF Schema-based repository and querying facility. DISC uses a SQL-like RDF-aware query language called RQL to retrieve the data from Sesame, and plans are to migrate to SeRQL (for references about Sesame, RDF and SeRQL see [Broekstra *et al.*, 2002]).

3 The Interface

Via a web interface DISC presents the user with the choice of possible presentation genres, like a biography or a CV. These

[1] An on-line demo of DISC can be found at `http://media.cwi.nl:8080/demo/i2rp/`.

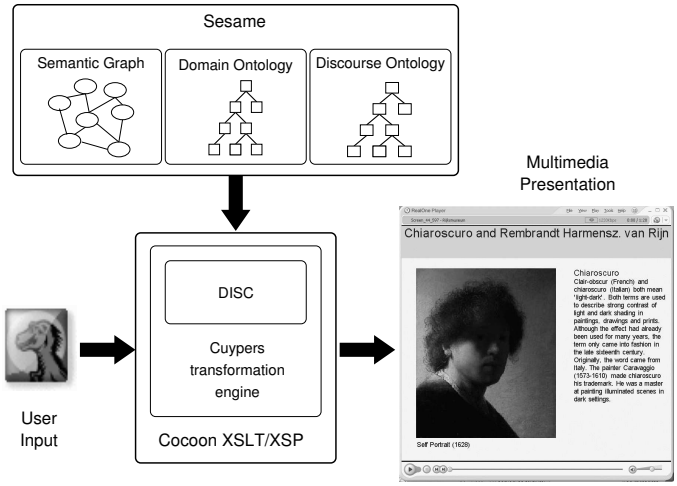


Figure 1: The multimedia presentation generation architecture

genres are retrieved by querying the internal ontology for instances of subclasses of the class *genre*. Each such instance has an attribute specifying the class of subjects (from the domain ontology) it can handle, e.g. a biography can handle instances of the class *Person* while an artist biography can handle instances of the class *Artist* (subclass of Person). This is one of the explicit mappings from the internal ontology to the domain ontology the system uses to be applicable to different domains.

Once the user has made a choice, DISC retrieves all instance from the domain ontology belonging to the selected class, e.g. Caravaggio, Rembrandt, etc. if the chosen genre is artist biography. The user can now use the web interface to select the subject of the presentation and DISC has enough user input to generate it.

4 The Discourse Knowledge

Each genre in the internal ontology contains narrative units: these are the building block of a presentation and can be seen as the chapters of the overall story, e.g. in case of artist biography the *career* narrative unit, the *private life* narrative unit, etc. Every such narrative unit contains rules to select multi-

media content to include in the presentation. A rule is basically looking for roles, e.g. in a biography the *Main Character* (this is user given), the *Spouse*, the *Offspring*, the *Teacher* and the *Pupil* in case of an artist biography, etc. Roles are found by querying the semantic graph for instances that have a particular semantic relation (from the domain ontology) with a character which is already part of the presentation.

When a role is found, the related (multimedia) information is added to the presentation. Every newly found character for the story can be the main character of a secondary branch of the main story. Characters do not need to be human, a painting style (e.g. Chiaroscuro) or a movement (e.g. the Caravaggist) can be the main or a secondary character in a biography or in another genre.

5 Conclusions

To date, we have only generated short presentations and focused on a single discourse structure (the biography). Using the Semantic Web-based framework described, the prototype selects relevant content from a semantically annotated information source and structures it into a multimedia presentation. More research is needed to scale these aspects of the system to more realistic scenarios.

5.1 DISC uses RDF-encoded Rules

Role-based rules can create complex narratives, more complex than when using templates, due to the recursive expansion of narrative units. On the other hand, this complexity needs to be dealt with by the designer of the rules.

Ontology languages such as RDF Schema are not designed for expressing rules. Therefore our rules are forced to be simple. For example, one cannot combine rules using logical AND or OR, or make one rule dependent on the outcome of another. A next step is to investigate the use of more powerful rule languages such as RuleML [Boley *et al.*, 2001] for expressing the rules within the system.

5.2 DISC uses Explicit Knowledge

All the intelligence of the engine creating the presentation is RDFS-encoded and explicit. The internal ontology is also used as a logical configuration tool (and also graphical, if using a graphical ontology editor like Protege-2000 [Grosso *et al.*, 1999])[2]. The ontology defines thus the framework a narrative designer would use to define his of her particular form of narrative.

5.3 DISC can handle Different Domains

All important domain relations are mapped to internal relations. This explicit mapping localizes all specific domain knowledge in the instances of the internal ontology. This has the advantage that the remaining transformations always deal with known internal concepts and are therefore reusable for different domain ontologies.

[2]Protege screen shots, the RDFS ontologies used and the online Sesame repositories can be found at: `http://www.cwi.nl/~media/conferences/ISWC2003/`.

5.4 Future Work

We are currently investigating what kind of rules can lead to more interesting narratives and the best way to encode them. In addition, we are investigating the expressiveness and granularity of the domain ontology in relation to the quality of the content selection process.

Acknowledgments

Part of the research described here was funded by the Dutch national NWO/NASH and ToKeN2000 I^{2}RP projects. We like to thank Lloyd Rutledge for his useful feedback.

References

[Boley *et al.*, 2001] Harold Boley, Said Tabet, and Gerd Wagner. Design Rationale of RuleML: A Markup Language for Semantic Web Rules. In *Semantic Web Working Symposium (SWWS)*, Stanford University, California, USA, July 30 – August 1, 2001.

[Broekstra *et al.*, 2002] Jeen Broekstra, Arjohn Kampman, and Frank van Harmelen. Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. In Ian Horrocks and Jim Hendler, editors, *The Semantic Web - ISWC 2002*, number 2342 in Lecture Notes in Computer Science, pages 54–68, Berlin Heidelberg, 2002. Springer.

[Clark, 1999] James Clark. XSL Transformations (XSLT) Version 1.0. W3C Recommendation, 16 November 1999.

[Geurts *et al.*, 2003] Joost Geurts, Stefano Bocconi, Jacco van Ossenbruggen, and Lynda Hardman. Towards Ontology-driven Discourse: From Semantic Graphs to Multimedia Presentations. In *Second International Semantic Web Conference (ISWC2003)*, Sanibel Island, Florida, USA, October 20-23, 2003. To be published.

[Grosso *et al.*, 1999] W.E. Grosso, H. Eriksson, R.W. Fergerson, J.H. Gennari, S.W. Tu, and M.A. Musen. Knowledge Modeling at the Millennium (The Design and Evolution of Protege-2000). Technical Report SMI Report Number: SMI-1999-0801, Stanford Medical Informatics (SMI), 1999.

[The Apache Software Foundation., 1999] The Apache Software Foundation. XSP Logicsheet Guide. See http://xml.apache.org/cocoon/userdocs/xsp/logicsheet.html, 1999.

[van Ossenbruggen *et al.*, 2001] Jacco van Ossenbruggen, Joost Geurts, Frank Cornelissen, Lloyd Rutledge, and Lynda Hardman. Towards Second and Third Generation Web-Based Multimedia. In *The Tenth International World Wide Web Conference*, pages 479–488, Hong Kong, May 1-5, 2001. IW3C2, ACM Press.

[W3C, 2001] W3C. Synchronized Multimedia Integration Language (SMIL 2.0) Specification. W3C Recommendation, August 7, 2001. Edited by Aaron Cohen.

Semantic annotation and search at the document substructure level

Dario Bonino, Fulvio Corno, Laura Farinetti
Dipartimento di Automatica e Informatica
Politecnico di Torino, Torino, Italy
{dario.bonino, fulvio.corno, laura.farinetti}@polito.it

Abstract

This paper proposes a modular architecture which separates ontology, annotations, lexical entities and search function, and offers automatic semantic annotation facilities at the document substructure level. The proposed architecture is compatible with a web services based infrastructure. Annotated resources can be XML or XHTML static or dynamic documents and need not to be stored nor modified. Preliminary results show the feasibility of the proposed approach.

1 Introduction

The promise of the Semantic Web [1] to innovate the way we design and use the web is slowly progressing, as proposed standards settle down and semantic applications are developed.
One interesting goal that can be achieved from the Semantic Web is being able to automatically synthesize a single document, as the result of a search operation, collecting and concatenating all relevant paragraphs from the available resources. This result requires development and open integration of several technologies: ontologies, semantic indexing, document substructure analysis.
Automatic extraction of semantic information and infrastructures for external annotation storage will allow for a quick and low-cost semantic encapsulation of existing resources. Many web sites will be indexed by a single annotation service, which will also offer semantic search capabilities over the entire collection.
In a wide scale deployment of Semantic Web technologies, a second problem would arise, related to the heterogeneity of content structure: the scale difference in document size and structure is in fact an important parameter in the annotation process, otherwise subsequent search operations would not be able to properly rank relevant results.
Since the problem of annotating web resources is central to the Semantic web development, many researchers are dedicating time and efforts to find good solutions for making this possible, in the easiest and user-friendliest possible way.
Systems such as Yawas [2] and Annotea [3] allow to create and share annotations attached to web documents. These annotations are stored separately from the documents, and in case of Annotea, they refer to the whole document or to selected parts of it. However, these systems do not use a structured ontology for the metadata associated to annotations.

2 Proposed Architecture

Our approach aims at covering all aspects mentioned in the introduction: it provides an architecture (Fig.**1**) for creating and managing annotations using previously defined ontologies, and it allows the tagging of documents at different granularity levels, ranging from the whole document to the single paragraph.
The proposed framework also includes the search functionalities able to exploit the semantic annotations for retrieving and composing new documents starting from the existing ones.
Annotations are stored in a standalone repository, independently from annotated resources which are related to annotations by means of Xpointers and URIs.
Automatic annotation is done by means of a module called Semantic Mapper which basically takes an ontology and a group of lexical entities (that we called *synset* as it is mainly composed of synonyms) as working components, and for each input resource it returns the collection of ontology concepts the resource is related to. Each concept of the given ontology is represented by a set of lexical entities that is used by a classical information retrieval technique [4] to classify resources and to identify the most reliable associations with the ontology concepts.
One of the most innovative aspect of the proposed architecture is the annotation repository, in which we introduced hierarchical relationships between annotations, obtaining a taxonomy in a first instance and an annotation ontology as foreseeable result.
The organization of annotations into a taxonomic structure allows the detection of the Level of Detail of each annotation by means of generalization relationships.

In a search task more relevant results will therefore be obtained by focalizing or widening annotation search according to the query.

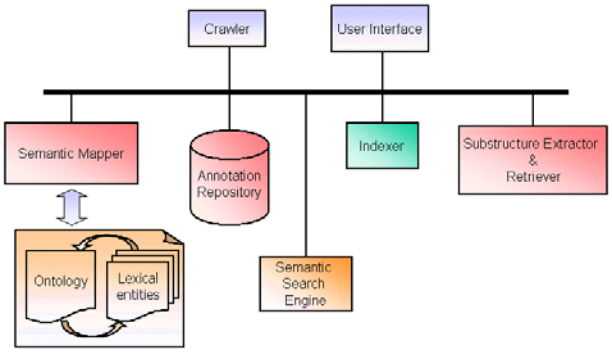


Fig. 1 Proposed architecture.

A Semantic Search module has also been designed in order to leverage the full potential of semantics. The proposed engine will use the Semantic Mapper, and will be able to translate text queries into conceptual queries. The ontology structure and in particular the relationships between concepts will offer the means for automatic search refining, while the taxonomic annotation repository will provide automatic Level of Detail detection.
Search results will be composed by many fragments coming from different web resources and will be collected into one or more pages using relevance criteria.

3 Experimental Results

A prototype of the described architecture has been implemented in Java. The Substructure Extractor and Retriever was developed using the XPath Explorer API [5] in order to extract Xpath/Xpointer constructs for the identified web resource substructures. Document supported are either XML or XHTML while HTML documents are converted into XHTML by the module using the Tidy API [6].
The Semantic Mapper has been implemented using the Jena API [7] for ontology access and navigation, and the Snowball API [8] for syntactic to semantics mapping and lexical stemming. We also developed preliminary implementations of the Annotation Repository and the Semantic Search Engine in order to set up an experiment for assessing the architecture viability.
We used an ontology on disabilities developed in collaboration with the Passepartout service of the City of Turin. It was composed of 65 concepts related each other by means of inheritance relationships, inverse relationships and some other relations. For each ontology concept a *synset* was specified using the RDF syntax (about 300 lexical entities).
Twenty-four pages were indexed, and correspondent annotations were stored in the Annotation Repository (the system generated 954 annotations in about 10s). Three queries were issued to the annotation architecture using the search engine. Human experts evaluated the relevance of retrieved fragments giving a qualitative indication of annotation effectiveness. Retrieved fragments were judged relevant in most cases showing interesting associations like the one which relates the words “Art. 1” to the concept “diritto”. This was an effective annotation because many times jurisprudence is organized in laws and laws are always subdivided in articles; in traditional search engines this result is not straightforward.

4 Conclusion

This paper presented an open architecture for building semantically enabled web services. A first prototype of the architecture has been evaluated showing the feasibility of automatic annotation of document substructures while permitting fine-grained semantic retrieval and composition of result document. Our current work is focused on improving the algorithms for the Semantic Search Engine, by including Level of Detail analysis and relevance feedback, and at deploying the distributed interface of the modules.

References

[1] Berners-Lee, T., Hendler, J., and Lassila,O. The semantic Web. Scientific American, 5/01, May 2001

[2] L.Denoue, L. Vignollet. «An annotation tool for web browsers and its applications to information retrieval, in Proc. of RIAO 2000, Paris, France, April 12-14, 2000

[3] J. Kahan, M. Koivunen, E. Prud’Hommeaux, R. Swick. Annotea: An Open RDF Infrastructure for Shared Web Annotations, in Proc. of the WWW10 International Conference, Hong Kong, May 1-5, 2001.

[4] R. Baeza-Yates, B. Ribeiro-Neto. Modern Information Retrieval, Addison-Wesley, 1999

[5] Xpath Explorer, project homepage at http://www.purpletech.com/xpe/index.jsp

[6] D. Ragget *et al.*, HTML Tidy project, project home page at http://tidy.sourceforge.net/

[7] Mc Bride, B. Jena: a semantic Web toolkit, Internet Computing, IEEE, Volume 6, Issue 6, Nov/Dec 2002, page(s) 55-59

[8] Porter MF (1980) An algorithm for suffix stripping. Program, 14: 130-137.

TRELLIS: Supporting Decision Making via Argumentation in the Semantic Web

Timothy Chklovski, Yolanda Gil, Varun Ratnakar and John Lee
USC Information Sciences Institute
4676 Admiralty Way,
Marina del Rey, CA, USA
{timc, gil, varunr, johnlee}@isi.edu

Abstract

Decision making is a fundamental human activity. Most important decisions require careful analysis of the factors influencing a decision. Surprisingly, there has been little work on tools to capture and assess validity of a heterogeneous set of facts and claims that bear on a decision.

Good decision making requires two components which are specializations of Semantic Web approaches: (i) sound argumentation about the factors involved and (ii) clear judgments about reliability of the information sources in which the argument is grounded. We describe TRELLIS, our vehicle for researching the problem and a tool supports making decisions that, as is often the case, must rest on possibly conflicting or unreliable information sources.

We report on recent progress in collecting and classifying argumentation acts which occur in real arguments, and outline our ongoing work on extending how argumentation and decision making over heterogeneous sources can be supported. The system is available at http://trellis.semanticweb.org

Keywords: argumentation, decision making, heterogeneous information, Semantic Web

1 The Need for Argumentation Tools

Much of what companies and knowledge workers engage in is decision making. For each such crucial decision, there may be dozens or hundreds of information sources requiring careful analysis of the factors involved.

Surprisingly, much of the work on supporting decision making has focused on helping to process numerical data (decision support systems), largely ignoring the central problem of tools to capture and assess validity of a heterogeneous set of facts and claims that bear on a decision.

Additionally, while there are tools such as spreadsheets (e.g. Excel) for exchanging numerical models, tools for diagram manipulation (e.g. Visio) and even comprehensive environments for scientific computation (such as Mathematica), there are not equivalent tools for argumentation. Because of this lack of tools, the common task of capturing arguments that support important decisions remains mainly a word-processor based activity. To an organization or individual seeking to track its reasons for making certain decisions (and learn from experience), there is currently no support for (i) locating relevant documents, (ii) browsing assertions about trustworthiness of sources used in a decision, and (iii) storing and retrieving arguments in structured form, which would allow for re-use of relevant parts of arguments. All this functionality, while requiring specific tools, are enabled by the sort of markup and protocols that is broadly envisioned by the Semantic Web.

Good decision making requires clear statement of the factors involved and explicit declaration of which factors outweigh other factors, which arguments can be dismissed, and so on. Therefore, clear, sound argumentation and explicit judgments about validity of sources form the basis of good decision making.

2 Trellis: Supporting Argumentation Grounded in Sources

TRELLIS [Gil and Ratnakar, 2002] allows users to add their observations, viewpoints, and conclusions as they analyze information by making semantic annotations to documents and other on-line resources. Users can associate specific claims with particular locations in documents used as "sources" for analysis, and then structure these statements into an argument detailing pros and cons on a certain issue. An illustrative example is given in Figure 1 and described in greater detail after the discussion of the role of Semantic Web. Other researchers are also looking at representing argumentation; in particular, see [Shum, Motta, and Dominigue, 2000] for a tool supporting argumentation in the domain of scholarly discourse.

Because evidence is often incomplete and/or biased (consider, e.g., most marketing literature used in making purchasing decisions), TRELLIS includes specific tools for indicating trustworthiness of a source with respect to a particular purpose.

The TRELLIS project contributes to the Semantic Web effort in the following ways:

- *Semantic Markup of Arguments*. Rather than handle arguments in fully textual form, TRELLIS supports construction of argument trees which can be searched, imported, and otherwise processed by both machines and humans.
- *Rating of Information Sources.* Trellis collects reusable semantic markup (reliability and trustworthiness for a given context) of documents from users.

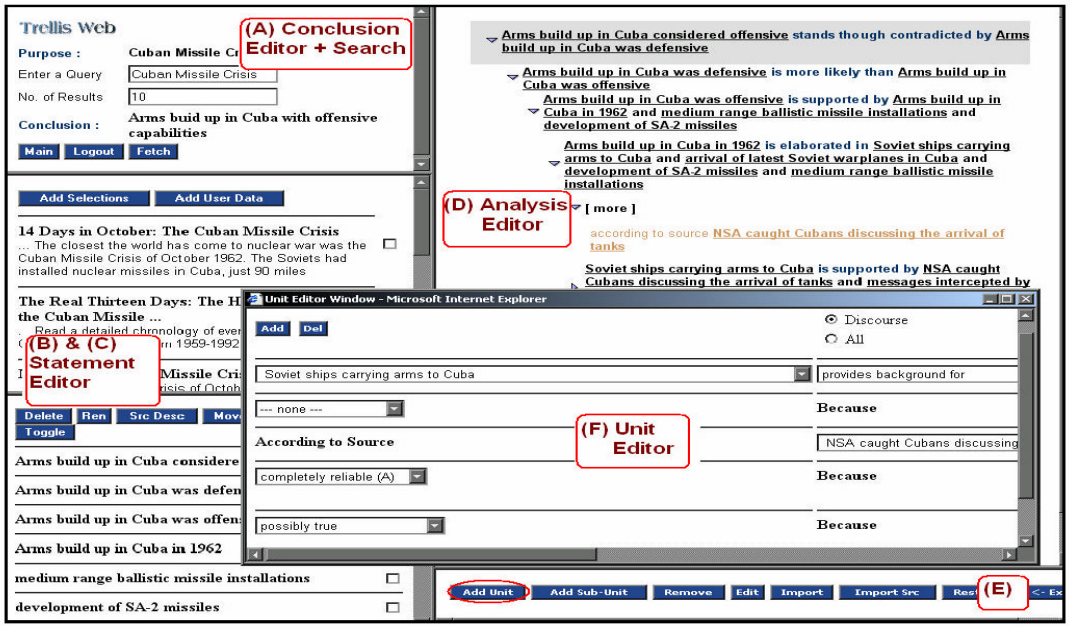


Figure 1. A Snapshot of the TRELLIS user interface. From top-left counter clock wise the system shows: purpose and conclusions of the analysis (A), original documents and associated statements (B&C), units of the analysis (E,F), and overall analysis (D).

- *Easy adoption path.* Users of TRELLIS are allowed to mix arbitrary natural language with structuring clauses.

Figure 1 shows the TRELLIS user interface. The example analyses the Cuban missile crisis, a thoroughly studied case of political decision making.

The purpose of the analysis and the final conclusion are shown in Frame (A). Analysis and opinions revolve around facts, statements, and hypotheses. With Frame (B), users can search the Web for relevant to an analysis documents, or can add their own documents.

Each resource is then associated with a short statement entered by the user in Frame (C). Users can specify several statements per resource, each summarizing a salient piece of information described within the resource in terms that are suitable to the user. Frame (E) invokes the Unit Editor (F). The overall analysis is composed using the Analysis Editor, shown in Frame (D).

TRELLIS can export user's analysis in several markup languages (plain XML, RDF, and DAML+OIL).

3 Recent Results and Ongoing Work

Recently, we have conducted an analysis of the kinds of support and objections used in approximately 30 real arguments constructed by TRELLIS users. The arguments spanned a variety of topics, from political and military decisions, to merits of a given operating system, to legality of abortions, to selecting a cat or a dog as a pet.

The analysis revealed that *comparisons*, in some form, are nearly universal to all arguments (a comparison is a statement such as "ABC's laptops are more reliable than laptops made by XYZ"). Furthermore, comparisons can be broken down into a number of well-defined types, a classification which we created based on the examples we had and additional research.

Comparisons classify by whether they are comparing things or actions. When comparing things, the comparison may be on an explicitly stated criterion ("cats are better than dogs") or via their action ("most cats eat less than most dogs"). Actions can be compared by a criterion ("jogging is better for your health than sitting"), and by their purposes – for example, "the best way to get regular exercise is to get a dog" encodes that a certain action A is best for some P.

Because of their observed importance to argumentation, we are currently focusing on extending supporting comparisons, creating tools that will automatically recognize a comparison and identify thematic roles in a given comparison statement, similarly to the roles introduced in and manually tagged in the FrameNet project.

Such markup of comparisons should allow us to extract which dimensions of comparison are applicable to a given entity, as well as retrieve additional dimensions of comparison pertinent to the current decision in a case-based fashion.

Additionally, we are refining the mechanisms for statement entry; the upcoming version of the tool will allow for multi-level, incremental breakdown of a text statement into more structured form, letting the user experience incremental payoff from structuring her argument.

References

[Gil and Ratnakar, 2002] Yolanda Gil and Varun Ratnakar. Trusting information sources one citizen at a time. *Proceedings of the First International Semantic Web Conference (ISWC)*, June 2002.

[Shum, Motta, and Dominigue, 2000] Simon B. Shum, Enrico Motta, John Dominigue. ScholOnto: An Ontology-Based Digital Library Server for Research Documents and Discourse. *International Journal on Digital Libraries,* 3 (3), Aug/Sept 2000.

Integrating Directories and Service Composition

Ion Constantinescu and **Boi Faltings**
Artificial Intelligence Laboratory
Swiss Federal Institute of Technology
IN (Ecublens), CH-1015 Lausanne (Switzerland)
{ion.constantinescu, boi.faltings}@epfl.ch
http://liawww.epfl.ch

1 Introduction

While the current WWW is most commonly accessed through a browser, the future semantic web will be very often accessed through web services. This will require automatic techniques for finding and then composing these services to achieve the desired functionality.

2 Discovery and Matchmaking of Web Services

A possible method for discovering Web Services is matchmaking. In this case the directory query (requested capabilities) is formulated in the form of a service description template that presents all the features of interest. This template is then compared with all the entries in the directory and the "matching" results are returned. In the example below we well consider how the match relation is determined between the query service Q and the library service S, that have only one ouput defining the provided style of music.

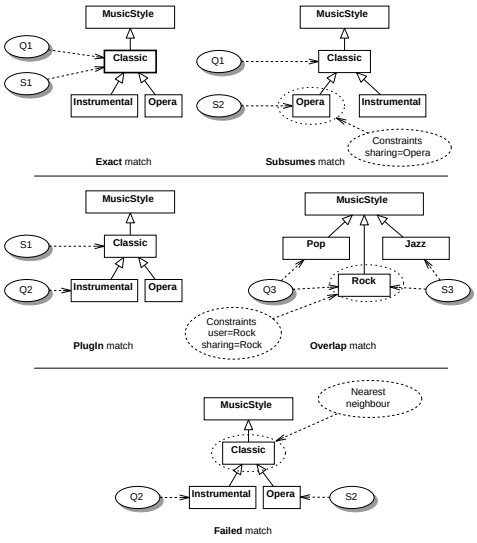


Figure 1: Match types of one output of query and library services Q and S by "precision": **Exact**, **PlugIn**, **Subsumes**, **Overlap**.

3 Efficient Service Directories

The novelty of our approach is to consider a service description as a multidimensional data record and then use in the directory techniques related to the indexing of such kind of information. This approach leads to local response times in the order of milliseconds for directories containing tens of thousands (10^4) of service descriptions.

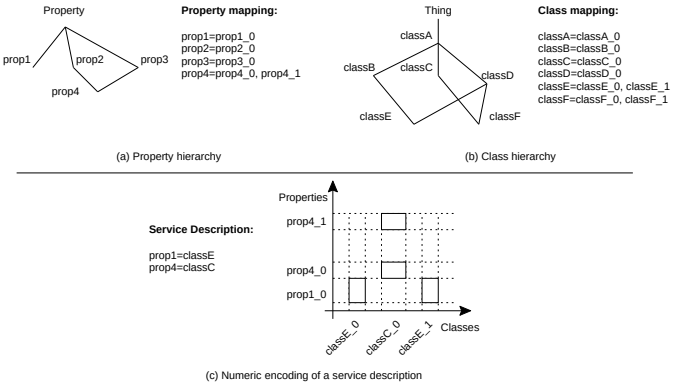


Figure 2: Numeric encoding of a service description

Taxonomies can be numerically encoded such that inclusion relations can be determined by very simple operations [3]. Our approach is to use an interval based representation for both classes and properties. The method is generalised in order to support multiple parents by allowing for the encoding of a class/property as a set of intervals instead of only a single interval. The numeric encoding of a service description is straightforward: the pairing of properties represented as sets of intervals with classes or values also represented as sets of intervals can be seen as a set of rectangles in a bidimensional space having on one axis Classes or Values and on the other Properties.

4 Service composition with directories

In this paper we analise a class of algorithms for building integrated services that incrementally extend an initial set of propositions until the set satisfies the initial integration query.

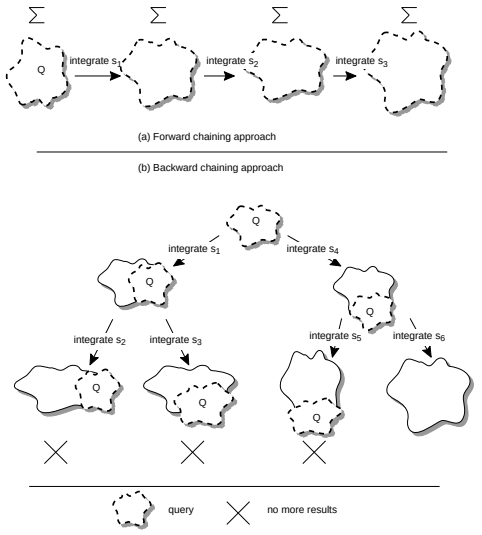


Figure 3: Two approaches to service integration: **forward chaining** (two algorithms) and **backward chaining**.

5 Service composition testbed and experimental results

For testing we have considered a model generated in a non-deterministic manner. As the main parameter of the model we have used the number of services defined over a *maximum services size* of propositions from the vocabulary of *vocabulary size*.

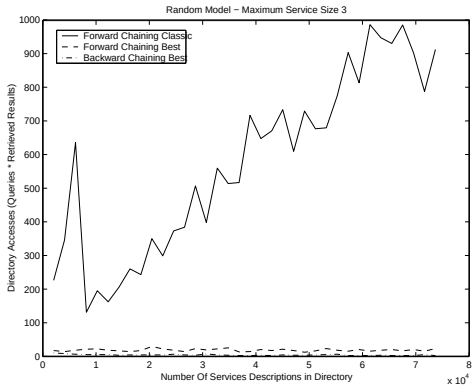


Figure 4: Random test model: forward chaining classic, forward chaining best match and backward chaining best match.

The results show that both the *forwardChainingBestMatch* and the *backwardChaining* algorithms make better use of the directory and outperform the classic *forwardChaining* algorithm while also being more scalable. *forwardChainingBestMatch* and *backwardChaining* have comparable performance which suggest that the decision of choosing one in favor of the other might have to be application dependent.

6 Conclusion

Web services will likely be a major application of semantic web technologies. Automatically activating web services requires solving problems of indexing and automatic service composition. We have presented approaches to both problems.

In conclusion integrating composition planning with a directory is important to achieve scalability, and we have shown an approach to do this that appears to be practical.

References

[1] A. Ankolekar. DAML-S: Web Service Description for the Semantic Web, 2002.

[2] R. J. Bayardo, Jr., W. Bohrer, R. Brice, A. Cichocki, J. Fowler, A. Helal, V. Kashyap, T. Ksiezyk, G. Martin, M. Nodine, M. Rashid, M. Rusinkiewicz, R. Shea, C. Unnikrishnan, A. Unruh, and D. Woelk. InfoSleuth: Agent-based semantic integration of information in open and dynamic environments. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, volume 26,2, pages 195–206, New York, 13–15 1997. ACM Press.

[3] Ion Constantinescu and Boi Faltings. Efficient matchmaking and directory services. Technical Report No IC/2002/77, Artificial Intelligence Laboratory, Swiss Federal Institute of Technology, 2002.

[4] AnHai Doan and Alon Y. Halevy. Efficiently ordering query plans for data integration. In *ICDE*, 2002.

[5] Joseph M. Hellerstein, Jeffrey F. Naughton, and Avi Pfeffer. Generalized search trees for database systems. In Umeshwar Dayal, Peter M. D. Gray, and Shojiro Nishio, editors, *Proc. 21st Int. Conf. Very Large Data Bases, VLDB*, pages 562–573. Morgan Kaufmann, 11–15 1995.

[6] Craig A. Knoblock, Steven Minton, Jose Luis Ambite, Naveen Ashish, Ion Muslea, Andrew Philpot, and Sheila Tejada. The Ariadne Approach to Web-Based Information Integration. *International Journal of Cooperative Information Systems*, 10(1-2):145–169, 2001.

[7] S. McIlraith, T.C. Son, and H. Zeng. Mobilizing the semantic web with daml-enabled web services. In *Proc. Second IntÍ Workshop Semantic Web (SemWebÓ2001)*, Hongkong, China, May 2001.

[8] Massimo Paolucci, Takahiro Kawamura, Terry R. Payne, and Katia Sycara. Semantic matching of web services capabilities. In *Proceedings of the 1st International Semantic Web Conference (ISWC)*, 2002.

[9] K. Sycara, J. Lu, M. Klusch, and S. Widoff. Matchmaking among heterogeneous agents on the internet. In *Proceedings of the 1999 AAAI Spring Symposium on Intelligent Agents in Cyberspace*, Stanford University, USA, March 1999.

[10] S. Thakkar, C. A. Knoblock, J. L Ambite, and C. Shahabi. Dynamically composing web services from online sources. In *Proceedings of AAAI-02 Workshop on Intelligent Service Integration*, July 2002.

Towards a Semantic Enterprise Information Portal

Emanuele Della Valle and **Paolo Castagna** and **Maurizio Brioschi**
CEFRIEL - Politecnico of Milano
Via Fucini, 2 - 20133 Milano - Italy
{dellava, castagna, brioschi}@cefriel.it

1 Introduction

Knowing what you know is becoming a real problem for many enterprises. Their intranets are full of shared information, their extranet support a flow of data both with suppliers and customers, but they have lost the integrated view of their information. Thus finding information for decision taking is every day harder. A comprehensive solution to this problem should provide at least an answer to the following questions: What information do we have? Where is it? How did it get there? How do I get it? How can I add more? What does it mean?

Portals, in particular Enterprise Information Portals (EIPs), some years ago have been brought into the limelight for their ability to address these questions by giving a unique and structured view of the available resources. However EIPs cannot be considered a final solution, because they do help people in managing the information, but they still require a huge amount of manual work. So, we believe that using state-of-the-art web technologies will not be sufficient in the immediate future, since the lack of formal semantics will make it extremely difficult to make the best use (either manually or automatically) of the massive amount of stored information and available services.

2 The concept

An ontology-oriented metadata-based solution

Metadata-based solutions provide enough *machine-processable* information for automating most information retrieval tasks, but, in a pure metadata based solution, the meaning associated to the metadata is not machine-processable. So a machine can process this metadata but it cannot "reason" upon it. A good deal of help can come from defining metadata using ontologies. In fact, ontologies, being explicit (hence formal) conceptualisations of a shared understanding of a domain can be used to make metadata machine processable. Only some years ago, it was the time for academics to experiment with such ideas, but today metadata-based ontology-oriented solutions are becoming feasible thanks to the ongoing Semantic Web researches. Therefore, soon enterprises would be able to build "corporate Semantic Web" represented by services and documents annotated with metadata defined by a corporate ontology. Thus they will need to update their EIPs in order to cope with ontologies and metadata. They will need a *Semantic EIPs*.

The idea

The innovative idea, first proposed by [Maedche *et al.*, 2001], is straightforward: can we use metadata defined by ontologies to support the construction of portals? And if so, does it help? Even if it might appear as a radical new departure actually it is not. On the contrary it is the bringing together of existing and well understood technologies: *Web Frameworks* (as Struts, Jetspeed, etc.) that implement Model-View-Controller design pattern, *WWW conceptual models* (as WebML [Ceri *et al.*, 2000]) that are proposals for the conceptual specification (using extended E-R models) and automatic implementation of Web sites, *Ontologies* to model the domain information space, the navigation, the access and the presentation, and *Metadata* to make resource descriptions available to machine in a processable way.

The approach

Concerning modeling, we have decided to follow an approach similar to those adopted in WWW conceptual modeling. We model separately the domain information space, the navigation and the access. The *domain information model* (in this case the corporate ontology) is a shared understanding of the information present in the corporate semantic web (hence a unique model) that doesn't change, or changes slowly, over the time. Moreover, its design is completely decoupled from the semantic EIP design. Therefore the semantic EIP cannot assume any "a priori" agreement except the use of a common set of primitives (e.g. OWL). However, if we want to access the corporate semantic web using a semantic EIP we need to define at least some *upper terminology*, known by the semantic EIP, that can be employed in defining both the navigation and the access model. The *navigation models* represent the heterogeneous paths the EIP users can adopt in traversing the corporate semantic web. They are not necessarily shared among users, but they are jointly employed by homogeneous categories of users. Navigation models should be built by *mapping* the corporate ontology terminology to the navigation upper terminology. Finally, the *access models* represents collections of resources not strictly homogeneous, highly variable and sometimes even related to a specific user, a sort of *views*. Also access models can be built via *mapping*, but they might require to explicitly draw some new relationships and, sometimes, also to add ad-hoc resources.

Concerning presentation, instead of modeling it, we have decided to use Model-View-Controller pattern, because we don't expect good graphic designer to be good modelers and vice versa. This way we aspect the same advantages, in term of visual coherence and accessibility, as modeling but at a more affordable effort.

Furthermore, we recognize in a metadata-bases ontology-oriented solution a major progress in interoperability. So, our approach, resigning any "a priori" agreement on the corporate ontology, enables a distributed environment where autonomous entities maintain heterogeneous shared resources, describing them with metadata defined by the corporate ontology.

Using ontologies at authoring time

At authoring time ontologies, in particular the corporate ontology, can be exploited in supporting the editorial task. It has already been shown that they can be employed in automating part of process for creating editorial interfaces. But we believe most of the benefits should come from reducing the effort required to augment resources with metadata. In the authoring environment we envision, authors are asked only what is strictly necessary, while the rest is inferred.

Using ontologies at browsing time

Web users interact with the Web in many ways, but two patterns are commonly recognized: searching and navigation. A semantic EIP should exploit metadata and ontologies in order to improve both interaction patterns. In particular we want to improve searching by resource discovering and navigation by automatic link creation. On the one hand, once an enterprise has got a corporate semantic web, search won't be exclusively based on full text search, but it could make lever on semantics, so it could "analyse" the resources finding those that match the user request. Thus it is no more a matter of searching but it becomes a matter of discovery by matching. On the other hand, when a user has retrieved a resource, he/she needs help in navigating to other related resources. So our idea is to *insert* the retrieved resource in a *navigation panel* that contains automatically generated links to the related resources.

In particular, we propose to place in the navigation panel of a semantic EIP three different kinds of links: *Access point links* that render, using one of the access models, a sort of views to guide the user in accessing the information, *categorized links* that render, using one of the navigation models, a set of boxes populated with links that are the result of a simple property-based query over the metadata describing the retrieved resource, *metadata links* that provide an intuitive navigation from and to the retrieved resource following the metadata used to describe it.

Related works

The approach that shows more similarities with ours is COHSE [Carr *et al.*, 2001]. Its main concern is in linkage and navigation aspects between web pages, but it doesn't model explicitly *views* using navigation and access models. Another similar approach is SEAL [Maedche *et al.*, 2001] and its recent evolution SEAL-II, but they both uses pre-semantic web technologies.

3 An early proof of concept

In order to proof this concept, we have built a first prototype of a semantic EIP following the presented approach (an on-line demo is available at `http://seip.cefriel.it`). We choose not to address authoring time issues but to concentrate instead on browsing time and in particular to automatic link creation. We have developed a servlet-based application that uses Velocity for implementing the model-view-controller pattern and RACER [Haarslev and Moller, 2001] as reasoner. It only "knows" some properties (a first draft of the introduced navigation and access upper terminology) but if a user inserts an ontology and maps its properties to these terminology the prototype is able to guide him through the resources, he eventually describes using such ontology.

4 Conclusion

The described approach for semantic EIPs brings many innovation in EIP development. It imposes no restriction but the use of RDF, RDF Schema and OWL in building the corporate ontology. It doesn't require the information carried by the metadata to be coded in any particular way, thus this information is reusable. It enables both resources and metadata management in a distributed and autonomous way as long as resources are network retrievable. Yet, it offers a homogeneous navigation experience over a corporate semantic web through mapping of corporate terminology to the portal terminology.

So, a semantic EIP, built using the proposed approach, will give a unified view of the information present in the corporate semantic web, while the enterprise can keep developing distributed and autonomous systems on an ad-hoc basis and singular enterprise departments can keep their degree of autonomy in managing such systems.

Acknowledgements

We thank our student Lara Marinelli and we report that the implementation of the prototype has been partially founded by Engineering as part of CEFRIEL XV Master IT

References

[Carr *et al.*, 2001] Les Carr, Wendy Hall, Sean Bechhofer, and Carole A. Goble. Conceptual linking: ontology-based open hypermedia. In *World Wide Web*, pages 334–342, 2001.

[Ceri *et al.*, 2000] Stefano Ceri, Piero Fraternali, and Aldo Bongio. Web Modeling Language (WebML): a modeling language for designing Web sites. *Computer Networks (Amsterdam, Netherlands: 1999)*, 33(1–6):137–157, 2000.

[Haarslev and Moller, 2001] Volker Haarslev and Ralf Moller. High performance reasoning with very large knowledge bases: A practical case study. In *IJCAI*, pages 161–168, 2001.

[Maedche *et al.*, 2001] Alexander Maedche, Steffen Staab, Nenad Stojanovic, Rudi Studer, and York Sure. SEAL – A framework for developing SEmantic Web PortALs. *Lecture Notes in Computer Science*, 2097:1–7, 2001.

Computational Ontologies and XML Schemas for the Web

Pradnya Dharia and **Anvith R. Baddam** and **R. M. Malyankar**
Department of Computer Science and Engineering
Arizona State University
Tempe, AZ 85287-5406, USA.
rmm@acm.org, {pradnya.dharia,anvith}@asu.edu

1 Introduction

This poster describes work on mapping formal ontological engineering products to XML schemas without resort to RDF or DAML (a "SemWeb-Lite", so to speak). Much of the general problem of mapping from ontological representations to some form of XML syntax (specialized as necessary to allow knowledge structures that allow inclusion of knowledge representation constructs) is already part of RDF and DAML+OIL, and instead of repeating these well-known ideas this paper attempts to focus on the specialized form of the general problem with the additional constraints of requiring the use of XML schema while allowing the application developers to work, albeit at a lower level of intelligent processing, without a knowledge of knowledge representation formalisms, DAML, or OWL.

2 Translations and Mappings

The mapping of domain ontology to schemas is guided by the rule that every class and attribute in the ontology must be represented in the XML schema and 'document objects' (as compared to domain entities) should also be represented in the source ontology (or ontologies), i.e., items such as a 'report form' which contains items such as names, dates, etc., should also be represented in one of the source ontologies.

2.1 Ontological Information

The main features of our approach to mapping and translation from ontologies to XML schemas are as follows:

1. Classes in the ontology are mapped into named "complex types" in the format of the W3C XML Schema specification slots in the ontology are generally turned into XML "simple types", which are (at the user's option) either attributes for the complex types corresponding to the classes, or sub-elements in those same complex types.
2. Range restrictions on the values of slots in the ontology are preserved for integers, floats, and symbol value types; these are converted into XML schema "restriction"s in the generated XML schemas. Numeric restrictions are converted into maximum/minimum XML schema restrictions, and enumerated ranges are preserved as enumerations.
3. Class inheritance relationships are largely preserved, by being converted into "extension" relationships between types in the XML schema specification. This seems to work fairly well in practice though it is not a full implementation of inheritance.
4. Metadata for classes and attributes that may be available within the ontology as documentation for classes or slots is preserved by being placed into the "annotation" elements allowed by the XML schema specification.

2.2 Structural Information

Two approaches were tried to represent structural information in document schemas. The first consisted of creating **ELEMENT** meta-classes with asociated additional information relating to XML structures (such as **CONTAINS**, **CHOICE** and **SEQUENCE**), then using the meta-classes for selected domain concepts (those desired to be associated with elements in the target document schema being designed), thus associating information that can be used to generate XML elements in the application schema. An alternative approach consists of reifying XML-schema relationships into a distinct 'XML-schema' ontology (to get 'XMLS-relation' classes such as *Element*, *complexType*, *simpleType*, etc.) and using a selected subset of such 'XMLS-classes' to create reified relations connecting instances of domain classes with instances of XMLS-relations. In short, the domain class is linked with a unique reified binary relation class which is also associated with the XMLS-class *xs:Element* (i.e, this represents a binary relation between the *xs:Element* and the corresponding domain class). This approach is more difficult for lay ontologists to understand and needs more steps during schema design but is more sound from a knowledge representation point of view and more complete than the first since it captures the logical interrelationships between domain ontologies and concepts in XML.

2.3 Products

Three levels of schemas are produced:

1. Two levels of *type library* schemas, one containing XML types obtained from type information in the domain ontology, and the second corresponding to classes in the domain ontology and complex types in XML. The types corresponding to slots and classes are global types and

have names which can be used to reference them in derived schemas.

2. An *application* schema with elements derived from types in the type library, and with element structure as entered by the schema designer.

Note that since tags are uniquely associated with objects (and attributes) in a separate ontology, the RDF advantage of identifying names with resources is retained. It is still possible to use different tags for the same concepts and still maintain interoperability between different XML schemas derived from the same base ontology.

3 Limitations

The most significant difference between ontologies and the XML schema syntax specification that leads to a significant limitation in any effort to generate XML schemas from ontologies is the lack of multiple inheritance in XML Schema. In ontologies, on the other hand, a class can be a subclass of more than one class; this cannot be directly captured in XML schemas. Other, more minor limitations arise from the restricted number of primitive "types" in ontological engineering tools in general and Protégé in particular and non-extensibility of the Protégé built-in type system; XML Schema, on the other hand, has a richer set of builtin types and allows user-defined types. This means that specifying XML types in Protégé (to be used in generating the schemas) requires a rather complex workaround in the schema generator and special additions to the ontology (basically, specification of XML builtin and derived types that is separate from the Protégé type system).

4 Discussion

In effect this reproduces what DAML+OIL does — broadly speaking, both provide an ontological backing for XML markup. DAML does this through the DAML-ont formalism while the process described in this paper allows the retention of other ontological formalisms separately from the document schemas, which are pure XML. Our implementation uses the Protégé knowledge base editor, but could easily be adapted to other representations. The "online" DAML ontology used in DAML is replaced by "off-line" ontological knowledge (which could still be made accessible if needed by a program, but since the XML schema is available, the ontological knowledge need not be transferred unless it is needed). The XML programmer is relieved from the need to learn a new formalism and new software tools but a transition path to newer technology such as OWL is kept open even for applications originally written to use ordinary XML.

As mentioned earlier, any application domain almost certainly has many different XML schemas for different applications in use. It is possible to extract ontological knowledge from these schemas (an active area of research in knowledge acquisition and ontological engineering). Combined with an approach like that described here, it becomes possible to assure interoperability for documents that originally used (and which may continue to use) different XML tags for the same information. In addition, tools such as this will contribute the glue for semantic web applications created at different levels of maturity of semantic web technology — for example by allowing a link between legacy XML applications and newer OWL-aware applications.

5 Related Work

An XML backend to Protégé, a tool for creating and editing ontologies and knowledge bases, [Grosso *et al.*, 1999] is available, and a DAML plugin is under development[1]. The focus is on converting Protégé ontologies to XML or DAML syntax respectively. The XML backend focuses on saving an ontology itself as an XML file, not on generating an XML schema for application programmers in the domain. There is another plugin, the *XML Tab* plugin, which also stores the ontology itself as an XML file (in a different format from the first), which could possibly be adapted to subsequently developing a schema separately, but imposes severe limits on what can be done in schema development, e.g., by placing subclasses as contained elements in their superclasses. No document schema for an application domain can be directly developed or created. At this point of time, the DAML plugin does not convert Protégé ontologies from other formats to DAML but only reads DAML+OIL ontologies and allows only those ontologies to be manipulated and saved[2] Additionally, there exist DAML editors similar to the system described here which store ontological information in frame-based representations and generate DAML output in RDF/XML syntax. Klein, et al. [Klein *et al.*, 2000] describe a translation of OIL specifications to XML schema. That translation is similar to the part of our translation which deals with ontology classes and slots (though not identical, it does not differ in any significant manner), but does not deal with the question of document structure in any detail — the question of document structure is dealt with only in passing, in terms of a statement about defining "a grammar for entity, associat[ing] basic datatypes with built-in datatypes, add lexical constraints if desired".

Acknowledgments

This work was partially supported by the National Science Foundation under grant EIA-9983267 and by the U. S. Coast guard and Sun Microsystems.

References

[Grosso *et al.*, 1999] W. E. Grosso, H. Eriksson, R. W. Ferguson, J. H. Gennari, S. W. Tu, and M. A. Musen. Knowledge modeling at the millennium: The design and evolution of Protégé-2000. In *Twelfth Banff Workshop on Knowledge Acquisition, Modeling and Management, Banff, Alberta*, 1999.

[Klein *et al.*, 2000] M. Klein, D. Fensel, F. van Harmelen, and I. Horrocks. The relation between ontologies and schema-languages: Translating OIL-specifications in XML-schema, 2000. http://delicias.dia.fi.upm.es/WORKSHOP/ECAI00/7.pdf.

[1] Described on the Protege web site at http://protege.stanford.edu

[2] Note at http://www.ai.sri.com/daml/DAML+OIL-plugin/.

Ontology Translation: Available Today *

Dejing Dou, Drew McDermott, and Peishen Qi
Yale Computer Science Department
New Haven, CT 06520, USA
{dejing.dou,drew.mcdermott,peishen.qi}@yale.edu

1 Introduction

One major goal of the Semantic Web is to get web-based agents to process and "understand" data rather than merely display them as at present [Berners-Lee *et al.*, 2001]. Ontologies, which are defined as the formal specification of a vocabulary of concepts and axioms relating them, are seen as playing a key role in describing the "semantics" of the data. As more and more ontologies are developed, the problem arises of communicating between agents that use different vocabularies. Any message or query sent from agent A to agent B must be translated from A's notation to B's. We call this process *ontology translation*.

We assume that translation can be modeled as a first-order deductive process. An attractive special case of deduction is that performed by description-logic (DL) systems. For many such systems, some inference problems are decidable. We must reject this alternative for a couple of reasons. One is that the wholesale translation of data from one notation to another requires forward chaining, whereas DL's are oriented around answering queries [Baader *et al.*, 2003]. Another is that different ontologies can embody fundamentally different analyses of a domain, especially if their foci are different. One may draw many and subtle distinctions where the other makes do with very superficial classifications. The axioms linking them together go beyond what DLs can express.

Our strategy, therefore, has been to use a first-order theorem prover, with forward and backward chaining plus equality substitution. Before going into details, we should make sure that the ontology translation problem is distinguished from two closely related problems.

1. Syntactic translation: There are a wide variety of data formats used to express information. Many are based on XML. Almost all can be translated into first-order logic, and can be generated from a first-order equivalent. We assume these processes are already automated. For instance, we provide front- and back-end translators to translate our internal notation to RDF.
2. Ontology mapping: Before translation is possible, the ontologies involved must be *merged*, yielding a *merged ontology* that captures the relationships between them. It is a reasonable conjecture that automated tools can help in the merging process by finding plausible links between symbols in the two ontologies to be merged. A set of such links is called a *mapping*. Mappings can be generated by looking for similarities of names of symbols, and of topological relationships among them. Our focus is on what happens *after* the merged ontology is built, not on building it.

Our system is called *OntoMerge*. The *merge* of two related ontologies is obtained by taking the union of the terms and the axioms defining them, using XML namespaces to avoid name clashes. We then add *bridging axioms* that relate the concepts in one ontology to the concepts in the other through the terms in the merge. Devising and maintaining a merged ontology must be done by human experts, both domain specialists and "knowledge engineers." Once the merged ontology is built, ontology translation can proceed without further human intervention. The inference mechanism we use, a theorem prover optimized for the ontology-translation task, is called *OntoEngine* [Dou *et al.*, 2002]. We use it for dataset translation, query handling, and other tasks space precludes us from describing.

Our internal representation language is *Web-PDDL* [McDermott and Dou, 2002], a strongly typed first-order logic language with Lisp-like syntax. It extends the Planning Domain Definition Language (PDDL) [McDermott, 1998] with XML namespaces and more flexible notations for axioms. Web-PDDL can be used to represent ontologies, datasets and queries. Figure 1 shows an example, part of the merged ontology (or "domain," to use the PDDL term) that links two genealogy ontologies, one produced by DRC and one by BBN. Note that even for a topic as simple as roles in a marriage, and *even though both are based on the GEDCOM genealogy notation, a widely accepted standard*, the two component domains reflect different design decisions. The DRC ontology has separate predicates `husband` and `wife`, and the BBN version has one predicate `spouseIn` plus a specification of a person's gender. The axiom shown is one of those required in order to relate the two. (The "`@`" notation is for namespace prefixes.) In our experience, most axioms are simpler than this, and could easily be expressed in a DL. However, there are almost always a substantial set of bridging axioms that are either impossible to express in DL terms, or expressible only by contortions that result in obscure, bug-prone formalizations.

*This research was supported by the DARPA/DAML program.

```
(define (domain drc_bbn_gen_merging)
   (:extends (uri "http://orlando.drc.com/daml/Ontology/Genealogy
                  /3.1/Gentology-ont.daml"
              :prefix drc_ged)
             (uri "http://www.daml.org/2001/01/gedcom/gedcom.daml"
              :prefix bbn_ged))

   (:types  Individual - Person
            Male Female - Individual ...)

   (:facts
      (forall (f - Family w - Individual m - Marriage)
           (if (and (@bbn_ged:sex w "F")
                    (@bbn_ged:spouseIn w f)
                    (@bbn_ged:marriage f m))
               (wife f w))) ...))
```

Figure 1: Fragment of a Merged Ontology

The problem of translating datasets can be expressed abstractly thus: given a set of facts in one vocabulary (the *source*), infer the largest possible set of consequences in another (the *target*). We break this process into two phases:

1. *Inference:* working in the merged ontology, draw inferences from source facts.
2. *Projection:* Retain conclusions that are expressed purely in the target vocabulary.

In an experiment with a genealogy dataset containing 21164 facts using the BBN ontology (concerning the pedigrees of European royalty for several centuries), OntoEngine was able to generate an equivalent dataset in the DRC ontology containing 26956 facts. The time taken was 59 seconds on a Pentium III workstation running at 800 MHz, with 256Mbytes of RAM. In another experiment involving geographic ontologies, 4611 input facts were translated into 4014 output facts in 18 seconds. OntoEngine is written in Java, and could be considerably faster if converted to Lisp or C++, but our current throughput of hundreds of output facts per second is quite adequate for the small-to-middle-sized datasets we expect in semantic-web applications. For larger datasets one would rethink the translation task in terms of backward chaining, in which queries are translated, not the datasets used to answer them. Obviously, for both forward and backward chaining, the timings one can expect for a given domain are dependent on its axioms, and the undecidability of first-order inference means that there exist domains for which this whole approach will fail completely. Our experience, however, is that ontology-translation tasks do not require intricate theorem proving with its attendant combinatorial explosions.

Prospective users should check out the OntoMerge website:

`http://cs-www.cs.yale.edu/homes/dvm/daml/ontology-translation.html`

We have put all URLs of existing merged ontologies there. The Ontomerge service is designed to solicit descriptions of ontology-translation problems, even when OntoMerge can't solve them, thereby letting us know of real-world problems in this area. In return for user input about a translation problem, the OntoMerge staff will undertake to produce a merged ontology that solves the problem. We are also working on automated tools that will allow domain experts to generate merged ontologies with less intervention from OntoMerge experts.

To summarize: Without waiting for the existence of perfect ontology-mapping tools, we have produced the world's first ontology-translation service on the World Wide Web. It serves as a demonstration that first-order inference is a viable technique for doing ontology translation between web agents. It is also ready to perform as a web service itself, acting as an intermediary between agents speaking different languages.

References

[Baader *et al.*, 2003] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider. *The Description Logic Handbook; Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[Berners-Lee *et al.*, 2001] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American* , 284(5), 2001.

[Dou *et al.*, 2002] Dejing Dou, Drew McDermott, and Peishen Qi. Ontology translation by ontology merging and automated reasoning. In *Proc. EKAW Workshop on Ontologies for Multi-Agent Systems*, 2002.

[McDermott and Dou, 2002] Drew McDermott and Dejing Dou. Representing Disjunction and Quantifiers in Rdf Embedding logic in Daml/rdf. In *Proc. Int'l Semantic Web Conference*, 2002. Available at `http://www.cs.yale.edu/~dvm`.

[McDermott, 1998] Drew McDermott. The Planning Domain Definition Language Manual. Technical Report 1165, Yale Computer Science, 1998. (CVC Report 98-003) Available at `ftp://ftp.cs.yale.edu/pub/mcdermott/software/pddl.tar.gz`.

Semantic Email

Oren Etzioni, Alon Halevy, Henry Levy, and Luke McDowell
Computer Science and Engineering
University of Washington
{etzioni,alon,levy,lucasm}@cs.washington.edu
http://www.cs.washington.edu/research/semweb/email

1 Introduction

There is significant interest in making portions of the WWW machine understandable as part of the broad vision known as the "Semantic Web". While the WWW is a rich information space in which we spend significant amounts of time, many of us spend even more time on email. In contrast to the WWW, where most of our interactions involve consuming data, with email we both create and consume data. With the exception of the generic header fields associated with each email message, email messages typically do not have semantic features. While the majority of email will remain this way, this paper argues that adding semantic features to email offers opportunities for improved productivity while performing some very common tasks. To illustrate, consider several examples:

- In the simplest case, imagine sending an email with a talk announcement. With appropriate semantics attached to the email, sending this message can also result in automatically (1) posting the announcement to a talks web site, and (2) sending a reminder the day before the talk.
- Suppose you are organizing a PC meeting and you want to know which PC members will stay for dinner after the meeting. Currently, you need to send out the question and compile the replies *manually*, leafing through emails one by one. With semantic email, the PC members can provide the reply in a way that can be interpreted by a program and compiled properly. In addition, after a few days, unresponsive PC members can be automatically reminded to respond, and those who have said they're not coming to the PC meeting need not be bothered with this query at all. This represents a substantial improvement over current practice where members of mailing lists are subjected to repeated entreaties to respond, even though many of them have already done so.
- As a variant of the above example, suppose you are organizing a *balanced potluck*, where people should bring either an appetizer, entree, or dessert, and you want to ensure that the meal is balanced. In addition to the features of the previous example, here semantic email can help ensure that the potluck is indeed balanced by examining the replies and requesting changes where necessary.
- As a final example, suppose you want to give away tickets to a concert that you cannot use. You would like to send out an announcement and have the semantic email system give out the tickets to the first respondents. When the tickets are gone, the system should respond politely to subsequent requests. Alternatively, you may sell the tickets to the highest bidder and have the system help you with that task.

These examples are of course illustrative rather than exhaustive. Because email is not set up to handle these tasks effectively, accomplishing them manually can be tedious, time-consuming, and error-prone.

In general, there are at least three ways in which semantics can be used to streamline aspects of our email habitat:

1. **Update:** we can use an email message to add data to some source (e.g., a web page, as in our first example).
2. **Query:** email messages can be used to *query* other users for information. Semantics associated with such queries can then be used to automatically answer common questions (e.g., asking for my phone number or for directions to my office).
3. **Process:** semantic email can manage simple but time-consuming processes that we currently handle manually.

The techniques needed to support the first two uses of semantic email depend on whether the message is written in text by the user or formally generated by a program on the sender's end. In the user-generated case, we would need sophisticated methods for extracting the precise update or query from the text. In both cases, we require some methods to ensure that the sender and receiver share terminologies in a consistent fashion.

This paper focuses on the third use of semantic email to streamline processes, as we believe it has the greatest promise for increasing productivity and is where the most pain is currently being felt by users. Some hardcoded email processes, such as the meeting request feature in Outlook, invitation management via *Evite*, and contact management via *GoodContacts*, have made it into popular use already. Each of these commercial applications is limited in its scope, but validates our claim about user pain. Our goal in this paper is to sketch a *general* infrastructure for semantic email processes. Feature rich email systems such as Microsoft's Outlook/Exchange offer forms and scripting capabilities that could be used to implement some email processes. However, it is much harder for casual users to create processes using arbitrary scripts, and furthermore, the results would not have the formal properties that our model provides.

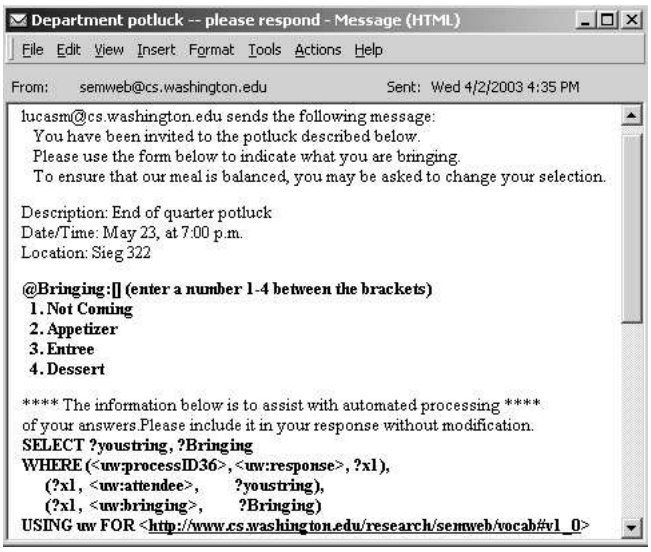


Figure 1: A message sent to recipients in a "Balanced potluck" process. The bold text at the top is a form used for human recipients to respond, while the bold text at the bottom is a query that maps their textual response to a formal language (e.g., RDF).

2 Formal model of Semantic Email Processes

We model a *semantic email process* (SEP) as an RDF *data set* affected by *messages* from a set of *participants*, controlled by a set of *constraints* over the data set.[1] For instance, when executing we may constrain a "potluck" process so it results in a balanced number of appetizers, entrees, and desserts. Figure 1 shows the initial message that would be sent to the participants in such a process. Users respond via any email client, and then (based on the constraints and other responses so far) the system either accepts the response or suggests alternative choices to the participant.

Our model enables us to pose several formal inference problems that can help guide the creation of SEPs as well as manage their life cycle. For instance, we have proven that, in many common cases (including all of the examples described here), the problem of inferring whether a specific message from a participant may be accepted and still allow the process constraints to be *eventually satisfied* is in P-time [Etzioni *et al.*, 2003]. Other tractable (and useful) inference problems include the ability to determine the set of all possible responses that may be accepted by a process in its current state.

3 Implementation Status

We have developed a prototype semantic email system and deployed it for public use.[2] So far we have developed simple processes to perform functions like collecting RSVPs, giving tickets away, and organizing a balanced potluck; these can be customized for many other purposes (e.g., to collect N volunteers instead of give away N tickets).

The prototype is integrated within our larger MANGROVE [McDowell *et al.*, 2003] semantic web system. This provides us with an RDF-based infrastructure for managing email data and integrating with web-based data sources and services. For instance, the MANGROVE web calendar accepts event information via email or from a web page. In addition, the RSVP email process could easily be expanded to accept an event description from an existing web page, then monitor this web data for location or time changes to include in a reminder email. Likewise, a semantic email client could utilize data from MANGROVE to answer common questions. When previously unknown questions are answered manually by the user, these responses could be stored for future use, thus enabling the automatic acquisition of semantic knowledge over time. Future work will consider additional ways to synergistically leverage data from both the web and email worlds.

[1] Note that the *users* of SEPs are not expected to understand or directly use the formal model. Generic SEPs are created by programmers and *invoked* by untrained users via a simple form.

[2] See www.cs.washington.edu/research/semweb/email

4 Related Work and Conclusion

Information Lens [Malone *et al.*, 1987] used forms to enable a user to generate a single email message with semi-structured content that might assist recipients with filtering and prioritizing that message. Mangrove's SEPs generalize this earlier work by enabling users to create an email *process* consisting of a set of interrelated messages governed by useful constraints. In addition, Mangrove extends Information Lens's rule-based message processing to support more complex reasoning based on information from multiple messages and data imported from web sources. Consequently, Mangrove's SEPs support a much broader range of applications than those possible with Information Lens [Etzioni *et al.*, 2003].

We have introduced a paradigm for semantic email and described a class of semantic email processes. These automated processes offer tangible productivity gains on email-mediated tasks that are currently performed manually in a tedious, time-consuming, and error-prone manner. Moreover, semantic email opens the way to scaling similar tasks to large numbers of people in a manner that is not feasible with today's person-processed email. For example, large organizations could conduct surveys and voting via email with guarantees on the behavior of these processes. Future work will explore additional applications, extend our formal analysis, and investigate any impediments to widespread adoption.

Finally, we see semantic email as a first step in a tighter integration of the semantic web and email. In essence, we have described a concrete approach to generalizing the original vision of the semantic web to also encompass email.

References

[Etzioni *et al.*, 2003] Oren Etzioni, Alon Halevy, Hank Levy, and Luke McDowell. Semantic email: Adding lightweight data manipulation capabilities to the email habitat. In *Sixth International Workshop on the Web and Databases*, 2003.

[Malone *et al.*, 1987] Thomas Malone, Kenneth Grant, Franklyn Turbak, Stephen Brobst, and Michael Cohen. Intelligent information-sharing systems. *Communications of the ACM*, 30(5):390–402, 1987.

[McDowell *et al.*, 2003] Luke McDowell, Oren Etzioni, Steven D. Gribble, Alon Halevy, Henry Levy, William Pentney, Deepak Verma, and Stani Vlasseva. Mangrove: Enticing ordinary people onto the semantic web via instant gratification. In *Second International Semantic Web Conference*, October 2003.

Static Knowledge Provenance

Mark S. Fox and Jingwei Huang
Enterprise Integration Laboratory, University of Toronto
40 St. George Street, Toronto, ON M5S 3G8 Canada
msf@eil.utoronto.ca
jingwei@eil.utoronto.ca

1. Introduction

This paper addresses the problem of how to determine the validity of web information. The problem arises from many directions: information may no longer be relevant (e.g., discontinued products or old operating procedures), may contain incorrect information (e.g., news stories), or may even be outright lies. For example, in 1999, two men posted fraudulent corporate information on electronic bulletin boards, which caused the stock price of a company (NEI) to soar from $0.13 to $15, resulting in their making a profit of more than $350,000 [Mayorkas]. This example reveals a problem: anyone can publish information on the web, the information may be true or false, valid or dated, however, no tool exists to discern the differences.

In this paper, Knowledge Provenance (KP) is proposed to address this problem by introducing standards and methods for how to model and maintain the evolution and validity of web information. KP need answer the following major questions. For any piece of web information, what is the truth value of it? who created it? Can it be believed? KP builds on research in trust management by providing means of propagating information validity over the web assuming its original sources are trusted.

Philosophically, we believe the web will always be a morass of uncertain and incomplete information. But we also believe that it is possible to annotate web content to create islands of certainty. Towards this end, Knowledge Provenance introduces 4 levels of Provenance that range from strong provenance (corresponding to high certainty) to weak provenance (corresponding to high uncertainty). Level 1 (Static KP) focuses on provenance of static and certain information; Level 2 (Dynamic KP) considers how the validity of information may change over time; Level 3 (Uncertain KP) considers information whose validity is inherently uncertain; Level 4 (Judgment-based KP) focuses on social processes necessary to support provenance. This paper focuses on Static KP.

2.What is Static Knowledge Provenance?

The basic unit of web information to be considered in KP is a "proposition". A proposition, as defined in First Order Logic, is a declarative sentence that is either true or false. A proposition is the smallest piece of information to which provenance-related attributes may be ascribed. Static KP focuses on provenance of static and certain information. Basically, any proposition has a truth value of: True, False or Unknown. The default truth value is "Unknown".

In the following, the underlying concepts of Static Knowledge Provenance are explored in the context of two case studies.

Case 1: Asserted Information
Consider the proposition found on a web page that "perennial sea ice in the Arctic is melting faster than previously thought at a rate of 9 percent per decade." From a provenance perspective, there are three questions that have to be answered: 1) what is the truth value of this proposition? 2) Who asserted this proposition? 3) Should we believe the person or organization that asserted it? In this example, a further examination of the text of the web page provides the answers: it can be believed as a true proposition, asserted by NASA, whom most people believe is an authority on the subject. Question is, how can this provenance information be represented directly without having to resort to Natural Language Processing of the page?

Other examples of asserted information include assertions made by persons or organizations, statistical data and observation data such as stock quotes and weather readings issued by organizations.

Case 2: Dependent Information
Consider the following information found in another web page: "In 2002, a satellite-based survey [NASA2002] found that 'Arctic sea ice coverage fell from around 6.5 million square kilometres to around 5.5 million square kilometres in one year'. The melting sea ice threatens to drive polar bears extinct within 100 years." It contains two propositions. The first is a quotation of the proposition in the previous case. The second is a derived conclusion, and the first is a premise of the second. What makes this case more interesting is that determining the truth of theses propositions is dependent upon other propositions that may be in other web pages. These types of propositions are called "dependent propositions" in KP. There are two types of dependency occurring. The first is quotation. The reproduction of a proposition is called "equivalent proposition" for it has the equivalent truth value as original proposition. Secondly, a proposition can be derived using logical deduction. Hence, the truth of the derived conclusion depends on the truth of the

premise and upon some hidden reasoning that led to the deduction. This type of derived proposition is classified as "derived information".

In practice, a proposition may be derived by applying different axioms. Derived propositions may also be dependent upon disjunctions, conjunctions and/or negations of other propositions.

From these two cases, a number of concepts required for reasoning about provenance emerge:

- Text is divided into propositions;
- Asserted proposition must have digital signature to guarantee author identification and information integrity;
- To believe an asserted proposition, its creator must be trusted on a topic which the assertion belongs to;
- Info dependencies must be maintained;
- A dependent proposition can be an equivalent copy or the result of a reasoning process;
- Validity judgment is based on trust relations (whom can be trusted in a specific field) and information dependency. So, provenance is context sensitive. The context is the trust relations that the provenance requester has.

3. Axioms

15 Static KP axioms have been defined in FOL to specify truth conditions of KP-props [Fox & Huang 2003]. Major considerations of the axioms are as follows:

- A proposition is "trusted", if its creator is "trusted" in the topic covering the proposition, and its digital signature is "Verified".
- An asserted-prop has its trusted truth value* as specified by its creator, if it is trusted.
- An equivalent-prop has the same trusted truth value as the proposition it depends on, if this equivalent-prop is "trusted".
- A derived-prop has its trusted truth value as specified, if it is "trusted" and the proposition it depends on (premise) is trusted to be "True".
- The creator and digital signature of a web document are the default creator and digital signature of each proposition contained in the web document.

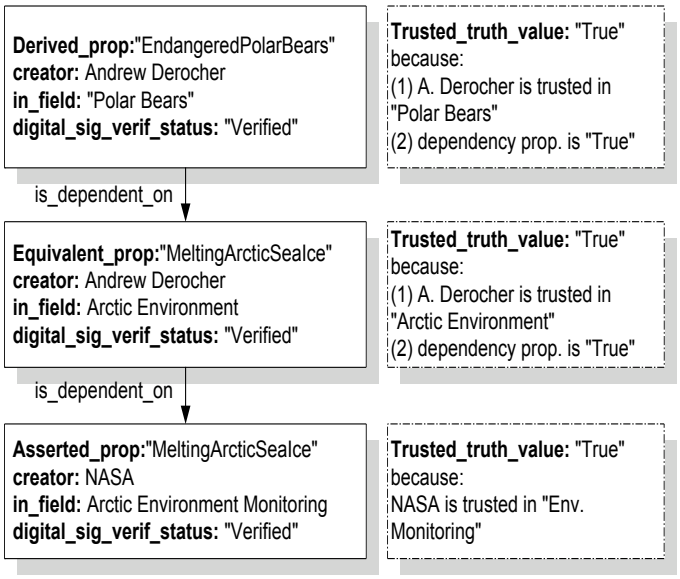
4. Implementation & Example

In order to use knowledge provenance to judge the validity of web information, two tasks need to be done: (1) to annotate web documents with KP metadata. We define KP metadata using RDFS; (2) to develop an online KP agent to conduct provenance reasoning on propositions contained in web documents by using KP axioms.

The following is a piece of example to annotate one proposition. The entire annotation example can be found in [Fox&Huang2003].

```
<kp:Derived_prop rdf:id="EndangeredPolarBears"
      truth_value="True"
      is_dependent_on="#MeltingArcticSeaIce"
      creator="Andrew Derocher"
      in_field="Polar Bears"
>
   <kp:proposition_content> The melting sea ice threatens to drive polar bears extinct within 100 years
   </kp:proposition_content>
</kp:Derived_prop>
```

For the first step, we have implemented the Static KP model with an experimental system, called RDFS-Prolog. The following figure illustrates provenance reasoning in the second case of section 2.



A Static Knowledge Provenance analyzer (based on earlier version of KP1 model) has been implemented in JAVA as a service available over the web at http://www.eil.utoronto.ca/kp1/. Given a URL, the analyzer extracts KP-props and their descriptions, and follows paths through the web to accumulate provenance information. The KP analysis result is then displayed in the web browser.

References

Berners-Lee, T., Hendler, J., and Lassila, O., "The Semantic Web", Scientific American, May 2001.

Blaze,M.,J. Feigenbaum & J. Lacy, Decetralized Trust Management, IEEE Conf. Security and Privacy,1996.

Khare, R., and Rifkin, A., "Weaving and Web of Trust", World Wide Web Journal, Vol. 2, No. 3, pp. 77-112.

Mayorkas, http://www.usdoj.gov/usao/cac/pr/pr2000/003.htm

Fox, Mark S., and Huang, Jingwei, "Knowledge Provenance: An Approach to Modeling and Maintaining the Evolution and Validity of Knowledge", EIL Technical Report, University of Toronto, 2003. http://www.eil.utoronto.ca/km/papers/fox-kp1.pdf

* The truth value that the provenance agent believes a proposition has, is called trusted truth value.

Understanding the Semantic Web through Descriptions and Situations

Aldo Gangemi and Peter Mika

Laboratory for Applied Ontology, Institute for Cognitive Sciences and Technology, National Research Council, I-00137 Rome, Italy
gangemi@ip.rm.cnr.it

Vrije Universiteit Amsterdam, 1081HV Amsterdam, The Netherlands
pmika@cs.vu.nl

1 Introduction

We propose a mechanism to mime the human cognitive ability to contextualize our ontological commitments, even when we have scanty evidence of them. This ability originates from extensive reification, and from the representation of other cognitive processes described e.g. by Gestalt psychology [Köhler, 1947], which allow us to refer synthetically to some commonly agreed context labels.

From the Semantic Web perspective, we propose that - when a complete theory is lacking- we may still recurse to context to help interpretation. An ontological context can be preliminarily defined here as a first-order entity, usually quite complex, which is defined by certain typical elements that result from the reification of the elements of a theory.

We have developed and are exploiting an ontology of contexts, called Descriptions and Situations (D&S), which provides a principled approach to context reification through a clear separation of states-of-affairs and their interpretation based on a non-physical context, called a description. The ontology of descriptions also offers a situtation-description template and reification rules for the principal categories of the DOLCE foundational ontology. Both DOLCE and the D&S extension to DOLCE are being developed in the EU WonderWeb project[1].

2 Approach

Foundational ontologies such as DOLCE are ontologies that contain a specification of domain-independent concepts and relations based on formal principles derived from linguistics, philosophy, and mathematics [Masolo *et al.*, 2002]. While formalizing the principles governing physical objects or events is relatively straightforward, intuition comes to odds when an ontology needs to be extended with *non-physical objects*, such as social institutions, organizations, plans, regulations, narratives, mental contents, schedules, parameters, diagnoses, etc.

In general, we feel entitled to say that representing ontological (reified) contexts is a difficult alternative to avoid, when so much domain-oriented and linguistic categorisations involve reification. However, we also want to provide an explicit account of the contextual nature of non-physical entities and thus aim for a reification that accounts to some extent for the partial and hybrid structure of such entities.

From the logical viewpoint, any reification of theories and models provides a first order representation. From the ontological engineering viewpoint, a straightforward reification is not enough, since the elements resulting from reification must be framed within an ontology, possibly built according to a foundational ontology.

3 D&S: an ontology of descriptions

The Descriptions and Situations ontology (D&S) is an attempt to define a theory that supports a first-order manipulation of theories and models, independently from the particular foundational ontology it is plugged in.

When we try to describe a state of affairs (not considered here as a model) according to a theory, some structure (a model) emerges (this reflects the "cognitive structuring" process). The emerging structure is not necessarily equivalent to the "real" structure.

D&S represents this intuition as an "epistemological layering", consisting of assuming that any logical structure L_i (either formal or capable of being at least partly formalised) is built upon a state of affairs described according to a theory T_i (either formal or capable of being at least partly formalised).

In other words, T_i describes what kind of ontological commitment L_i is supposed to represent within the epistemological layer that is shared by the encoder of an ontology. Epistemological layering reflects the so-called "figure-ground shifting" cognitive process.

D&S implements reification rules for any T_i, called a description, and a basic framework for any L_i, called a situation, and for their elements.

3.1 Implementation of D&S in DOLCE

DOLCE has four top categories: endurant (including object and substance-like entities), perdurant (event- and state-like entities), quality (individual attributes), and abstract (mainly conceptual regions for attributes of entities) [Masolo *et al.*, 2002].

A situation is a (new) top category in DOLCE, while a description is a non-physical endurant. A description may be satisfied by a state of affairs. A description satisfied by a state of affairs is an s-description. A state of affairs satisfying a description is a situation.

[1] http://wonderweb.semanticweb.org

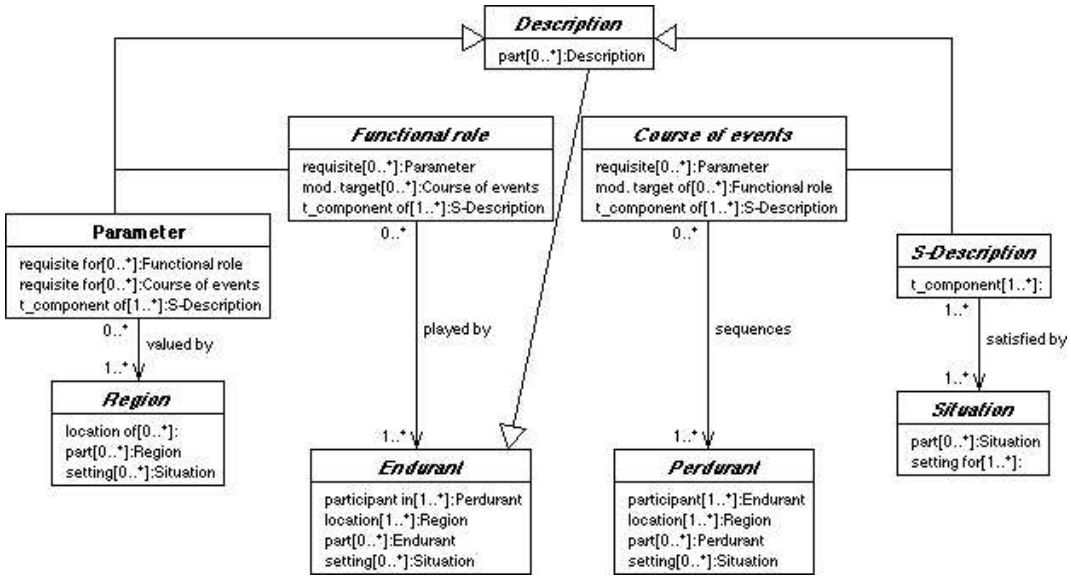


Figure 1: UML overview of the D&S ontology of descriptions

Concerning the reification of the elements of a theory, the descriptions that reify a selection rule on DOLCE regions (e.g. speed limit or visibility) are called parameters, the descriptions that reify a functional property of DOLCE endurants (e.g. citizen or judge) are called functional roles, and the descriptions that reify sequences of DOLCE perdurants (e.g. schedule or pathway) are called courses.

Situations and s-descriptions are systematically related as shown in Fig. 1. The basic relation is "selects", and it reifies the instantiation relation between an individual in a model and a concept in a theory. Within DOLCE, selects relates components of an s- description to instances of DOLCE categories. Intuitively, selects(x,y) binds an individual y classified in a DOLCE category to a situation s that satisfies the s-description d that has x as a component. In particular: parameters are valued-by regions, f-roles play endurants, and courses sequence perdurants.

D&S results to be a theory of ontological contexts because it is capable to describe various notions of context (physical and non-physical situations, topics, provisions, plans, assessments, beliefs, etc.) as first-order entities.

Examples of descriptions and situations include a clinical condition (situation) with a diagnosis (s-description) made by some agent (f-role), a case in point (situation) constrained by a certain norm (s-description), a murder (situation) reported by a witness (functional role) in a testimony (s-description), a 40kmph (region) as the value for a speed limit (parameter) in the context of an accident (state of affairs) described as a speed excess case (situation) in an area covered by traffic code (s-description) etc.

4 Applications

The Descriptions and Situations ontology as a template for context modelling have been applied in a number of ontology developments:

- **An ontology of communication.** We have used D&S to formalize Roman Jacobson's theory of communication and the theory of semiotics developed by Ferdinand de Saussure. Theories of communication and interpretation exhibit a clear contextual nature in giving structure ("meaning") to an underlying exchange of symbols.

  We have extended and used this ontology to describe communication in a Semantic Web experiment, a peer-to-peer ontology-based knowledge sharing environment developed within the EU SWAP project [2].

- **An ontology of Web Services.** In our latest work, we apply the D&S template to develop an ontology of (web) services which takes into account the multitude of views on a service: the offering of the provider, the expectations of the requestor, the contract agreed, the service norms etc.

  This ontology serves as an upper layer of the ontologies used to describe the software components hosted by the Application Server for the Semantic Web (ASSW), the central brokering facility in the WonderWeb infrastructure.

References

[Köhler, 1947] Wolfgang Köhler. *Gestalt Psychology*. Liveright, New York, 1947.

[Masolo *et al.*, 2002] Claudio Masolo, Stefano Borgo, Aldo Gangemi, Nicola Guarino, Alessandro Oltramari, and Luc Schneider. The WonderWeb Library of Foundational Ontologies. WonderWeb Deliverable 17, 2002.

[2] `http://swap.semanticweb.org`

Grounding Semantic Markup in Text: An Interactive Approach

Yolanda Gil and Varun Ratnakar
USC Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA 90292
gil@isi.edu, varunr@isi.edu

Abstract

We propose a new approach to develop semantic annotations that captures at different levels of formality and specificity how a user decided to render each statement after consulting a set of documents that may or may not be consistent or contributing to the final statement entered by the user. We believe that this kind of trace of information about how each annotation is defined will make the annotations easier to reuse, extend, and translate. We are investigating these issues with IKRAFT, an interactive tool to elicit from users the rationale for choices and decisions as they analyze information used in building semantic markup annotations. IKRAFT helps users create semantic markup grounded in the original documents that the user consulted to create it, including documents that were considered but were dismissed and intermediate statements used in the creation of the final markup.

Introduction

Our work investigates an alternative design of ontologies and knowledge bases in the Semantic Web that may avoid the challenges that arise in understanding, reusing, extending, translating, and merging existing technology for knowledge bases. Large knowledge bases contain a wealth of information, and yet browsing through them often leaves an uneasy feeling that one has to take the developer's word for why certain things are represented in certain ways, why other things were not represented at all, and where might we find a piece of related information that we know is related under some context. Whatever fits the language will be represented and other things are left out, for reasons such as available time and resources or perhaps lack of detailed understanding of some aspects of the knowledge being specified. When the knowledge base needs to be extended or updated, the rationale for their design is lost and needs to be at least partially reconstructed. The knowledge sources are no longer readily available and may need to be accessed. While it is the case that entire knowledge bases can be reused and incorporated into new systems, it is harder to extract only relevant portions of them that are appropriate in the new application. Parts of the knowledge base may be too inaccurate for the new task, or may need to be modeled in a different way to take into account relevant aspects of the new application.

The goal of our work is to capture the results of analyzing various information sources consulted by content developers as they design the detailed contents of a knowledge base. IKRAFT (Interactive Knowledge Representation and Acquisition from Text) is a tool that enables content developers to keep track of the knowledge sources and intermediate knowledge fragments that result in a formalized piece of knowledge, described in detail in [1]. We have extended IKRAFT so that it can be used to create RDF Schemas that are linked to the original documents consulted by the user and to intermediate statements derived from those documents. The resulting semantic markup is enhanced with pointers that capture the rationale of its development.

Figure 1 illustrates how IKRAFT helps users create annotations. First, the user selects original sources (shown on the top right) and selects from them relevant knowledge fragments by highlighting them in the source text. Then the user restates the knowledge fragments in terse English statements (shown on the top left). Typically these new fragments are phrased as unambiguously and briefly as possible. They may be organized in a list of items and sub-items. The developer may combine two or more fragments into one sentence, or break a fragment into several sentences that reflect different aspects of the content discussed. IKRAFT will keep pointers back to the document fragments that were highlighted by the user in creating each statement. Finally, the user formalizes those fragments into the target representation (shown at the bottom). Notice that some of the fragments may extend existing definitions in pre-developed schemas or ontologies. IKRAFT generates RDF Schemas to reflect the classes and constraints defined by the user, and include pointers to the original documents.

Figure 2 shows how IKRAFT supports an application that we are currently developing to create end-to-end earthquake simulations from smaller components that model different aspects of the simulation and represented as web services [2]. Each simulation model is designed by scientists to take into account specific types of earth shaking phenomena, which result in constraints that should be taken into account by the end users (e.g., building engineers) using the models. In this application, users can access the documentation of the simulation models by retrieving the IKRAFT annotations that justify each constraint.

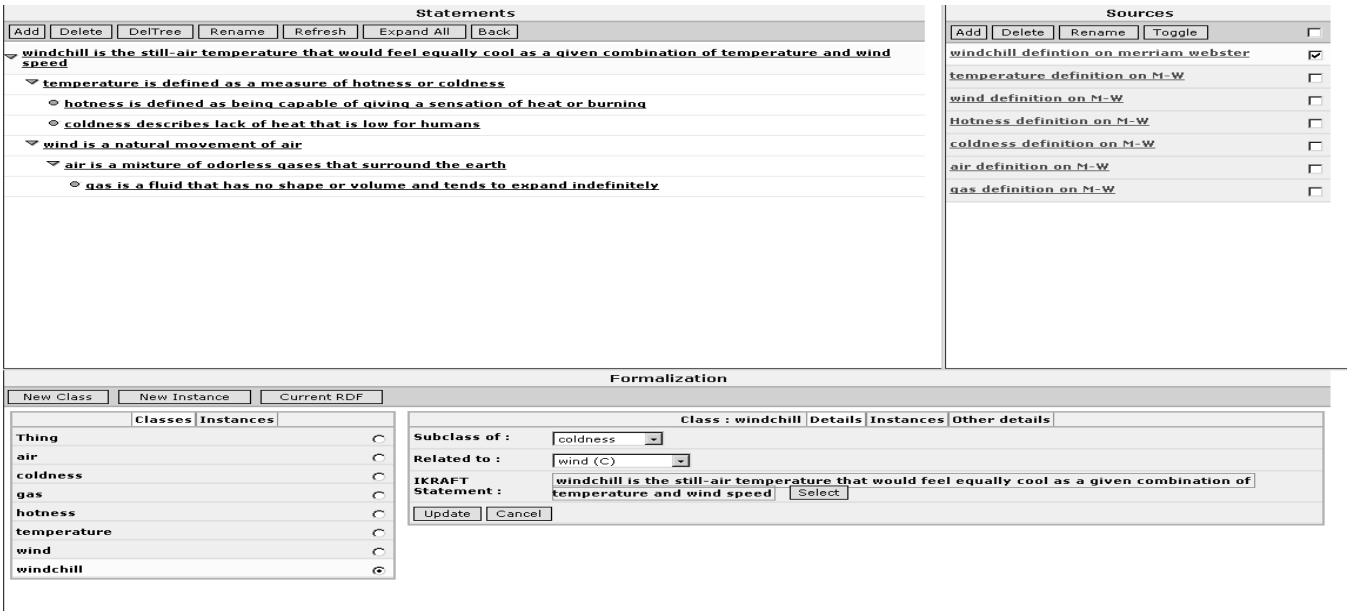


Figure 1: IKRAFT interface for creating semantic annotations.

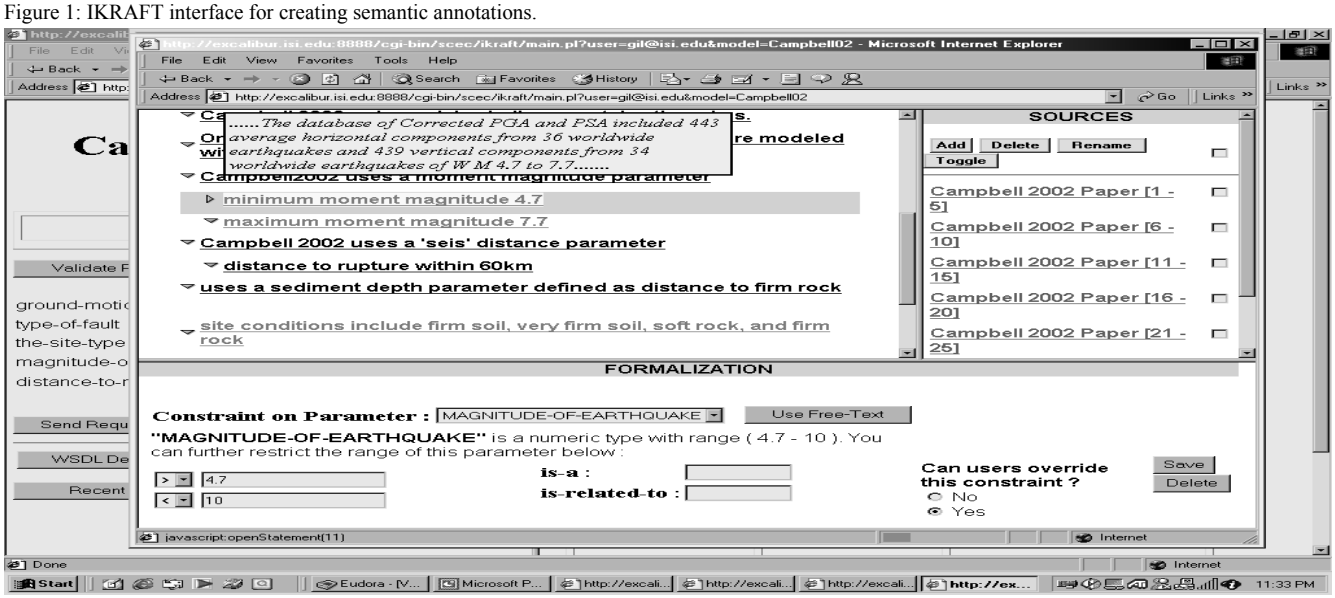


Figure 2. Using IKRAFT in an earthquake simulation application.

Backing up the formalized constraints with the appropriate documentation sources is very useful, especially when users need to make judgments about the severity and possible dismissal of constraint violations. In addition, it is useful to accommodate constraints in different degrees of formalization since some characteristics of the models are hard to formalize (e.g., that the model relies on "recordings with unknown or poor estimates of magnitude mechanism distance or site excluded from data set").

In summary, in developing a semantic model of some body of knowledge, users may consult many sources presenting contradictory or complementary information, analyze the different implications of each alternative belief, and decide what and how to model the knowledge. Instead of having annotations that only represent their final beliefs, IKRAFT captures the rationale for modeling the knowledge the way it appears in the semantic annotations. We believe that this facilitates future extensions of the annotations by other users, as well as integration across diverse schemas and ontologies.

We are currently extending IKRAFT with natural language processing tools to support the formalization of the statements and the mapping of terms to pre-existing schemas and ontologies. We continue to use it for content development, and plan to release IKRAFT open source in the near future.

References

[1] Y. Gil and V. Ratnakar. 2002. "IKRAFT: Interactive Knowledge Representation and Acquisition from Text", Proceedings of EKAW-02. http://www.isi.edu/~gil/papers/ikraft-ekaw02.pdf.

[2] http://www.isi.edu/ikcap/docker.

Semantic groupware and its application to KnowWho using RDF

Nobuyuki Igata and **Hiroshi Tsuda** and **Yoshinori Katayama** and **Fumihiko Kozakura**
Fujitsu Laboratories Ltd.
4-1-1 Kamikodanaka Nakahara Kawasaki Kanagawa, 211-8588, Japan
{igata, htsuda, katayama.yoshin, kozac}@jp.fujitsu.com

1 Introduction

This paper presents a novel approach to apply Semantic Web technologies to groupware and KnowWho in Knowledge Management.

Most groupware combine some applications such as a scheduler, mailer, BBS, etc. and integrate information in application-oriented manners. Generally, they work well if all the group members use the same product. However, as our work style is changing rapidly and a group is becoming flexible, it is hard to imagine all members use the same software application. Here, required information sharing approach is not by application-oriented but contents-oriented.

The Semantic Web is also aimed at integrating heterogeneous Web contents in a content-oriented manner by using metadata and ontology. To cope with the above problems in groupware, we utilize Semantic Web technologies such as RDF (Resource Description Framework) and Web Ontology to Knowledge Management in intranets.

2 Semantic groupware: WorkWare++

Our system called *WorkWare++* is not only a yet another groupware but also a meta-level groupware that can produce, integrate, and manage metadata about people, documents, schedulers, and so on of heterogeneous applications using RDF.

Figure 1 shows the architecture of WorkWare++. WorkWare++ is composed of three layers: the application layer, the metadata layer, and the multiple views layer. WorkWare++ not only manages persons' schedule, but also semi-automatically relates employee databases, office documents, E-mail, and schedule information semantically in the metadata layer. The relations are stored in RDF metadata.

Metadata management of WorkWare++ consists of two steps: metadata generation and link attachment.

First, WorkWare++ generates five kinds of metadata from applications as shown in Figure 2. Properties of *Document* objects are extracted from office documents and E-mail by information extraction technologies. Properties of *Employee* are given from the employee database. Properties of *Schedule* objects are given from the scheduler. Sometimes, the same meeting is represented in different strings by different person. To integrate such information, WorkWare++ semi-automatically generates *Meeting* objects.

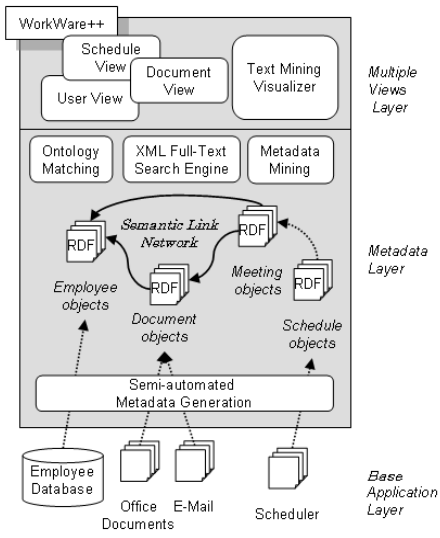


Figure 1: Architecture of WorkWare++

Second, WorkWare++ attaches links among existing different kind of RDF nodes such as *Meeting-Employee* and *Document-Employee*, using ontology matching techniques such as name identification. For example, by comparing *Employee*'s Name to *Document*'s Author, the name identification makes the link between them as "dc:creator" link. The links between *Employee* objects and *Meeting* objects is also similar as "Participant" link. The links between *Document* objects and *Meeting* objects are currently given manually as "Has_a" link. Thus, each object is semi-automatically connected, and becomes to a large-scale network structure.

3 KnowWho in WorkWare++

KnowWho processes are not so simple as to input several keywords and find related person like the bag-of-words model in document retrieval. There are several paths to find and reach desired people, for example, skill keywords, related documents, related person, and so on. Here, we treat the KnowWho process of WorkWare++ as a sequence of searching and visualizing information around people, documents, schedules, and skill keywords.

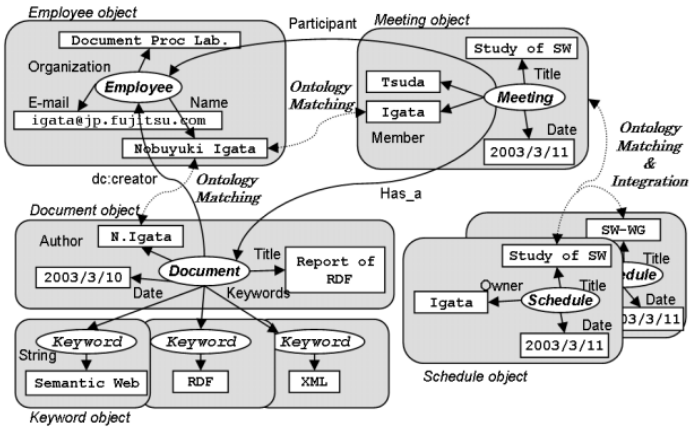


Figure 2: Metadata in WorkWare++

WorkWare++ enables the following navigation steps to access a skilled person from starting topic keywords.

1. Find target technologies from starting topic keywords (*Technical Term Map*).
2. Find skilled groups of the target technologies (*Personal Connection Map*).
3. Find the most skilled person in the group by comparing personal skills (*Personal Skill History Map*).

The navigation steps are performed with a high speed full-text XML search engine [Nakao and Igata, 2002] and visual text mining engine . We applied WorkWare++ with about 1,000 employees and tens of thousands of documents.

Figure 3 is an example of the technical term map. This map visualizes the relations between technical terms and organizations that relate to a topic keyword “XML”. This map is made from *Document* (with *Keyword*), and *Employee* objects in Figure2. Node’s names of the map, for example, are given from *Keyword*’s String and *Employee*’s Organization. The relevance of each node is derived from the co-occurrence in the same document.

Figure 4 is an example of The personal connection map. This map visualizes the closeness of people, which is derived from the co-participant relations from *Meeting*. Related people are selected from *Meeting* and the number of *Document* that contain the relevant keywords. The map clarifies key-persons who connect the subgroups.

The personal skill map visualizes personal skill keywords in a time series. Transition of skill keywords is derived from combination of *Employee* and *Document*.

4 Concluding Remarks

WorkWare++ has the following features:

1. Semi-automatically generating RDF metadata around person from his routine work.
2. Searching and visualizing information around people, documents, schedules, and skill keywords.

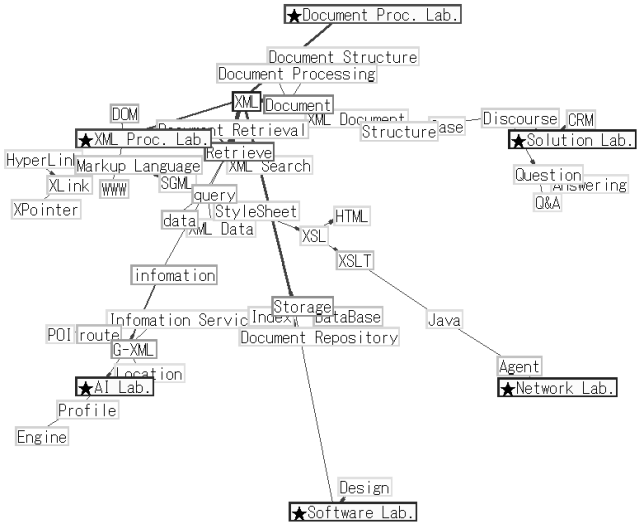


Figure 3: Technical Term Map for a topic keyword “XML”

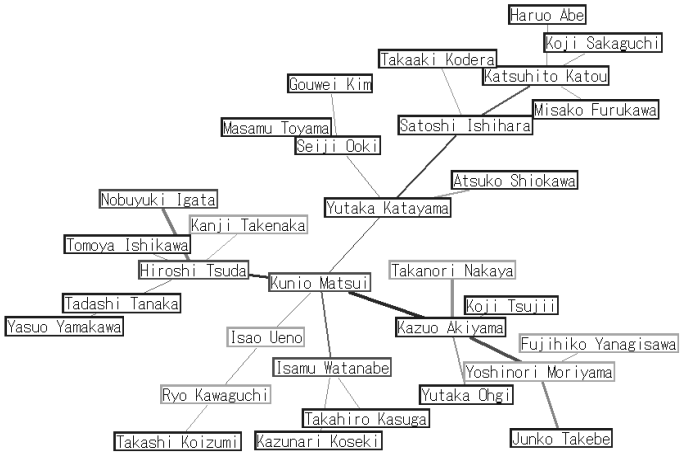


Figure 4: Personal Connection Map

We checked several KnowWho results of more than 1,000 employees of our research departments and the quality is reasonable. But it may be based on the work style of researches, i.e., most of the results are given as a set of documents such as papers. Researchers who output a certain keyword frequently in papers can be seen as specialists of the keyword topic. However, another kind of workers such as System-Engineers, they seldom write accurate documents. Even in the intranets, not all the information is trustworthy. We are now extending WorkWare++ to other departments, and the solution to this problem is one of our remaining research topics.

References

[Nakao and Igata, 2002] Yoshio Nakao, Nobuyuki Igata: *XAR: an XML Document Retrieval System as a Tool for Analysis of Biomedical Literature*. Pacific Symposium on Biocomputing 2003.

DL-workbench: a meta-model based platform for ontology manipulation

Mikhail Kazakov [1,2] **, Habib Abdulrab** [2]
[1] Open Cascade S.A., mikhail.kazakov@opencascade.com
[2] INSA de Rouen, abdulrab@insa-rouen.fr

1 Introduction

Nowadays many application domains are looking for use of ontologies. And each area, that uses ontologies, sets its own requirements for tools. One of such areas is the software integration domain. The main challenge of integration is: how to make working together software entities that were not initially created to work with each other. We are currently working on the topic of semi-automated integration of various numerical simulation multi-physics solvers [Salome, 2003] within a distributed environment. Here the use of ontologies (as formal description models) helps to share the same view on specification of the solvers coming from different vendors. Reasoning procedure allows retrieving required software configuration. Integration topic is out of the scope of this article and will be published in a separated paper. Some preliminary results were given in [Kazakov, 2003].

Taking into account the research origin of our work, we have formulated a set of requirements for an ontological tool that would be convenient when working within a specific application domain:

- Full support of at least one ontological formalism (such as description logics [Baader, 2003], first order logic and others) with presence of reasoners
- Convenient user interface for manipulation of logical expressions
- Ability to integrate that tool with a specific domain environment (API)
- Ability to manipulate ontologies and their elements from specific domain environment (API)
- Fast and portable user interface
- Ability to work with structured "ontological projects" but not with "files".
- Ability to switch among ontological formalisms for research purposes

We noticed that the same requirements arise often within the variety of other domains where ontologies are used (manufacturing engineering, configuration, etc.).

Then we have studied the state of the art of ontology creation/edition tools and programming interfaces. However no tool complied with our requirements and was convenient for our needs. Thus the decision to create own platform was taken. The name of the platform is DL-workbench (short of Description Logic Workbench, since the SHIQDn_ description logic [Horrocks, 2003] is the main formalism that is used). DL-workbench is published under open source license (http://www.opencascade.org/dl-workbench).

The DL-workbench is a meta-model based ontological edition platform. The extension and integration APIs are open and documented. Meta-model allows us to switch easily between different ontological formalisms and gives the ability of easy and extensible integration with other environments. DL-workbench is implemented as a set of IBM Eclipse plug-ins using Java language. That fact allows seamless integration of DL-workbench with many other software engineering tools (IBM WebSphere, Rational XDE, etc.).

Semantic Web activity is an important world-wide effort that combines many techniques and touches many application domains. We believe that our experience in creating of software tools for manipulation of ontologies (and DL-workbench itself) will be useful for the semantic web community.

2 DL-workbench

The next list of principles was formulated and it constitutes the main concept of DL-workbench:

- DL-workbench is based on a meta-model that is capable to describe the structure of ontological formalisms and third-party data that can be used within a specific domain.
- DL-workbench has a modular (plug-in based) architecture with clearly specified dependencies among modules.

Main processing module of DL-workbench is based on the meta-model. It does not depend on any of specific formalisms. This module implements features that can be implemented using only generic meta-model (persistence skeleton, reasoning skeleton, tracking of changes, transactions, lifecycle of instances and many others). Only top level modules (leafs of module dependencies tree) are formalism-specific. Each generic module provides a set of documented extension points that can be used by other modules / software to customize the platform.

DL-workbench is based on the following principles:

- The work with ontologies is performed using a notion of a project. A project here is a structured set of ontological files (ontological resources) and other domain specific data if needed.

- DL-workbench supports manipulation/edition of complex expressions and axioms. In many application domains, the complete and "reasoning-able" ontologies require the use of logical expressions and axioms.
- DL-workbench defines an internal data model and "UI-ready" data model. These allow organization of different views on the same data (i.e. project view, namespace view, taxonomy view).
- DL-workbench provides a structured and documented API over all above mentioned features. DL-workbench provides the user interface of an ontological editor.

We strongly believe that an ontological manipulation platform, integrated with some software engineering environment, shall be based on these principles to be and useful for researchers, software architects and end users. DL-workbench source code is public and we hope it will be useful for vendors of ontology edition tools.

DL-workbench can be viewed both as a meta-model based platform for creating ontology-manipulation tools and as an ontological editor that supports SHIQ description logic (the meta-model is used to describe the structure of SHIQ logic formalism, such as "concept", "object property", "data type", "axiom", "expression", etc.). DL-workbench uses DAML+OIL as persistent format and Racer [Haasler, 2001] as DL reasoner. SHIQ description logic was chosen due to the following main reasons:

- Description logics were taken as the most appropriate language for formal specifications within our domain [Kazakov, 2003].
- SHIQ is a very expressive decidable description logic that has implemented reasoners [Horrocks, 2000]
- SHIQ and DAML+OIL are supported by semantic web community.

The meta-modeling approach to creation of tools was successfully used earlier for products such as IBM WebSphere, Rational Rose and many others. DL-workbench is implemented as a set of plug-ins to IBM Eclipse platform. Eclipse is an emerging open source Java-based environment for creation of project-based tools. Eclipse implements its own portable UI widget library (SWT) that uses native OS calls and is much faster than SUN Swing library. Below we give details of the meta-model kernel and some interesting aspects of DL-workbench.

The major advantage of the DL-workbench is the use of a meta-model. It allows easy-to-use definition of entities and relations of an ontological formalism that has to be used within the workbench.

The meta-model is implemented as a set of Java interfaces for the convenience of use from programming environments. Meta-model is a language that is used for description of ontological formalisms by specifying their elements, structure and invariants (invariants have to be satisfied when instances of these elements are created or modified). The meta-model is basically intended for specifying structural models. Semantics of the meta-model were inspired by Description Logics [Baader, 2003].We tried to keep the meta-model as simple as possible, but powerful enough for many possible needs. The structure of metamodel may be seen in a full version of this article can be found in [DL-workbench, 2003].

The meta-model can be used not only for description of ontological formalisms, but also for description of other data formats that are needed for specific application domain. For example, we use description of Java interfaces expressed with the same meta-model within the software integration domain. Many structural data formalisms can be easily expressed in the presented meta-model.

3 Conclusions

Presence of meta-model for implementation of ontological formalism and connection with other data structures is very important for software integration domain. Our current research for software integration is based on DL-workbench. The ability to have many different views (by namespaces, by taxonomies, by files, graphical view and many others) on the same ontological structure helps a lot in many real cases. From our point of view, working with ontologies must follow the project-oriented paradigm. It's hard to imagine a real industrial ontology that is saved in one file and has no references to other files. However, it is worth to mention that this requirement may be not important for the Semantic Web community. Axioms and logical expressions are extremely important for creating of complete and reasoning-ready ontologies. We've implemented our own GUI of expression editor; however we strongly believe that some deep research must be conducted on ergonomics of expression editor.

Today the DL-workbench is a research prototype and it lacks the stability that is needed for industrial development of ontologies. We use DL-workbench for development of our domain specific extensions and integration with other tools. We use described concepts for creation of extensions of DL-workbench that facilitate our experiments with integration of numerical solvers and creation of "good enough" ontologies verified by reasoner.

References

[Baader, 2003] F. Baader et all, "*The Description Logic Handbook: theory, implementation and applications*", Cambridge University Press, 2003 ISBN 0-521-78176-0

[DL-workbench, 2003] DL-workbench project web site. Online: http://www.opencascade.org/dl-workbench

[Haarslev, 2001] Haarsler V., Moller R., "*High Performance Reasoning with Very Large Knowledge Bases: A Practical Case Study*", In proceedings of IJCAI 2003 conference, 2001

[Horrocks, 2000] I. Horrocks, lU. Sattler, and S. Tobies. "Reasoning with individuals for the description logic *SHIQ"*. LNAI number 1831 pp. 482-496. Springer-Verlag, 2000

[Kazakov, 2003] M. Kazakov, H. Abdulrab, E. Babkin, "*Intelligent integration of distributed components: Ontology Fusion approach*", In proceedings of CIMCA 2003 conference, 2003, ISBN 1-740-88069-2

[Salome, 2003] SALOME project, Online: http://www.opencascade.org/SALOME

The Semantic Object Web: An Object-Centric Approach to Knowledge Management and Exploitation on the Semantic Web

Brian Kettler, James Starz, Terry Padgett, and Gary Edwards

ISX Corporation
4301 North Fairfax Drive, Suite 370
Arlington, VA, 22203
{bkettler,jstarz,tpadgett,gedwards}@isx.com

1. Introduction

The Semantic Web (SW) will dramatically improve knowledge management and exploitation. Current SW tools and applications, however, are largely still document-centric. A complementary approach is to provide an *object-centric index* of documents, databases, and services. Knowledge objects representing entities in the world such as people, places, things, and events are linked into ***Semantic Object Webs™***, which can be navigated, queried, and augmented by software agents (using the DAML/OWL ontologies' underlying semantics) and by humans via visualization tools. These semantically integrated webs provide views of the underlying knowledge space across multiple distributed, heterogeneous sources.

We have been developing one of the first end-to-end tool suites to index, manage, and exploit knowledge via semantic object webs. We are applying these tools in the USAF Research Lab's Effects-Based Operations project, Horus project (sponsored by the DARPA DAML program and the Intelligence Community) and other military and business domains.

2. Technologies and Tools

The **Semantic Object Web (SOW)** approach extends the Semantic Web by focusing on how users and software agents can more easily access and exploit information about specific entities in the world – people, places, events, etc. – that is *semantically integrated* from multiple distributed, heterogeneous sources. The "semantic" part refers to our use of ontologies, formal, shared vocabularies represented in languages such as the DARPA Agent Markup Language (DAML) and the Web Ontology Language (OWL). These ontologies specify the type (class) and properties of entities. Each entity is represented by machine-understandable **Semantic Knowledge Object (SKO)**, an instantiation of one or more ontology classes. SKO's are linked into semantic object webs by ontologically-grounded links (properties), unlike hypertext links on the Web. These links may be browsed by humans or navigated by software agents.

The "integration" part of "semantic integration" refers to the population of SOW's and SKO's from multiple, distributed heterogeneous sources, including web pages, documents, and databases. A SKO can be thought of as encapsulating or indexing information associated with an entity, e.g., a Person such as Saddam Hussein. A SOW is thus a rich, interconnected index of an underlying information space, where the bulk of the information about an entity will reside in various data sources. A SOW indexes information on the Semantic Web, which is typically distributed, or on a individual user's PC, network, or intranet.

Besides being a guide to the underlying information, the index itself contains information that can be used to answer users' questions: e.g., "What are all the countries in Europe?". could be answered using instance data on continents and countries populated from a geographic source. Because the index can capture complex relationships (e.g., financial transactions, terrorist networks, etc.), it can support more specific queries with higher precision results than the keyword-based indices commonly used by most search engines on the web today. *A fragment of a sample SOW is shown in Figure 1.*

Ontologies enable integration. Each data source's underlying structure is mapped to classes and properties in a set of interlinked ontologies. Data can then be rehosted in a SOW index repository – a knowledge base (KB) – at update time or accessed at runtime. We have developed and applied several technologies for defining, populating, exploiting, and maintaining SOW's.

Our logical architecture for the SOW toolkit, shown in Figure 2, includes:

- **Ontology Authoring and Maintenance tools** such as (1) COTS authoring tools such as Sandpiper's Medius™ Visual Ontology Modeler, and (2) XML Schema to Ontology import tools such as ISX's Semagen
- **Ingest tools** to build an index from heterogeneous data sources including (1) automatic markup tools using text entity extractors (e.g., Inxight's Thingfinder™); (2) XML

to OWL import tools such as ISX's Semagen; (3) relational database to OWL import tools; (4) e-mail markup tools; (5) web scrapers; and (6) form-based, manual markup tools.

- **Index Management tools** to provide storage and retrieval from indices using inference. These include (1) a repository for assertions (and metadata) from markup utilizing a KBMS/RDBMS hybrid (e.g., U. Maryland's Parka KBMS and Oracle, or Postgres) for storing assertions from markup; and (2) co-reference determination tools to combine assertions from different sources that pertain to the same entity (SKO)
- **Exploitation tools** including (1) a customizable, ontology-organized knowledge portal supporting SOW/SKO navigation and visualizations (e.g., tree/graph; form-based) for human users; and (2) software agents for automated knowledge discovery that crawl SOW's to find patterns.

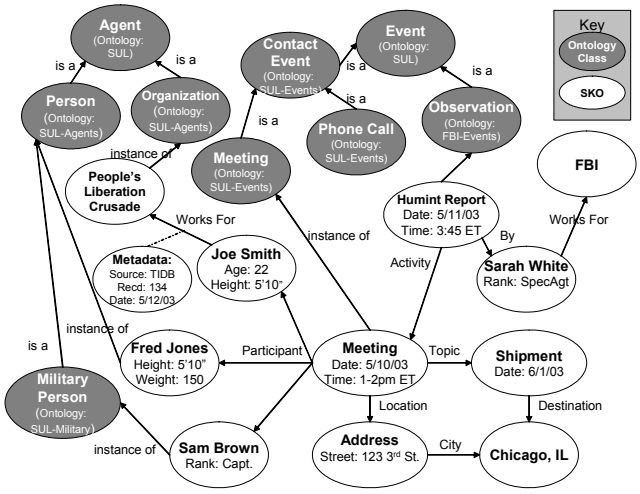


Figure 1. Fragment of Sample Semantic Object Web describing a Meeting Event

3. Applications

We have applied the SOW technologies to a number of military, intelligence, and commercial domains. The **Horus project**, sponsored by DARPA (**DAML Program**) and the Intelink Management Office, is an early adopter of DAML/OWL and chartered with transitioning emerging tools to the Intelligence Community. On the **Effects-based Operations (EBO)** project sponsored by the Air Force Research Labs, we have built tools for military air operations planners to author plans using an ontologically-grounded representation of strategy (strategy templates), effects (and their mechanisms and indicators), and the battlespace (situation entities). This work leverages DAML ontologies and a reasoning service based on the Java Expert System Shell (JESS) and the DAML axioms. Ontologies, axioms, business logic, and instance data is converted into JESS rules and facts for forward chaining inference. This approach allows business rules to augment the constraints specified in the ontology.

Additional applications of the SOW technology include **DARPA's new Semantic Enabling and Exploitation** seedling. A collaborative SOW portal using Groove™ for **DARPA's Terrorism Information Awareness program.**

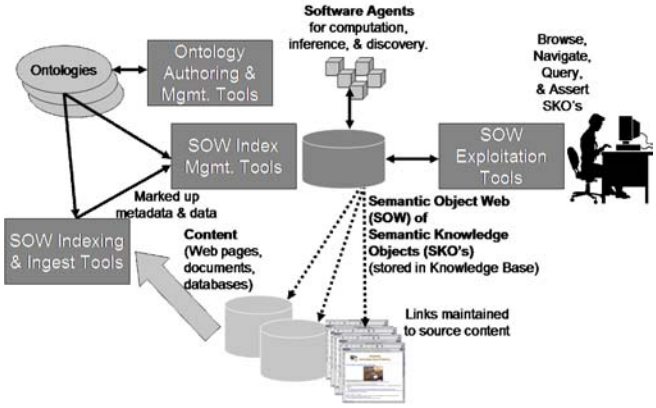


Figure 2. Toolkit for the Semantic Object Web

4. Future Work

We are currently extending our toolkit in a number of directions. The KBMS is being augmented with additional (inferencing) support for OWL. New services will also include increased support for co-reference determination and support for additional rule-based inference using business rules. On the exploitation side, we are exploring more graphical visualizations of SOW's and tools to specify queries graphically and in natural language. New tools will assist with dependency tracking in support of ontology maintenance, in addition to the ontology versioning constructs built into DAML/OWL. Markup tools with increasing automation are planned to aid users in document summarization, with OWL markup as a by-product. We are also developing new kinds of SOW-savvy agents.

5. Acknowledgements

Our SOW work has been funded in part by DARPA under the DAML Program through a contract from BBN Technologies. We particularly wish to thank the current and former DAML Program Managers (Dr. Mark Greaves, Mr. Murray Burke, Dr. Jim Hendler); our IMO sponsor, Dr. David Martin-McCormick, and our Horus teammates: Dr. A. Joseph Rockmore (Cyladian Consulting) and Mr. Don Conklin (BBN). This work derives in part from U. Maryland SHOE Project by Prof. Jim Hendler (U. Md.) and Prof. Jeff Heflin (now at Lehigh Univ.).

Semantic Tuple Spaces: A Coordination Infrastructure in Mobile Environments*

Deepali Khushraj
Nokia Inc.
deepali.khushraj@nokia.com

Tim Finin and Anupam Joshi
University of Maryland, Baltimore County
[finin , joshi]@cs.umbc.edu

1 Introduction

The Tuple Space model was initially conceived for parallel computing in David Gelernter's Linda system[2]. Tuple Spaces offer a coordination infrastructure for communication between autonomous entities by providing data persistence, transactional security and temporal and referential decoupling– properties that make it desirable in distributed systems for e-commerce and ubiquitous computing applications. In most Tuple Space implementations tuples are retrieved by employing type-value matching of ordered tuples, object-based polymorphic matching, or XML-style pattern matching. We present a new approach to retrieve tuples from a Tuple Space. By annotating tuples with semantic descriptions and by making use of a description-logic reasoning engine we can enhance the interaction between independent entities. Semantic description is added to tuples by making use of the DAML+OIL ontology language. Additional inference rules are drawn by making use of a reasoning engine that works well with description-logic based languages. Specialized agents, like the Tuple-Recommender Agent and Task-Execution Agent reside on the space to enhance interaction in mobile environments. Our prototype was integrated with Vigil[1], a framework for intelligent services in pervasive environments.

2 Motivation, Design and Implementation

The representation of a tuple and its retrieval from the space are two significant and slowly evolving features of Tuple Spaces. The simplistic matching that Linda uses is extended in JavaSpaces and other OO space implementations to support polymorphic type matching. In subsequent implementations, support for XML type representation and querying is provided; however, current implementations have certain limitations. An XML representation of a tuple offers syntactic interoperability, but no semantic interoperability. Tuples lack the expressiveness to support extended reasoning by machines. Current implementations do not support inexact matching and there are no standards to share common ontologies.

The Semantic Spaces system is an endeavor to enhance the way tuples are represented and retrieved from the Tuple Space. The key ideas are: to make use of a semantically rich representation to describe the data in a tuple, and to make use of semantic reasoning to search for tuples on the space. The use of semantics enables systems that have been developed independently to coordinate with each other.

At the core of our system is the Outrigger implementation of Sun's JavaSpaces(TM) specification. The "Semantic Tuple Manager" and the "Semantic Tuple Matcher" are the chief components of the system. The "Semantic Tuple Manager" is primarily responsible for validating the semantic consistency of tuples that are added to the space. The "Semantic Tuple Matcher" handles the "read", "take" and "notify" operations.

We introduce the notion of a "Semantic Tuple", which acts as a role marker in our system. System designers can extend this tuple to create application specific tuples. The semantic information is marked up using DAML+OIL. The tuple contains either a URI, which points to the DAML description, or can contain the complete DAML content embedded in it.

SEMANTIC TUPLE MANAGER: When a semantic tuple is written into the space, the semantic description of the tuple is asserted into a description-logic based reasoning engine (like RACER[3]). In addition to the DAML+OIL description, we assert all newly encountered URIs that occur in the namespace of the description. While asserting the description, the reasoner validates class consistency. If the reasoner detects an inconsistency then the description is retracted from the knowledge base and an error is reported. The description can contain both the instance data i.e. A-Box (facts) and the structure of the domain i.e. T-Box (rules), or just the A-Box.

SEMANTIC TUPLE MATCHER: A tuple can be retrieved from the space by performing a "read" or "take" operation. In order to invoke these operations, a semantic template that best matches the consumer's requirement is passed as an input. A predefined DAML+OIL ontology is used to express the tuple template. A snapshot of the semantic template ontology is given in figure 1 . The tuple template has the "hasDegreeOfMatch" property, using which the user can specify the type of matches that are acceptable. Using the "hasField" property the user can specify the list of desired and undesired fields. The "hasFieldWithGroup" property allows the user to specify a "FieldGroup", which is essentially a bunch of tuple fields. Template matching is done by posing queries to the reasoning engine. The following sequence of steps is performed for every "TupleField" of the tuple template:

*This work was partially supported by the DARPA contract and was done while the first author was at UMBC.

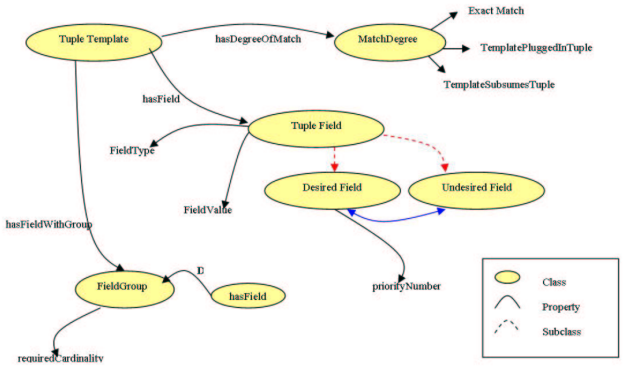


Figure 1: The ontology of the tuple template

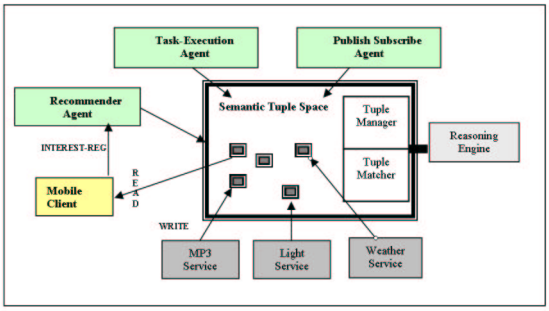


Figure 2: Semantic Tuple Space with specialized agents. A mobile client communicates with the space directly, or through agents.

1. The first step is to find an exact match, which occurs when a tuple and a template are equivalent [DL]. In our prototype, a template is considered equivalent to a tuple if all the "TupleField" properties (including the "DesiredField" and "UndesiredField" properties) specified by the template "exactly match" the description of a tuple. If no match is found further matches are carried out based on the preferred "hasDegreeOfMatch" property specified by the template.
2. If the preferred degree of match has the value "TemplatePluggedInTuple", all templates that are subsumed by tuples are considered as valid matches.
3. The "TemplateSubsumesTuple" degree accounts for cases where a template subsumes a tuple.
4. If none of the aforementioned cases are satisfied then the match results in a failure and no tuple is returned.

At each step a weight is assigned to every tuple that gets selected, based on its degree of match. If an undesired field is present in the tuple, then there is a clash of interest and the tuple is assigned a negative weight. After processing all the tuple fields in the template the tuple with the highest weight gets selected.

In order to demonstrate the working of the Semantic Space infrastructure in pervasive environments we used the Vigil framework to create clients and services. To enhance the utility of our system, we introduce three specialized agents that reside on the Semantic Space: Tuple Recommender Agent, Task Execution Agent and Publish Subscribe Agent. These agents make use of two extensions of the SemanticTuple namely, ServiceTuple and ObjectTuple. A ServiceTuple is used to advertise services on the Space, whereas an ObjectTuple is primarily meant for the Publish-Subscribe Agent.

TUPLE RECOMMENDER AGENT: Clients register their interests with this agent to get notified of all service tuples that match their interests. The interest is specified by the client using a pre-defined ontology. The agent unburdens the client by handling user movement and disconnections that occur due to varying QoS.

TASK EXECUTION AGENT: This agent is closely integrated with the Vigil infrastructure. Clients register atomic or composite Vigil tasks with this agent, and the agent tries to execute the registered tasks on behalf of the client. The client specifies the task using an ontology with control constructs such as "Sequence", "Concurrent" and "Unordered". The user can also specify the start time and stop time of an atomic task for tasks that do not require immediate execution.

PUBLISH-SUBSCRIBE AGENT: This agent dynamically delivers data/events to subscribed users. In addition to the sharable data, the published object tuple contains a list of subscribed users and a semantic description to describe the content of the object. The agent polls the space periodically to look for tuples that a user in its domain is subscribed to.

Semantic inferencing was done using RACER. The main difficulty we faced with RACER was that it created a new Knowledge Base for every DAML+OIL file, which makes it difficult to load additional files specified in the namespace. However, it works very well with DAML+OIL because it has built-in constructs for description logic languages. The inference classification provided by RACER is particularly useful for deducing the class of a tuple. Similar to a rule-based expert system's forward-chaining mechanism, RACER supports a publish-subscribe mechanism. This feature is particularly useful when performing operations like "notify". The "notify" of a tuple space directly maps to RACER's publish-subscribe mechanism.

In the future we plan to introduce semantics to express the functionality of methods. We would also like the Task-Execution agent to use a planner to execute composite tasks. Security can be enhanced by using the DAML+OIL policy ontology.

References

[1] Lalana Kagal et al. Vigil: Enforcing security in ubiquitous environments. In *Grace Hopper Celebration of Women in Computing 2002*, 2002.

[2] D. Gelernter. Generative communication in linda. *ACM Transactions on Programming Languages and Systems*, 7(1):80–112, 1985.

[3] Volker Haarslev and Ralf Möller. RACER system description. *Lecture Notes in Computer Science*, 2083:701ff, 2001.

Towards Interactive Composition of Semantic Web Services

Jihie Kim and Yolanda Gil
Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90292, U.S.A.
jihie@isi.edu, gil@isi.edu

Abstract

We are developing a framework for interactive composition of services that assists users in sketching their task requirements by analyzing the semantic description of the services. We describe the requirements that an interactive framework poses to the representation of the services, and how the representations are exploited to support the interaction. We also describe an analysis tool that takes a sketch of a composition of services and generates error messages and suggestions to users to help them complete a correctly formulated composition of services.

1 Introduction

Existing approaches to generate compositions automatically are limited in their use when explicit goal descriptions are not available and when users want to drive the composition process, influencing the selection of components and their configuration. The goal of our work is to develop interactive tools for composing web services where users sketch a composition of services and system assists the users by providing intelligent suggestions.

Interactive service composition poses additional challenges to composing services. Users may make mistakes and the system needs to help fix them. Also, user's input is often incomplete and may even be inconsistent with existing service descriptions. In order to help users in this context, we have developed a framework for providing strong user guidance by reasoning on the constraints associated with services. The framework is inspired by our earlier work in KANAL to help users construct process models from pre-defined components that represent objects and events [Kim and Gil, 2001]. In our previous work, we have built a tool that performs verification and validation of user entered process models by exploiting domain ontologies and event ontologies. In this work, we take simple service descriptions (in WSDL) and augment them with domain ontologies and task ontologies that address various constraints in the domain. Our analysis tool then use these ontologies in examining user's solutions (i.e., composition of services) and generating error messages and suggestions to correct the errors. We believe that as ontologies become richer, the tool can provide more direct and focused suggestions.

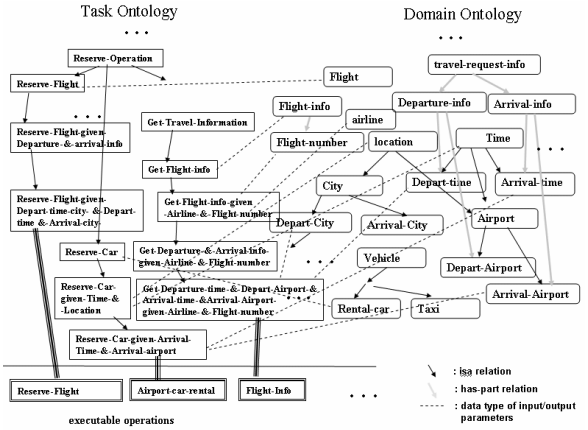


Figure 1: Task Ontology and Domain Ontology.

2 Approach

Our approach is to provide strong user guidance through constraint reasoning, as described above. First we take definitions of services and analyze relations between service operations in the composition sketch based on their input and output parameters. We then detect gaps and errors from the analysis including missing steps, missing connections, incomplete steps, etc. Finally we produce suggestions based on the problem type and context. In performing the analysis, we assume a knowledge rich environment where services and their operations are described and related in terms of domain objects. (We are investigating some ways to exploit existing ontologies that are available on-line.) Currently we are exploiting two types of ontologies: domain term ontology and task ontology. That is, data types are represented using domain objects, and their task types are defined in terms of their input and output data types. Figure 1 shows such ontologies that we are using in a travel planning domain. For example, a task type Reserve-Car-given-Arrival-Time-&-Arrival-Airport represents a service operation that has Arrival-Time and Arrival-Aiport as the input and Flight-Info as the output. Its parent Reserve-Car-given-Time-&-Location represents a more general class of operations including Reserve-Car-given-Arrival-Time-&-Arrival-Airport. Note that because the system has

an ontology of operation types that describes high-level task types as well as specific operations that are mapped to actual operations, users can start from a high-level description of what they want without knowing the details of what operations are available. We often find that users have only partial description of what they want initially, and our tool can help users find appropriate service operations by starting with a high-level operation type and then specializing it.

The tools we built is called CAT (Composition Analysis Tool). CAT's analysis is driven by a set of desirable properties of composed services. Given a sketch of a service composition and a user task description (i.e., a set of initial input and expected results), CAT checks if (1) all the expected results are produced, (2) all the links are consistent, (3) all the input data needed are provided, and (4) all the operations are executable (there are actual operations that can be executed). In addition, it generates warnings on (5) unused data and (6) unused operations that don't participate in producing expected results. Given any errors detected, CAT generates a set of specific fixes that can be potentially used by the user. The following shows the general algorithms.

- **Checking Unachieved Expected Results**:
  Detect problem: for each expected result, check if it is linked to an output of an operation or directly linked to any of the initial input (i.e., the result is given initially).
  Help user fix problem:
  1. find any available data (initial input or output from introduced operations) that is subsumed by the data type of the desired result, and suggest to add a link
  2. find most general operation types where an output is subsumed by the data type of the desired result, and suggest to add the operation types.
- **Checking Unprovided Data**:
  Detect problem: for each operation introduced, for each input parameter of the operation, find if it is linked to any (either to the initial input or to some output from introduced operations).
  Help user fix problem:
  1. find any initial input data or output of operations that is subsumed by the desired data type, and suggest to add a link.
  2. find most general operation types where an output is subsumed by the desired data type, and suggest to add the operation types.
- **Checking Inconsistent Links**:
  Detect problem: for each link between data types, find if the type of the data provider is subsumed by the type of the consumer.
  Help user fix problem:
  1. find most general operation types where an output is subsumed by the type of the consumer and an input subsumes the the type of the provider, and suggest to add the operation types.
- **Checking Unexecutable Operation**:
  Detect problem: for each operation type introduced, check if there is an actual operation of that type that can be performed.
  Help user fix problem:
  1. find a set of qualifiers that can be used to specialize it and suggest to replace the operation type with a more special one base on the qualifiers.
  2. find the subconcepts of the task type in the task ontology and suggest to choose one of them.
- **Checking Unused Data**:
  Detect problem: for each initial input data type and the output from the introduced operations, check if it is linked to an operation or an expected result.
  Help user fix problem:
  1. find any unprovided data or unachieved results that subsumes the unused data type, and suggest to add a link.
  2. find most general operation types where an input subsumes the unused data, and suggest to add the operation types.
- **Checking Unused Operation**:
  Detect problem: for each operation introduced, check if its output or any output from its following operations is linked to an expected result.
  Help user fix problem:
  1. suggest to add a link to connect the operation

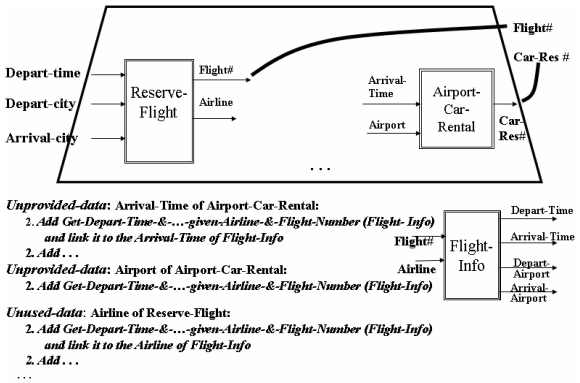


Figure 2: Travel Planning: CAT finds errors and help users fix them.

Figure 2 shows a process of composing services for a travel planning. The user wants to reserve a flight first and then reserve a car based on the reserved flight. Currently two input parameters of Reserve-Car operation, Arrival-Time and Airport, are not linked yet. CAT points that both of them can be potentially linked if the Flight-Info operation is added in between, since it produces data on Arrival-Time and Airport (Depart-Airport and Arrival-Airport) given an Airline and a Flight-number. This addition will also resolve the warning of unused data (Airline of Reserve-Flight). In this case, as the system has richer ontology of trips so that the airport of the Airport-Car-Rental actually means the Arrival-Airport, then the suggestions will become even more specific.

3 Current Status

The current implementation of CAT has a text-based interface for reporting errors and suggestions. We have applied CAT in composing computational pathways to put together end-to-end simulations for earthquake scientists where the problem is to analyze the potential level of hazard at a given site. The preliminary tests show that CAT can help users formulate correctly formulated pathways by pointing specific ways to fix errors. Our plans for future work include development of graphical user interfaces for CAT, dynamic generation of task ontologies from service descriptions, and incorporation of automatic service composition approaches.

References

[Kim and Gil, 2001] Jihie Kim and Yolanda Gil *Knowledge Analysis on Process Models.* Proceedings of IJCAI-2001.

Systematization of Nanotechnology Knowledge Through Ontology Engineering
- A Trial Development of Idea Creation Support System for Materials Design based on Functional Ontology -

Kouji Kozaki, Yoshinobu Kitamura and Riichiro Mizoguchi
The Institute of Scientific and Industrial Research, Osaka University
8-1 Mihogaoka, Ibaraki, Osaka, 567 -0047 Japan,
{kozaki,kita,miz}@ei.sanken.osaka-u.ac.jp

Abstract

The research of nanotechnology is extended in various domains, and each domain intertwines with each other closely. The objective of our research is to systematize fundamental knowledge using ontology engineering to fill the gap between materials and devices through establishment of common concepts across various domains. We also aim at building a creative design support system using the systematized knowledge. In this paper, we outline a prototype of a support system for innovative nanotech-made device design based on functional ontology and functional decomposition tree which helps developers' creative design processes.

1 Introduction

The research of nanotechnology is extended in various domains, and each domain intertwines with each other closely. Therefore, sharing the knowledge in common among different domains contributes to facilitate research in each domain through cross fertilization. In this background, the Structuring Nanotechnology Knowledge project, which is a NEDO (Japanese New Energy and Industrial Technology Development Organization) funded national project, has been carried out. The goal of the project is to build a material-independent platform for supporting development of innovative nano-materials. It is not a database, a set of simulation tools or a knowledge base, but is an integrated environment composed of structured knowledge supported by advanced IT.

Among many factors, the authors have been involved in building ontology of nanotechnology and its application to knowledge systematization. The key issues of knowledge structuring include how to harmonize different terminologies and viewpoints of the respective domains and how to interface end users with the platform. Ontology of nanotechnology plays a role of glue for seamless connection between different domains and between users and the platform, since it provides us with a conceptual infrastructure of nanotechnology and with a unified framework in which functional knowledge for conceptual design of nanotechnology-made materials and devices and their realization processes.

In this paper, we outline a prototype of a support system for innovative nanotech-made device design based on functional ontology and functional decomposition tree which helps developers' creative design processes.

2 A System for Supporting Creative Design of Nanomaterials

Aiming at bridging required functions stated by engineers in industries and basic functions (or quality) and at facilitating the creative design, systematization of function achievement ways in a particular domain and development of a support system of functional design of materials are currently conducted in parallel (Figure.1).

2.1 Idea Creation Support by Providing Alternative Function Achievement ways

In general, a function is achieved by performing multiple sub-functions. For example, a function of incandescent lamp "emit light" is achieved by sub-functions "apply a current to a filament", "the filament heats up", and "emit light". The achievement is supported by a physical principle and/or structure of the device or materials which is conceptualized as ***Function achievement way***. (In this example, the principle is "radiation".) The decomposition is continued concerning each sub-function until it reaches a basic function or quality of a material to eventually form

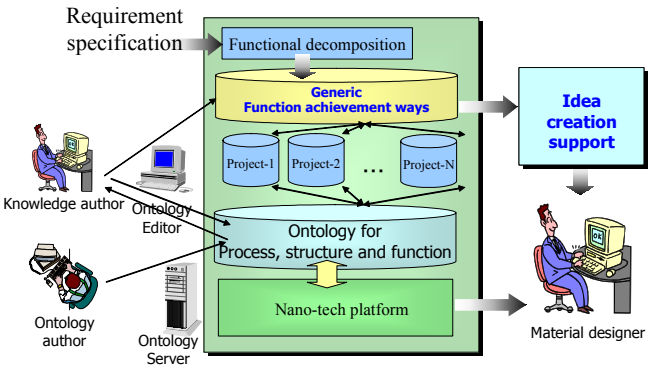


Figure.1. Idea creation support system for materials design

a function decomposition tree for each device/material. In this way, the gaps between required functions and basic functions (or quality) are bridged. There exist multiple ways of functional decomposition so that the computer can help device/material designers to help their design process by giving possible alternative ways stored in a function achievement way server.

2.3 Development of Functional Ontology and Idea Creation Support System

We developed a functional ontology containing such concepts that are used in describing requirement specification for devices together with a set of functional decomposition knowledge which bridges the gap between requirement specification of a device and fundamental properties of materials.

Then we stored some common knowledge represented based on the ontology in the ontology server and investigated the performance of the ontology server. And we built a creative design support system based on the functional ontology and a formalism of functional decomposition tree. It is considered as a prototype system for an intelligent support system for designing nanotech-made materials.

Figure.2 shows a snapshot of the system. It supports the user's creative design process by the following steps:

(1) The system displays the lists of functions, and the user selects one function as a requirement function
(2) The system searches the function achievement ways which can realize the selected function and show the results.
(3) The user selects an achievement way.
(4) Then the system expands the functional decomposition tree based on the selection.
(5) Continue functional decomposition of sub-functions

Our system is developed as a web-based application which is connected our ontology sever. And we realized the cooperation mechanism with other subsystems developed by other group in our project and confirmed it works well using the result explained in the item.

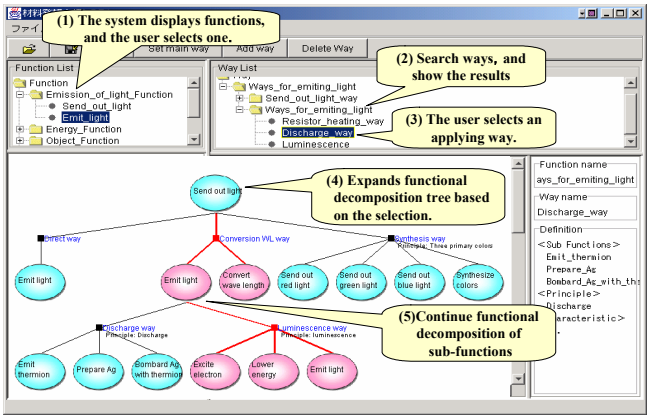


Figure.2. a snapshot of Idea creation support system

3.3 Advantages of Our System

The system supports the idea creation by allowing to replace alternative ways of function achievement, and the user's selection results are preserved. The selection from alternatives is regard as an explication of design decisions so that recording past design processes might be effective to facilitate idea creation. Moreover, the function decomposition tree is very useful to compare between past designs. And it is effective analysis of patents because improvement factors are expressed explicitly as the replacement of ways.

3 Concluding Remarks and Future work

In this paper, we summarized an idea creation support system for materials design based on the functional ontology and a formalism of functional decomposition tree as a part of systematization of nanotechnology knowledge with ontology engineering. Improvement of the prototype system through the applications to several examples with augmentation of the ontology and knowledge is the important future work. It is based on the evaluation of them and includes the following research items:

- Design of upper ontology for nanotechnology
- Augmentation of the function achievement way knowledge for function decomposition tree building
- Improvement of the nanotech-ontology server.

Acknowledgments

This work was supported by the New Energy and Industrial Technology Development Organization (NEDO).
We are grateful to, Dr. H. Tanaka and Dr. T. Nakayama for their valuable discussions.

References

[Kitamura *et al*., 2003] Kitamura, Y. and Mizoguchi, R.: Ontology-based description of functional design knowledge and its use in a functional way server, *Expert Systems with Applications*, Vol.24, pp.153-166, 2003.

[Kozaki et al., 2000] Kozaki, K., et al: Development of an Environment for Building Ontologies which is based on a Fundamental Consideration of "Relationship" and "Role":PKAW2000, pp.205-221, Sydney, Australia, December, 2000

[Kozaki et al., 2002] Kozaki, K., et al: Hozo: An Environment for Building/Using Ontologies Based on a Fundamental Consideration of "Role" and "Relationship", Proc. of the 13th International Conference Knowledge Engineering and Knowledge Management(EKAW2002), pp.213-218, Sigüenza, Spain, October 1-4, 2002

Personal Agents on the Semantic Web *

Anugeetha Kunjithapatham, Mithun Sheshagiri, Tim Finin, Anupam Joshi, Yun Peng
Department of Computer Science and Electrical Engineering
University of Maryland, Baltimore County
Baltimore MD 21250 USA
{anu1,mits1,finin,joshi,ypeng}@cs.umbc.edu

1 Introduction

The Semantic Web is a vision to simplify and improve knowledge reuse and dissemination on the world wide web. Efforts are underway to define the format and meaning of the language of such a Semantic Web that could serve both humans and computers. The EU-NSF strategic workshop report on the semantic web identifies 'the applications for the masses such as intelligent personal assistants' as one of the key applications enabled by the semantic web. Personal assistants gather and filter relevant information and compose it into a coherent picture with regard to the user's preferences. An intrinsic and important pre-requisite for a personal assistant or rather any agent is to manipulate information available on the Semantic Web in the form of ontologies, axioms, and rules written in various semantic markup languages. ;the means of information gathering being centralized (event notification services) or de-centralized (peer agents).In this paper, a model architecture for such a personal assistant, that deals with real-world semantic markup is described.

2 Personal Agents (PA) and the Semantic Web

As the amount of information grows on the web, the average user is overwhelmed by the cognitive load involved in making decisions and making choices. Our endeavour involves delegating some of the tasks to the PA thereby helping the user make better use of his time. We demonstrate this concept using a talk notification service with a human and an agent interface;an illustration of this concept is provided in the figure below. The tasks performed by our PA involve information filtering and filtering through peer collaboration. The PA has a model of the user's preferences expressed in DAML+OIL[daml.org, 2001]. The use of DAML helps the PA leverage semantic inferencing. The PA interacts with the user using MS Outlook Calendar and schedules talks based on criteria like user's interest in the talk, user's availability, and recommendation from peers. The PA makes use of a host of third party services that aid the agent in its decision making. These third parties are wrappers that convert unstructured information on web pages (like mapquest) to structured facts in the agent's KB. We feel that in the near future more and more services will be offered that cater to machines rather than humans. We use DAML+OIL as the inter-lingua for communications among PAs and between agents and service providers. We have used the JADE agent framework to build our agents. Please refer to the extended version of this paper[1] for a more detailed description.

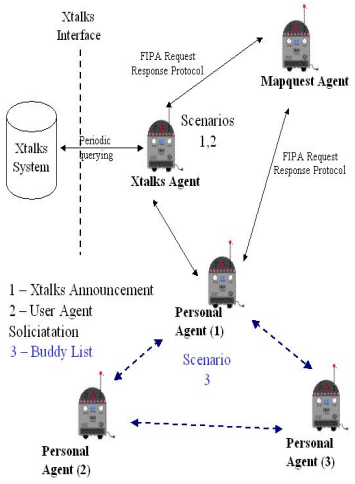


Figure 1: **Multi-Agent Scenario and Interactions**

3 Reasoning in PAs

We use the Java Expert System Shell (JESS) for storing knowledge and inferencing. The PAs beliefs of the world are stored as VSO triple-based facts in JESS. DAMLJessKB[Kopena and Regli, 2003] provides a set of axioms that are used for reasoning over RDF[Lassila and Swick, 1999] and DAML+OIL[w3.org, 2001]. As new facts are entered into the KB, rules corresponding to these axioms fire and new facts are asserted into the KB. This is the mechanism for inferencing in our PAs. Apart from the DAML axioms, user's preferences are also expressed as rules in JESS. Certain rules also fire off events that are captured and an appropriate indication is made to the user. For example, when all conditions for scheduling a talk are met and the corresponding rule

*This research was supported in part by DARPA contract F30602-97-1-0215.

[1] http://users.ebiquity.org/docrepos/2003/paper/PersonalAgents-ISWC03.pdf

fires, this event is captured and the Bridge2Java API is used to schedule the talk in the user's Outlook Calendar.

4 Interaction with Peer PAs

The PA also has an implicit module that enables it to interact and collaborate with peer agents. The PA, on receiving the talk notification, consults a list of PAs that are regarded as buddies. A FIPA-ACL[FIPA, 2000] message is sent to peer-PAs and as per the query-ref FIPA interaction protocol, the PA receiving the message is obliged to send back a reply. The content of these messages are DAML+OL assertions. The peer PAs are determined through a buddy-list that the user maintains. We also have developed a discovery mechanism for discovering buddies. Our mechanism makes use of a popular search engine to locate the homepage of the owner of the peer agent. Inspite of the vast size of the web, it is easy for a search engine engineered to index billions of pages to locate the homepage of a person with reasonable web presence. For the sake of simplicity, we refer to the person initiating the discovery as the user and the person being located as the owner. A HTML META tag in the homepage points to the owner's profile in DAML. The profile among other things includes the location (ip:port) of owner's agent. A FIPA *subscription request* is sent to the owner's agent. The owner's agent on receiving the request sends its owner an email. This mail is in the form of a HTML with embedded scripts. The e-mail contains user's details and hyperlinks to capture the owner's decision. In response to the owner's response a corresponding FIPA *inform* message is sent back to the user's agent which might/might not update the buddy list based on the owner's decision.

An agent can also pose queries to peer agents. We have developed a querying mechanism that uses a combination of DAML Query Language(DQL)[DQL, 2002], JESS *defqueries* and FIPA agent communication protocols. DQL enables us to describe queries in DAML, and FIPA protocols provide the transport mechanism for the queries by defining the interaction between the agents involved. The query is framed as a set of PSO triples with unbound variables and sent to one of the buddy agents as a FIPA *query-ref* message. The receiving agent on receiving the query, converts the triples into a JESS *defquery* and fires it in its KB. The triples acquired by firing the query are packed into multiple FIPA *inform-result* and sent back to the querying agent. For more information on DQL and our extensions to it, please refer[Sheshagiri and Kunjithapatham, 2003].

5 Trust and Privacy in PAs

The PA possesses a wide range of knowledge including some persistent data and dynamic data acquired through notification services, interaction with peers and reasoning performed at various stages. Such information could be of great use to the peer PAs and hence it would be worthwhile to share it with interested parties. While interacting with a peer for sharing information, the PA will have to determine if the requested information exchange can be shared. It may be impossible for the PA to come up with a decision emulating it's user's choice of action in such situations; but, we believe that a simple and straight-forward mechanism to determine the credibility of the requesting party and the nature of information requested would help the PA to take a user desirable decision. We describe below a mechanism that we propose:

In our model architecture, the data in the PA's Knowledge base is categorized as *sharable*, *non-sharable* or *sharable with the user's consent*. Personal information, past appointments and class schedule information of the user are classified as *sharable* facts. *Non-sharable facts* consists of confidential information. Facts such as the current location, future appointments etc. are categorized as facts *sharable with user's consent*. The PA on receiving a query responds based on the type of information requested. If the information is categorized as *sharable with user's consent*, the PA sends a mail to its user about the request and replies according to the user's response.

Additional rules based on the user's relationship with the requestor and the requester's role are also defined. To enable the PA to identify the appropriate rules to execute, we have come up with a set of rules to identify the order of thier execution. Some of the implemented rules based on the relationship with the requestor are as follows: (1)*If the requestor is a friend and not a family member - share only information marked as sharable.* (2)*If not a friend/family member but Advisor- share SSN, schedule information.* (3)*Peer agents belonging to family members have access to all information* A *cache* component has been designed to keep track of rejected queries, and to allow the PA to determine the urgency of the query, possibly based on the number of times the PA got the same query.

6 Conclusion

We have demonstrated a Personal Agent application that leverages the capabilities of semantic web languages and agent technology to perform some of the user's tasks. Automation achieved through applications like this can help the user manage his/her time more efficiently.

References

[daml.org, 2001] DAML+OIL Specification daml.org. http://www.daml.org/2001/03/daml+oil, 2001.

[DQL, 2002] DQL. Daml query language http://www.daml.org/dql, 2002.

[FIPA, 2000] FIPA. http://www.fipa.org/, 2000.

[Kopena and Regli, 2003] Joe Kopena and William Regli. DAMLJessKB: A tool for reasoning with the semantic web. *IEEE Intelligent Systems*, 18(3):74–77, 2003.

[Lassila and Swick, 1999] Ora Lassila and Ralph Swick. Resource description framework model and syntax specification http://www.w3.org/rdf, 1999.

[Sheshagiri and Kunjithapatham, 2003] Mithun Sheshagiri and Anugeetha Kunjithapatham. A fipa compliant query mechanism using daml query language (dql) http://www.cs.umbc.edu/ finin//papers/dqlfipa.html, 2003.

[w3.org, 2001] DAML +OIL Reference Description w3.org. http://www.w3.org/tr/daml+oil+reference, 2001.

Ontology based chaining of distributed Geographic Information Systems

Rob Lemmens

Department of Geo-information Processing
International Institute for Geo-information Science and Earth Observation (ITC)
P.O. Box 6, 7500 AA Enschede, The Netherlands
e-mail: lemmens@itc.nl

1. Geographic Information Systems as components

For the last decade, Geographic Information Systems (GIS) have provided planners and geo scientists with tools to analyse, maintain and present geo spatial information (information that is, in one way or the other, referenced to the earth surface). In the early days of GIS, its software systems were sold as monolithic systems. As the software became more mature, the systems were offered as modules containing a module with basic functionality and a variety of plug-in modules with extended functions. Main software producers came to realise that specific users who wanted to customise their systems needed a development environment with smaller system building blocks (components).

Today, a product like ESRI's ArcObjects provides the software elements to create an entire GIS. However these building blocks in themselves do not provide executable GIS analysis capabilities, they have to be assembled by a programmer. Unfortunately, these GIS ob jects' are of little use to the common GIS end-users whose interest is to apply certain common GIS processing functions to give solution to their geographic problems. GIS applications can be characterised by the wide variety of datasets (themes and data structure) and the often complex, but reusable operation-data chains. Many GIS applications, in particular in environments that require ad hoc queries, can greatly benefit from the use of interoperable components. To enable on demand component chaining we need data components and software components that are well defined and well described in terms of functionality, together with a user interface that facilitates the user-interpretation of these descriptions. Component-based applications have been around for some time, but their deployment in GIS is still in its infancy. This can be explained by the fact that GISs have to deal with complex (spatial) data types and software manufacturers tightly couple their functional parts with internal data structures.

2. Supporting data-operation connectivity, a multi-layer approach

In order to construct a component chain, users seek for *meaningful* combinations of data and process components. The term meaningful can be interpreted on different abstraction levels of connectivity between data and operation and depends on possible other requirements in the component chain. For example, suppose we want to calculate the shortest route between two house addresses and we make use of a chain of distributed operations. There can be different reasons why a typical GIS operation such as an address matcher[1], as first part of the chain, would not meaningfully operate on a certain address dataset. First the address matcher may use only street names (and no house numbers) as reference entities. Thus the geographic resolution is not appropriate for this component chain. Further the, address matcher may output the coordinates in a coordinate system that is unknown to the subsequent components of the chain. Generally speaking, we can distinguish three levels of abstraction, namely conceptual model, data structure and data format, where connectivity appears on all three levels. In this layered approach an address appears respectively as a concept (meaning of an address as interpreted by the information provider), its representation in a database as field(s) and the actual field values as output in a string or file.

In a more generalised geographic point of view, the address is a possible absolute location of a phenomenon as depicted in figure 1. In order to identify the connectivity between an operation and a dataset, we need descriptions on these different levels. Whether descriptions are needed on all levels depends on the context of the component chain. For example, if we would like to convert a dataset from one geographic coordinate system to the other, we do not need to know whether we deal with street features or houses (information at the conceptual level). A mediator identifies

[1] An address matcher finds the location coordinates (e.g. X,Y) of an address (street address with or without house number).

potential connectivity, based on dataset and operation descriptions, referred to as metadata (see figure 2).

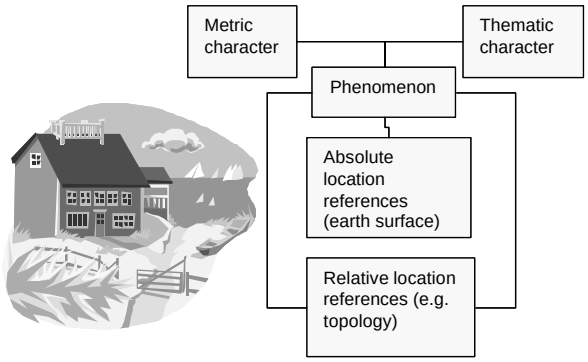


Figure 1. Generalised conceptual data model of a phenomenon in geographic space.

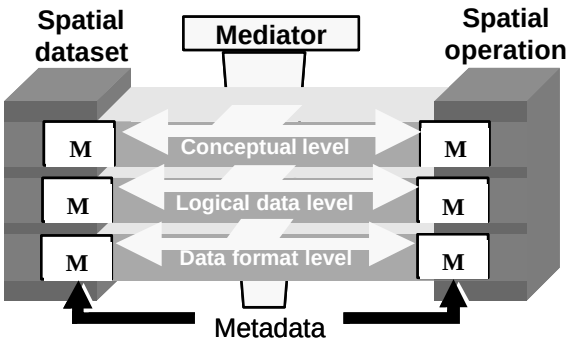


Figure 2. Connectivity layers and metadata, after [Lemmens *et al.*, 2003]

3. The role of geo ontologies

Descriptions of data and operations have to be measured against a reference frame of known artefacts and for the sake of automation such a reference frame must rely on machine processible information.

Reference framework topic	**Starting points**
Geographic coordinate systems	EPSG classification [EPSG, 2003]
Atomic and composite operations	ISO 19119 [ISO, 2002]
Location identifiers of geographic phenomena such as *address*	This research
Geodata structures	Geography Markup Language [OGC, 2002]
Thematic types of geodata (e.g. land use classification)	Domain specific taxonomies, e.g. CORINE land-cover classification [Bossard *et al.*, 2000]

Table 1. Reference frame topics for geo ontologies

Currently, the emerging Semantic Web provides several techniques to handle such reference frames with XML based ontologies. Table 1 indicates important reference frames for geo-information based processing that are partially existent, however not implemented yet in processible ontologies.

This research has initiated the creation and testing of a limited address ontology as partly depicted in figure 3.

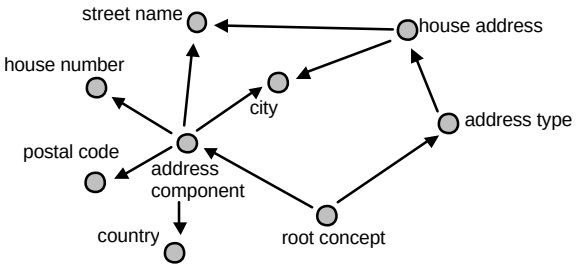


Figure 3. Graphic representation of a part of the *address* ontology

The address ontology is used in a natural disaster event scenario where multiple users need to identify the danger zone around their current location by providing an address. Depending on the kind of address they provide (e.g. with or without house number) a dedicated address matcher is selected. In the descriptions of the address matching components, the address ontology is referenced in RDF triples giving a conditional statement that clarifies which address type is used.

References

[Bossard *et al*., 2000] M. Bossard, J. Feranec and J. Otahel. *CORINE land cover technical guide– Addendum 2000* Technical report No 40 European Environment Agency Copenhagen Denmark. Available at http://www.eea.eu.int

[EPSG, 2003] European Petroleum Survey Group. *Geodesy Parameters Version 6.3 data model and data set*. February 2003. Available at http://www.epsg.org.

[ISO, 2002] *ISO Draft International Standard 19119 Geographic Information Services*. ISO, Geneva, Switzerland.

[Lemmens *et al.*, 2003] Lemmens, R., de Vries, M., Aditya, T. (2003). *Semantic Extension of Geo Web Service Descriptions with Ontology Languages.* In Proceedings of the 6th AGILE. April 24th – 26th, 2003, Lyon, France.

[OGC, 2002] *OpenGIS® Geography Markup Language (GML) Implementation Specification, version 2.1.1.* Available at http://www.opengis.net/gml/02-009/GML2-11.pdf

A Proposal for Web Information Systems Knowledge Organization

Miguel-Ángel López-Alonso

School of Library and Information Science, Extremadura University, 06071 Spain (E.U.)
malopalo@alcazaba.unex.es

María Pinto

School of Library and Information Science, Granada University, 18071 Spain (E.U.)
mpinto@ugr.es

1 Introduction

The authors reflect on the type of processing model that might combine the analytical advantages of the human mind with the computer's potential for statistical calculations.

They depart from a subjective, multiparadigmatic consideration of semantic problems in order to approach pragmatic problems in Knowledge Organization from Web Information Systems.

2 Structural Components of Information Processing

On the basis of the similarities between argumentation and the automatic processing of knowledge, they attempt to relate *syllogistic deductive reasoning* with *information interpretation schemata* in such a way that an Integrated Model for Knowledge Management from Web Information might be developed. It would locate information by virtue of its significance, in view of the concepts defined by the user or extracted from a given Knowledge Database (e.g. hyper textual ontologies).

Computer comprehension of natural language means bi-directional communication. It leads researchers on a more complete background study of the linguistic levels of the text (morphological, syntactic, semantic or inductive) and of the conceptual techniques that detect pragmatic considerations (heuristic or inferential). Yet this communicative process presents two fundamental problems: one is the ambiguity of natural language; and the other, the lack of powerful "model interfaces" to translate the query from human natural language to the computer system languages [Gaizauskas *et al.*, 2001].

With the arrival of more powerful computers and big corpora in digital format, *novel approaches to Documentary Content Analysis and to Scientific Discourse can be seen.* Most new models revolve around the automatic extraction of different linguistic forms according to their representative multi-functionality: simple morphemes, nominal or phrasal syntagmas, or full paragraphs. There have even been attempts at in-depth semantic analysis to locate, through other documents, related knowledge not contained explicitly in the fragments of the original text, by means of the statistical study of the associative relationships among concepts, or "cross language" [Foltz *et al.*, 1998].

Noteworthy, among the statistical approaches to the semantic analysis of discourse, is Latent Semantic Analysis (LSA) [Deerwester *et al.*, 1990]. Its effectiveness has been contrasted by psychometric tests. This variant of Vectorial Space uses large frequency matrices for documentary representation, and applies *matricial decomposition* and *dimensional reduction* to the term-document vectorial space (by associating descriptors to the most meaningful passages of texts). The units of information are compared to one another in order to determine the *factor analysis* (correlation meaning) on the basis of synonymous, antonymous, hyponymous, plural, etc. terms that may be used in a similar way in different contexts. Thus, LSA derives contextual occurrences from the automatic affinities of reading and from superficial literal co-reference, through mechanisms analogous to those of the contextual analysis of users.

3 Integrated Model for Knowledge Management from Web Information

The proposed Integrated Model for Knowledge Management from Web Information uses *slightly structured associative networks* to represent information, while a *general and multivalent system of ontologies* is used for its organization. The framework for applying the model would harmonize the information system with user preferences, by means of the development of powerful conceptual tools integrated in user interfaces.

1) According to Kintsch [Kintsch, 1988], meanwhile, this type of fixed structure is not flexible enough to adapt quickly to the demands of a contextualized documentary setting in constant evolution.

The system for representing knowledge he proposes is an associative neuronal network with a minimum of organization: nodes of concepts or fragments of the original text, with no pre-established structure, enriched by feedback from the context of the task at hand. "The arguments of a proposition are concepts or other propositions". This implies that the latter are not expressly defined in an "ad hoc" knowledge database; rather, their meaning may be elaborated on the basis of their position in the network. The immediate associates and semantic neighbors of a node constitute the nucleus of its meaning, so that the full meaning can only be arrived at by exploring a node's relationships with the rest of the nodes of the network [Haenggi and Kintsch, 1995].

In this context, Latent Semantic Indexing (LSI) can be proposed as a model for representing meaning, understood as the semantic content of the documentary terms —in addition to its utility as an automatic tool for analyzing the semantic content of digital documents, the aforementioned LSA. This model allows, the generator using one's mental model, applying newly pruned conceptual structures in the context of each application, stemming from the main structural network.

What they are proposing as *a framework for the representation of information* is a set of slightly structured associative networks in which the conceptual units represented by nodes would be semantic entities, and the relationships represented by links would be associations of entities [Chung *et al.*, 1998]. In hybrid systems like this, knowledge databases would be treated as text collections linked among them by means of indexing, supported by *the Ontological Organizational Space* proposed in the following section.

2) Otherwise, they challenge for *an organizational structure of information based on specialized ontologies* (designed from the knowledge databases of the different areas) that link with the specific questions in the area dealt with. Serving as an architectural model for the organization of information knowledge and as a way to improve the precision of documentary organization and retrieval. Knowledge can be represented by the use of associative networks and the concepts of the ontology [Baclawski *et al.*, 2000].

Just as Web technologies have a tremendous impact in the dispersion of information, they will necessarily influence the development of specific ontologies for the organization and retrieval of knowledge. Given the diversity of information sources on the Internet, a system of ontologies between Web sites should be very general and multivalent at its first hierarchical level. That is, it should incorporated in *a Dynamic Super-ontology Space* in permanent evolution, stemming from numerous sub-ontologies, each adapted for survival in its usual area of work.

4 Frame of Introduction

The development of an Integrated System of Organization Knowledge implies analyzing and describing user needs in a way that helps specify the tasks assigned to the system.

The foremost of these tasks is *the interpretation of the natural language used in the search equations,* as it may contain terms that are ambiguous or imprecise, and therefore difficult to translate to a system-controlled language.

Secondly, because the documents must be located within the *Documentary Hyperspace,* the system should feature varied modes of manual interaction (e.g. through plausible inference), supported by precise rules for the means of visualization, manipulation and application of data (defined at the "core" of the AI system).

Third, the results must be presented to the user in the same way that interpersonal reporting/feedback takes place in problem-solving —with a *reliance on representational structures of discourse* and various levels of natural language processing procedures.

References

[Baclawski, K. *et al.,* 2000] Knowledge Representation and Indexing using the Unified Medical Language System. *Proceedings Pacific Symposium on Biocomputing,* Singapore: World Scientific Publi. Co., vol. 5, 490-501.

[Chung, Y.M. *et al.,* 1998] Automatic Subject Indexing Using an Associative Neural Network. *Third ACM Conference on Digital Libraries* (June 1998), Pittsburgh, PA: ACM, 59-68.

[Deerwester, S. *et al.,* 1990] Indexing by latent semantic analysis. *Journal of the ASIS,* 41, 221-233.

[Foltz, P.W. *et al.,* 1998] Learning From Text: Matching Readers and Texts by Latent Semantic Analysis. *Discourse Processes,* 25(2-3), 309-336.

[Gaizauskas, R. *et al.,* 2001] Intelligent Access to Text: Integrating Information Extraction Technology into Text Browsers. *Proceedings of Human Language Technology Conference*, San Diego, 189-193.

[Haenggi, D. and Kintsch, W., 1995] Spatial Situation Models and Text Comprehension. *Discourse Processes,* 19, 173-199.

[Kintsch, W., 1988] The role of Knowledge in Discourse Comprehension: A Construction Integration Model. *Psychological Review,* 95(2), 163-182.

A Visual Concept Ontology for Automatic Image Recognition

Nicolas Maillot, Monique Thonnat and **Alain Boucher**
INRIA Sophia Antipolis - Orion Team
2004 Route des lucioles - B.P. 93
06902 Sophia Antipolis, France
{Nicolas.Maillot, Monique.Thonnat, Alain.Boucher}@sophia.inria.fr

keywords : Data Semantics, Ontologies, User Interfaces.

1 Introduction

Multimedia content understanding is expected to play an important role in the future of the Semantic Web. For instance, many image retrieval engines are currently under development. Many of these systems limit their recognition mechanism to low-level image descriptors which are far from semantic notions. On the other hand, other types of system only rely on human annotations [Von-Wun *et al.*, 2002]. We propose an intermediate approach to image understanding. Our approach stems from the fact that experts (e.g. biological and medical experts) of a specific domain often use and share a generic visual vocabulary to describe objects of interest. This paper introduces a domain-independant visual concept ontology which is used as a guide for describing the objects of a domain of expertise. This ontology driven description is planned to support automatic recognition based on image processing techniques. Section 2 of the paper gives an overview of the proposed approach. Section 3 is dedicated to a presentation of a visual concept ontology. A knowledge acquisition tool is introduced in section 5. We finally conclude and present our future work in section 6.

2 Ontology Driven Knowledge Acquisition

In many application domains, concepts of the domain can be structured as a hierarchy of classes with their associated subparts. For instance, this approach is used for organizing knowledge about medical pathologies or biological organisms. This knowledge is shared by the experts of the domain. When describing images, experts also use usual visual notions. To ease knowledge acquisition, we propose a visual concept ontology based on these shared visual notions. The knowledge acquisition process we propose is described in fig. 1. The resulting knowledge base is to be used by a knowledge-based image understanding system [Maillot *et al.*, 2003].

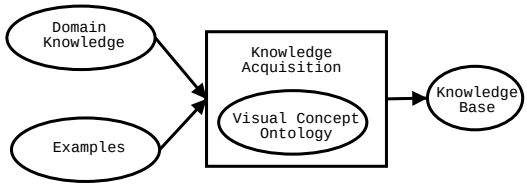


Fig. 1. Knowledge Acquisition Process

3 Visual Concept Ontology

We propose to use a visual concept ontology which is divided in three main parts : (1) spatio-temporal concepts, (2) color concepts, (3) texture concepts.

3.1 Texture Concepts

This branch of the ontology is based on experiments performed by the cognitive science community. The hierarchy presented in fig. 2 is the result of a statistical study on the perception of texture images.

3.2 Color Concepts

The ISCC-NBS[1] color dictionary contains three types of color notions: twenty-eight hue concepts, five lightness concepts (Very Dark, Dark, Medium, Light, Very Ligth) and four saturation concepts (Grayish, Moderate, Strong, Vivid). Note that some color concepts can be combined. For instance, the concept Brillant is defined as the conjunction of the concepts Light and Strong.

3.3 Spatio-temporal Concepts

This part of the ontology provides concepts for describing objects from a spatio-temporal point of view. It is composed of geometric concepts (e.g. Circular Surface, Line) and RCC-8 spatio-temporal relations.

3.4 Context Description

Providing information on the acquisition conditions is necessary to maintain knowledge coherence. For instance, microscopic objects appearance depends on the sensor used for observation. Different context concepts (e.g. sensor, illumination conditions) are used to contextualize visual description.

[1] Inter-Society Council-National Bureau of Standards

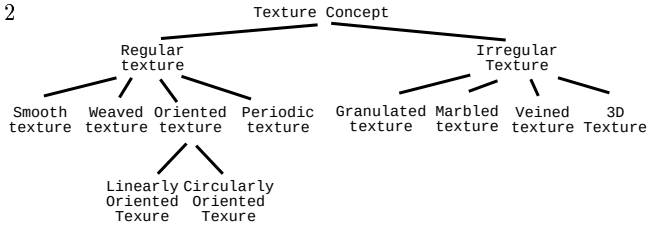


Fig. 2. Texture Concepts

4 Knowledge Representation

Knowledge representation is based on the formalism of a Description Logic. DAML+OIL is used for implementation. A domain object is described through four relations respectively for *hasForSpatioTemporalDescription*, *hasForTextureDecription*, *hasForColorimetricDescription* and *hasForDescriptionContext*: (1)*hasForSpatioTemporalDesc*, (2)*hasForTextureDesc*, (3)*hasForColorimetricDesc*, (4)*hasForDescContext*. A description logic is used to structure the concepts provided by the visual concept ontology:

$$C_i \equiv C_j \sqcap (\exists \text{ "}isASubpartOf\text{"}.C_k)$$
$$\sqcap (\exists \text{ "}hasForSpatioTemporalDesc\text{"}.C_{SpatioTemporal_a})$$
$$\sqcap (\exists \text{ "}hasForTextureDesc\text{"}.C_{Texture_b})$$
$$\sqcap (\exists \text{ "}hasForColorimetricDesc\text{"}.C_{Color_c})$$
$$\sqcap (\exists \text{ "}hasForDescContext\text{"}.C_{Context_d})$$

This means that C_i is a subclass of C_j and a subpart of C_k. The relations *hasForSpatioTemporalDescription*, *hasForTextureDescription*, *hasForColorimetricDescription*, *hasForDescriptionContext* are respectively restricted to concepts $C_{SpatioTemporal_a}$, $C_{Texture_b}$, C_{Color_c}, $C_{Context_d}$. The powerful expressiveness of description logics allows to define $C_{SpatioTemporal_a}$, $C_{Texture_b}$, C_{Color_c}, $C_{Context_d}$ as unions or intersections of different concepts provided by the visual concept ontology.

5 A Knowledge Acquisition Tool

We have implemented a knowledge acquisition tool called OntoVis [2] composed by three main modules. (1) Domain knowledge acquisition, (2) Ontology-driven visual acquisition, (3) Image example management. We have used the JAVA language to build this tool. The Jena toolkit [3] is used for knowledge acquistion and ontology management.

5.1 Domain Knowledge Acquisition

Our tool allows the expert to define domain objects hierarchy (taxonomy). It is also possible to define a subpart hierarchy (partonomy).

[2] http://www.inria.fr/orion/personnel/Nicolas.Maillot/OntoVis
[3] http://www.hpl.hp.com/semweb/jena2.htm

5.2 Visual Description

This module allows the ontology-driven description of domain objects Currently, a list of visual concepts are displayed to the screen in a symbolic manner. The user is then able to select useful concepts for description.

5.3 Example Database Management

Whenever a domain object or a subpart is described with visual concepts, it is useful to give examples of the visual concepts used for description. For instance, the visual concept Circular Surface can used to describe the shape of a specific object. The user can provide images which examplify the visual concepts. For subpart description, it is also possible to select specific regions of interest in the provided images. Once examples have been provided, the symbolic description guided by the ontology is attached to them.

6 Conclusion and Future Work

We propose an approach to knowledge acquisition for the visual description of the objects from a domain of expertise. Our approach is based on a visual concept ontology. This ontology is used as a guide for describing taxonomies and partonomies of objects and their subparts. A graphical tool is also proposed and allows knowledge acquisition based on the visual concept ontology. Visual concepts used during knowledge acquisition can be examplified. The advantage of using a visual concept ontology is to partially fill the semantic gap between the image signal and domain concepts. Indeed, visual concepts are close to image features which can be computed thanks image processing techniques.

As explained in the previous section, visual concepts are displayed in a symbolic form. We are planning to display them in a graphical way: In particular, spatio-temporal visual concepts should be manipulated with a drawing tool. The important point is that every visual primitive drawn should remain semantically anchored. Texture and color concepts should also be represented in a graphical and user-friendly way.

Our goal is to fill the gap between image features and visual concepts used during knowledge acquisition. We are currently experimenting machine learning techniques to achieve this goal.

References

[Von-Wun *et al.*, 2002] S. Von-Wun, L. Chen-Yu, Y. Jaw Jium and C. Ching-Chih. Using sharable ontology to retrieve historical images. In *Proceedings of The Second ACM/IEEE-CS joint conference on Digital libraries (JCDL 2002)*, pages 197–198, Portland, Oregon, USA, June 2002.

[Maillot *et al.*, 2003] Nicolas Maillot, Monique Thonnat and Alain Boucher. Towards Ontology Based Cognitive Vision. In *Proceedings of The Third International Conference On Computer Vision Systems (ICVS 2003)* , LNCS 2626, pages 44–53, Graz, Austria, April 2003. Springer-Verlag Heidelberg, 2003.

Mining and Annotating Social Relationship

Yutaka Matsuo[†], **Hironori Tomobe**[‡], **Kôiti Hasida**[†], and **Mitsuru Ishizuka**[‡]
[†] National Institute of Advanced Industrial Science and Technology
[‡] University of Tokyo

1 Introduction

On top of the famous "layer cake" [5], we have a trust layer. Anyone can say anything on the Web; therefore without trust we can not decide which statement we should believe. Trust is an important factor to utilize Semantic Web fully.

However, we have no prominent proposals for the trust layer yet. This paper describes our view toward realization of trust and one approach to build a trust network using a Web mining approach.

2 Local Trust Network

For realizing trust on the network, some research focuses on authentication, access control, and delegation by digital signature. Using a digital signature to RDF statements, we can verify that a certain person wrote them. However, even if we verify the author, how can one verify the author's reliability? information on the Web, how can you know the information is written by a Therefore, it is important to argue whether the source of information is reliable and credible aside from authentication techniques.

The physical world already offers a "web of trust"; it is a kind of social network. I trust one of my friends, therefore I also trust a person introduced by that friend. I trust a company by the reason that one of my patronized companies deals with that company. In this way, our social network works well to assess trustworthiness. Such a mechanism is likely to work well on the Semantic Web, too. Especially, the trustworthiness of persons is important because web resources are usually created by a group or person. Usually, if a person is reliable, what he writes is also reliable.

However, a person usually has many friends, partners, and acquaintances. According to social scientists, a person can name 200 to 5000 people with no aid [1]. It is overwhelmingly demanding to write down all the relations that one has. To make matters worse, such relations are dynamic. New relations appear every day, and old relations weaken gradually. The degree of relation will change over time.

To tackle this problem, two solutions can address that problem:

- Focus only on important relations: For example, permission to access confidential files would only be given to a couple of close friends. However, this network will be so sparse that it might not work well to judge the reliability of a person and a resource.
- Alleviate the cost to write down relations: If everyday software (e.g., mailers, browsers, schedulers and groupware) are equipped with a detector of relation to others, we can automatically generate a list of persons that one may trust. Alternatively, if we could extract a social network from the Web through a Web mining approach, it could be used as a surrogate for the "Web of Trust."

This paper employs the latter option, especially, the Web mining approach.

3 Social Network Extraction

There are many communities in a physical world and online: students at a university, workers at a corporation, members in an academic society, members in an interest groups, and so on. This paper targets an acadeic society: the Japanese Society of Artificial Intelligence (JSAI).

3.1 Invention of Nodes and Edges

We first pick up contributors to the last four annual conferences (JSAI99, JSAI2000, JSAI2001, and JSAI2002) as active members of the JSAI community. Each active member of JSAI is considered as a node in a social network.

Next, edges between nodes are added utilizing Web information. Assume we are to measure the relevance of two names 'Yutaka Matsuo" (denoted X) and "Hironori Tomobe" (denoted Y). We first put queries "X" and "Y", respectively to a search engine and get $\#X$ and $\#Y$ documents including each word in the text. Also, we put a query "X and Y", and obtain $\#(X \wedge Y)$ matched documents. Relevance of "Yutaka Matsuo" and "Hironori Tomobe" is approximated by some relevance measure such as a Jaccard coefficient. We employ the following one, where k is a constant.

$$rel(x,y) = \begin{cases} \frac{\#(A \wedge B)}{\min(\#(A),\#(B)} & if \#(A) > k \; and \; \#(B) > k, \\ 0 & otherwise. \end{cases}$$

It is more useful if each edge has a "label" for the relationship between two persons. We define labels (i.e., classes) for each edge as follows:

- Coauthor: Coauthors of a technical paper

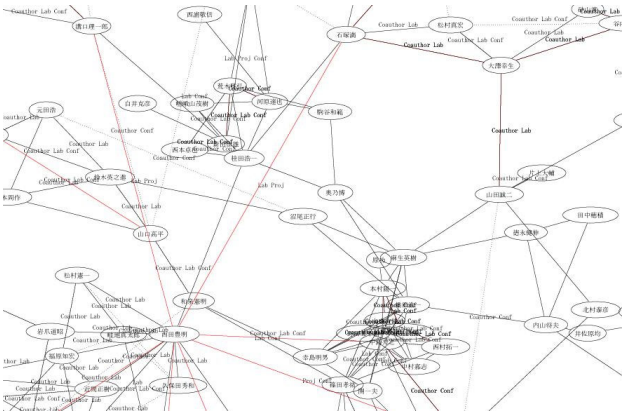


Figure 1: A part of the social network of JSAI.

- Lab: Members of the same laboratory or research institute
- Proj: Members of the same project or committee
- Conf: Participants in the same conference or workshop

We discriminate the relationship by consulting retrieved page contents and applying classification rules. These rules are obtained by a machine learning approach [4].

Figure 1 is a part of the social network of JSAI community. A node is labeled as the corresponding participant name (in Japanese), and an edge is labeled as "Coauthor", "Lab", "Proj", or "Conf".

4 Representing Social Relation by RDF

The relation between two persons extracted from the Web in the previous section is naturally expressed by an RDF statement. A subject and an object are (URIs of) two persons; a relation such as Coauthor and Lab corresponds with a predicate.

Dan Brickley and Libby Miller invented an RDF vocabulary, called FOAF (Friend-of-a-Friend), to create a social network. A user creates one or more FOAF files on her Web server and shares the URLs so software can use the information inside the file[2]. FOAF provides a basic expression for describing people, their basic properties, and the "knows" relation. Jennifer Golbeck et al. extended FOAF so that a user can express a ten-fold degree of trust to others [3]

Here we define new properties "Coauthor", "Lab", "Proj", and "Conf" as subproperties of "foaf:knows" property in our RDF Scheme, shown in Fig. 2. A sample RDF using the new properties is shown in Fig. 3. ("acsn" stands for "academic community social network.") We must prepare an RDF scheme that is appropriate to the community based on a simple expression such as FOAF because the necessary properties depend on a community,

5 Trust Calculation

Using the social network, we can obtain the "authoritativeness" of a node. It can be considered to represent reliability, or in other words, social trust.

```
<rdf:Property
     rdf:about=''http://www.carc.aist.go.jp/~y.matsuo/acsn/0.1/Coauthor''
     rdfs:label=''Coauthor''
     rdfs:comment=''A person coauthors with this person.''>
    <rdfs:domain rdf:resource=''http://xmlns.com/foaf/0.1/Person''/>
    <rdfs:range rdf:resource=''http://xmlns.com/foaf/0.1/Person''/>
    <rdfs:idSefinedBy rdf:resource=''http://xmlns.com/foaf/0.1/''/>
    <rdfs:subPropertyOf rdf:resource=''http://xmlns.com/foaf/0.1/knows''/>
</rdf:Property>
```

Figure 2: RDF scheme to describe the "Coauthor" property.

```
<rdf:RDF
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:foaf="http://xmlns.com/foaf/0.1/"
     xmlns:foaf="http://www.carc.aist.go.jp/~y.matsuo/acsn/0.1/">

<foaf:Person>
    <foaf:name>Yutaka Matsuo</foaf:name>
    <foaf:mbox>y.matsuo@carc.aist.go.jp</foaf:name>
    <foaf:workplacehomepage rdf:resource="http://www.carc.aist.go.jp/">
    <acsn:Coauthor>
         <foaf:Person>
               <foaf:name>Mitsuru Ishizuka</foaf:name>
         </foaf:Person>
    </acsn:Coauthor>
</foaf:Person>%
```

Figure 3: Sample code of FOAF made from mined relation from the Web.

We employ here a PageRank-like model to measure authoritativeness of each member. Each node v has authority value $A_n(v)$ on iteration n. The authority value propagates to neighboring nodes in proportion to the relevance to the node.

The top listed people by this algorithm are authoritative and reliable in the JSAI community. However, authoritative people are not always listed highly by our approach. This results from their relative lack of information that is accessible online. Some people do not post their information online. Especially, elder authorities tend to have produced many publications before the WWW achieved daily use.

6 Conclusion

This paper argues that local trust networks will eventually produce a huge "Web of Trust." We focus on the academic community and show an algorithm to mine a social network using a search engine and a machine learning. The relation can be described in RDF using FOAF vocabulary. Furthermore, the relation is utilized to measure the authoritativeness of members as social trustees.

References

[1] Albert-László Barabási. *LINKED: The New Science of Networks*. Perseus Publishing, Cambride, MA, 2002.

[2] Foaf: the 'friend of a friend' vocabulary. http://xmlns.com/foaf/0.1/.

[3] Jennifer Golbeck, James Hendler, and Bijan Parsia. Trust networks on the semantic web. In *Proc. WWW 2003*, 2003. to appear.

[4] Y. Matsuo, H. Tomobe, K. Hasida, and M. Ishizuka. Mining social network of conference participants from the web. In *Proc. WI2003*, 2003. to appear.

[5] W3C. The semantic web wave (slide). http://www.w3.org/2003/Talks/01-siia-tbl/slide19-0.html, 2003.

Using RDF and Deductive Databases for Knowledge Sharing in Healthcare

Fabiane Bizinella Nardon[1,2], Lincoln de Assis Moura Jr[1,2], Beatriz de Faria Leão[3]

[1] Escola Politécnica, University of São Paulo, Av. Dr. Luciano Gualberto, 380,
05508-900 São Paulo, SP, Brazil
fabiane.nardon@uol.com.br
[2] Atech Tecnologias Críticas, Rua Funchal, 263, 9° Andar,
04554-020 São Paulo, SP, Brazil
lincoln@atech.br
[3] Ministry of Health, Esplanada dos Ministérios, Anexo, Ala B, 1° Andar,
70058-900, Brasília, DF, Brazil
bfleao@attglobal.net

1. Introduction

Throughout a person's lifetime they may receive treatment from many different healthcare providers and each of them will store a piece of the person's medical history. Having that information available in a way that it could be easily retrieved is important not only for the patient, but also for research purposes. Each medical organization usually has its own information system and even inside a single healthcare organization, information tends to be distributed over different departmental systems, with a particular combination of software and hardware platforms. Besides that, there is a large number of different medical vocabularies that can be used to code the same health information in different levels of granularity and for different purposes. This inherently heterogeneous environment makes the task of sharing healthcare information a challenge to be met.

The representation of medical knowledge is not a trivial task, since it is necessary to use a highly expressive, platform-independent formalism that would allow several scattered peers to communicate and exchange knowledge. There is also need for a powerful query language to be able to make complex queries over complex data.

Obviously, the need for data integration and knowledge sharing does not exist only in healthcare and the Semantic Web initiative has been addressing this problem with standards like RDF (Resource Description Framework), DAML+OIL [Horrocks *at al.*, 2001] and OWL (Ontology Web Language). This paper presents our experience in using the RDF and DAML+OIL standards for data integration and knowledge sharing in healthcare. Our approach is to use a representation of the UMLS [NLM, 2003] semantic network in RDF/DAML+OIL as the basic ontology for medical concepts and a deductive database system for inferring and querying the knowledge base. As a test case, an ontology was created for the Brazilian National Health Card data interchange format, a standard for capturing and transmitting health encounter information throughout the country. Since Brazilian health providers have to integrate somehow their data to that standard, this ontology can be the starting point of a huge distributed medical knowledge base.

2. Knowledge Representation and Sharing in Healthcare

Standards for medical knowledge representation and sharing have been subject of research for many years, but none of the proposed standards reached a level of acceptance that would allow sharing health information in large scale. One of the reasons for that is the need of a very flexible and powerful formalism that would allow capturing the complexity of the medical field. The standards proposed by the Semantic Web initiative, like RDF, DAML+OIL and OWL are promising tools for addressing that problem.

There is a large number of medical vocabularies to code information in healthcare. The Unified Medical Language System (UMLS) is an effort of the U.S. National Library of Medicine aiming at facilitating the retrieval and integration of information from multiple biomedical information sources. UMLS is probably the most comprehensive ontology in healthcare, as it defines relationships among a large number of different vocabularies. In order to have a consistent basic ontology for sharing knowledge in healthcare, as part of this work we created a representation of the UMLS semantic network as a DAML+OIL ontology. The complete ontology can be found on <http://www.tridedalo.com.br/2003/07/umls/>.

The Brazilian National Health Card project is an ongoing project sponsored by the Brazilian Ministry of Health aiming at creating an infrastructure for capturing encounter information at the point of care and allowing for the construction of the national repository of clinical data. The system behind the health card stores clinical events and has a multi-level architecture (local, regional, state and federal). The system is in its pilot phase in 44 municipalities, and aims to hold information of fourteen million patients, as part of the pilot-project. In order to

explore the knowledge base created for the Brazilian National Health Card System and provide additional semantics to the data handled by it, we propose a DAML+OIL ontology for that Project. This ontology can be used not only to answer complex queries but also to help healthcare providers with legacy systems to transform their data into the Ministry of Health's format. Using deduction rules and the UMLS knowledge base, this can be accomplished. The National Health Card ontology can be found at http://www.tridedalo.com.br/2003/07/cns/.

The RDF statements created from the UMLS knowledge sources and from the National Health Card System are the basis for inferring new facts from healthcare organizations database. In this work, our approach is to use a deductive database to infer new facts and to answer complex queries, as discussed in the next section.

3. Inference and query over knowledge bases: the TRI-DEDALO system

In order to query and infer new facts from an RDF knowledge base, it is necessary a query language and an inference mechanism. Deductive Databases [Ceri *at al.*, 1990] are databases that provide all the services of a traditional Database Management System and, additionally, allow for deduction of new information using the data explicitly inserted in the database. The deduction of new facts is done by a set of deductive rules that are part of the database schema.

TRI-DEDALO (TRIples, DEduction, DAta and LOgic) is a deductive database that uses a Datalog language extension as query language. The TRI-DEDALO query language supports negation, aggregate functions, arithmetic operations, disjunction, comparison, updating and fuzzy reasoning. The main goal of the TRI-DEDALO system was to build a deductive database to be used in real world applications and the extensions added to the language are essential to achieve that goal.

The TRI-DEDALO system has also features that allow using RDF statements, or triples, as relations in the body of the rule. Also, statements can be used in the head of the rule, allowing, therefore for the inference of new statements. Figure 1 presents a few examples of TRI-DEDALO rules.

Besides the query capabilities, the TRI-DEDALO language allows to update the knowledge base, providing a syntax to express insert, update and delete operations over RDF statements.

The TRI-DEDALO system is implemented as an additional layer to any relational database. The sentences expressed in the TRI-DEDALO language are translated in a set of SQL statements that are submitted to the underlying database by the TRI-DEDALO server.

When an RDF or DAML+OIL document is loaded in a TRI-DEDALO knowledge base, its statements are translated in terms of tuples of the *statement* relation and deduction rules. For example, a DAML+OIL *sameClassAs* property would be translated as:

```
stm(x, rdf:type, y) :- stm(x, rdf:type, z),
                       stm(z, daml:sameClassAs, y).
```

Translating a DAML+OIL ontology in TRI-DEDALO rules, it is possible to use the knowledge expressed on the ontology to answer to queries using not only the explicitly inserted facts in the database, but also the derived information inferred from these facts by the deductive rules.

```
(1) query(patient?x) :- statement(subject?x,
                predicate?<http://sample.com#assignedDoctor>,
                object?<mailto:smith@samplehospital.com.br>).
(2) patientsWithCancer(patient?x) :- statement(subject?x,
                predicate?<http://sample.com#diagnosis>,
                object?<http://sample.com#cancer>).
(3) xmlns:sh=<http://www.samplehospital.com.br#>.
    patientsOlderThan18(patient?x) :-
            patients(name?x, age?y) , y > 18.
    query(name?y):-stm(s?x,p?sh:assignedDoctor,
                    o?<mailto:smith@samplehospital.com.br>) ,
            stm(s?x, p?sh:name, o?y),
            not(patientsOlderThan18(patient?y)).
(4) xmlns:voc=<http://www.avocabulary.org.br#>.
    xmlns:sh=<http://www.samplehospital.com.br#>.
    stm(x, sh:diagnosis, voc:cancer) :-
            stm(x, sh:diagnosis, y) , stm(y, voc:isa, voc:cancer).
```

Figure 1. Examples of rules using statements in TRI-DEDALO. (1) retrieves all patients that were assigned to Dr. Smith. (2) retrieves all patients with a diagnosis of cancer. (3) declares the *sh* namespace and has two rules: the first one retrieves all the patients older than 18 and the second retrieves all the patients assigned to Dr. Smith that are not older than 18. (4) derives new statements stating that a subject *x* has a diagnosis of cancer if there is a statement saying that *x* has a diagnosis *y* and *y* has a "is a" relationship with the concept cancer.

4. Conclusions

Using a combination of ontologies and the TRI-DEDALO deductive database it is possible to share knowledge in healthcare in an efficient and flexible way. The process of information integration starts with a health provider mapping their information to the UMLS and the National Health Card ontologies. Once this mapping is available, the TRI-DEDALO rules can infer information such as semantic equivalent concepts represented in different forms, hierarchy of concepts and so on, achieving this way the much needed semantic interoperability. In healthcare, a highly heterogeneous, distributed and complex domain, the possibility of sharing information can greatly improve the quality of care.

References

[Ceri *at al*., 1990] Stefano Ceri; G. Gottlob; Letizia Tanca. Logic *Programming and Databases*. Berlin: Springer-Verlag, 1990. 284p. (Surveys in Computer Science).

[Horrocks *at al.*, 2001] Ian Horrocks; et al. *DAML+OIL*. See: http://www.daml.org/2001/03/daml+oil-index.html. March 2001.

[NLM, 2003] National Library of Medicine (NLM). *Unified Medical Language System*. 14th. ed. National Library of Medicine, 2003.

Cerebra Server and Construct: Usable Semantics for Domain Experts

Gary Ng and Matthew Quinlan
Network Inference Limited, 25 Chapel Street, London NW1 5DH

1. Introduction

Authoring for the HTML web has become the daily work of many people, supported by standardized, easy-to-use tools and methodologies. Authoring for the Semantic Web [Berners-Lee *et al*., 2001] is a more difficult task, requiring a formalization of domain knowledge in a logically consistent format, to enable machine-processing, while also being intelligible to knowledge engineers and domain experts.

This paper describes Network Inference's combination of two Semantic Web technologies, both utilizing W3C recommendations, to accelerate realization of the full potential of the Semantic Web for business applications and end users.

Construct is an MS-Visio based modeling tool for the graphical editing of ontologies. Cerebra Server is an enterprise platform architected around a commercial inference engine, originally based upon the FaCT reasoner [Horrocks, 2000].

Together, they provide a modeling and inference framework which has the logical reasoning power of a Description Logic Inference Engine, but is as simple to use as MS-Visio.

2. Cerebra Server

Cerebra Server is an enterprise platform, deploying a Description Logic based inference engine with reasoning support for the Semantic Web recommendation OWL [McGuinness et al., 2003], more specifically for OWL-DL. Cerebra Server is deployed as a web service for ease and flexibility of integration. Its XQuery API provides a flexible, expressive and easy-to-use querying syntax.

Cerebra Server is required to support the creation and maintenance of large scale ontologies. The inferencing technology minimizes the complexity and the number of direct relationships needed to represent the business and data models. It also ensures consistency across multiple models, departments and business partners. The engine detects inconsistencies in respect to specified concepts and axioms including disjunction or equivalence.

"The software industry is building an alphabet but hasn't yet invented a common language" Hasso.Plattner SAP AG, 2002 [Gilbert, 2002]. Plattner characterizes the typical use case for solutions in an EAI or SCM scenario where database schemas or business object models of various sources have to be mapped onto a common ontology. Semantic integration using a common vocabulary is one of the greatest challenges for current IT systems. Using Cerebra Server, enterprises are able to process data based on semantics without restricting the vocabulary, allowing the identification of the available resources and services in their field. This will provide a dynamic environment where resources can be exchanged to maintain the integrity of the value-chain as new resources become available or existing resources become redundant.

Reasoning engines are used in non-graphical ontology modeling tools like OilEd [Bechhofer *et al*., 2001] and Protege [Grosso *et al*, 1999], which rely on an Edit-Compile-Reasoning-Edit cycle. They are a great improvement on textual creation of ontologies in languages such as SHIQ, F-Logic and others. Even if some of these modeling tools can generate graphical representations of the ontology, they are not a WYSIWYG, real-time, graphical modeling environment like mind mapping or BPM tools. Our experience shows that the process of creating ontologies is an active process of collaboration - discussion, argument, presentation and politics - involving domain experts with often divergent points of view. They need a real-time, graphical tool to arbitrate their interactions. Tool support for ontology creation should therefore follow the design-pattern of a white-board rather than a database or an Excel sheet.

3. Construct

Construct enables users to create and edit concept taxonomies, and extend these simple structures to support axioms according to the OWL specification using graphical symbols and advanced reasoning.

Complex logical expressions can be made in a graphical notation similar to nested blocks. The expressions are used in two ways: as assertions for the

ontologies and as queries for testing and validating the ontologies. Traditionally the definition of logical queries is a task which can only be fulfilled by a few experts. The queries are expressed in XQuery and processed by Cerebra Server.

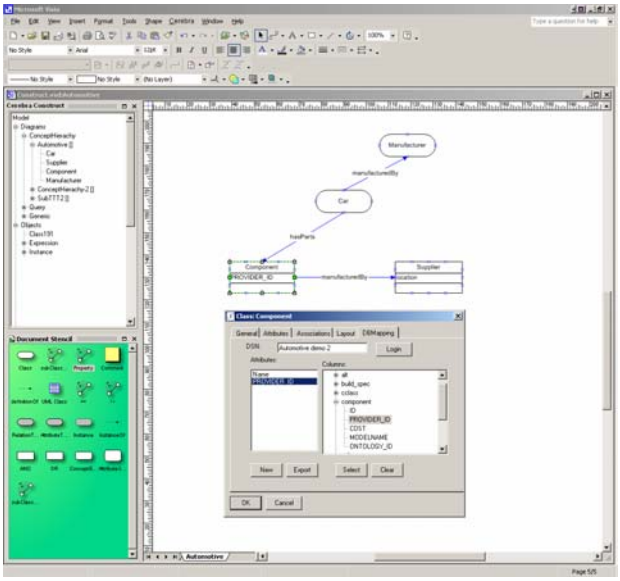


Figure 1 Construct User Interface

Using Construct, ontologies can be mapped to database schemas (see Figure 1). This enables end users to specify queries to Cerebra Server against multiple databases using a common abstract ontology or single database syntax, instead of taking the details of multiple database schemas into account.

Construct's use of OWL-DL, integrated with Cerebra Server's enterprise integration support, can be used to extend 'pure' knowledge representation with actionable business logic and 'policies' to provide adaptive behavior to business systems.

Construct is embedded in the MS-Office tool Visio on MS-Windows platforms. Construct communicates with Cerebra Server via a SOAP interface. This architecture ensures a highly scaleable system configuration, since Cerebra can be used on high-end hardware in order to consolidate large and distributed ontologies from multiple sources. The engine can also reflect instance data from databases or OLAP systems.

Construct and Cerebra Server support distributed ontology development, for example through 'upper' ontologies and individual 'federated' ontologies. They ensure consistency of the local model with linked or associated ontologies. They will detect, for example:

- if another user has defined an equivalent concept even if he is using a different name
- cases in which logical constraints such as conjunctions have been violated.

4. Summary

Cerebra Server is used to integrate hybrid IT-systems, knowledge bases and databases through the use of an ontology layer. It enables users to extend models to capture actionable business rules for automated processing. Ontologies are the critical success factor for these systems, subject to the 'garbage-in, garbage-out' adage. In times of growing demand for ontologies, Construct and Cerebra Server facilitate the move of the responsibility for knowledge specification from highly skilled modeling experts to the end-users who have the domain knowledge.

References

[Berners-Lee *et al*., 2001] Berners-Lee, T., Hendler, J. and Lassila, O. A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities, *Scientific American*, May 2001

[Horrocks, 2000] Horrocks, I. Benchmark Analysis with FaCT. *TABLEAUX 2000*, pages 62-66, 2000

[McGuinness et al., 2003] McGuinness, D. L., van Harmelen, F. OWL Web Ontology Language Overview W3C Working Draft, W3C, 2003

[Gilbert, 2002] Gilbert, A. SAP exec pushes for software harmony, CNET News, 2002

[Bechhofer *et al*., 2001] Sean Bechhofer, Ian Horrocks, Carole A. Goble, Robert Stevens. OilEd: a Reason-able Ontology Editor for the Semantic Web, *Description Logics*, 2001

[Grosso *et al*, 1999] Grosso, W. E., Eriksson, H., Fergerson R. W., Gennari J. H., Tu, S. W., Musen, M. A. Knowledge Modeling at the Millennium (The Design and Evolution of Protege-2000), In *Proc. of KAW99*, 1999

[Fillies et al 2003] Fillies, C., Ng, G., Thunell, A. Cerebra Construct : Inferences for End Users, W3C Conference Poster, Budapest 2003

Tracking Complex Changes During Ontology Evolution

Natalya F. Noy
Stanford Medical Informatics,
Stanford University,
Stanford, CA 94305
noy@smi.stanford.edu

Michel Klein
Vrije University Amsterdam
De Boelelaan 1081a
1081 HV Amsterdam, The Netherlands
michel.klein@cs.vu.nl

1 The Need and Requirements for Version Comparison

For the Semantic Web to succeed, it will require the development and integration of numerous ontologies. As ontology development becomes a more ubiquitous and collaborative process, support for *ontology versioning* [Klein, 2001; Noy and Klein, 2003] becomes necessary and essential. This support must enable users to compare versions of ontologies and analyze differences between them.

There are several reasons to maintain and compare ontology versions. First, ontologies that support the Semantic Web undergo **regular changes**, just as other artifacts do. Second, as ontologies become larger, **collaborative development** of ontologies becomes common. Ontology designers working in parallel on the same ontology need to maintain and compare different versions, examine the changes that others have performed, and so on. Third, the more expressive languages for the Semantic Web, such as DAML+OIL and OWL, are Description Logic (DL) languages. One can view the task of **comparing the asserted and the inferred subsumption hierarchies** in a DL ontology as a versioning problem: The user needs to see how the classification has changed the hierarchy, where were the classes moved, and so on.

We can reuse some of the approaches from the fields of software versioning and collaborative document processing for ontology versioning, but we must keep in mind one crucial difference: In the case of software code and documents, what is compared are *text files*. For ontologies, we need to compare the *structure* and *semantics* of the ontologies and not their textual serialization.

2 Complex ontology changes

The first step in comparing the structure of ontologies rather than their textual serialization is establishing correspondences between concept definitions in two versions, identifying that a concept A in one version became A' in the other.

Identifying correspondences between concepts in different versions leads directly to the second step: identifying simple changes between versions, such as addition or deletion of concepts, change in concept defintions, and so on.. However, in order to assist users in analyzing and understanding the changes that have occurred from one version to another, we must identify *complex* changes as well: For example, it is more useful to know that a concept was *moved* from one place in the hierarchy to another than to know that it was deleted from one and added to the other.

More specifically, the following are some of the complex changes that we have identified.

Add a subtree: Create a new class and create one or more of its subclasses.

Delete a subtree Delete a class and all its subclasses.

Move a subtree to a different location Move a subtree of classes to a different location in the class hierarchy. This operation is essentially equivalent to changing a superclass of the root of this subtree.

Move a set of sibling classes to a different location Move two or more classes that are siblings in the class hierarchy to the same new location in the class hierarchy (i.e., they remain siblings, but under a different parent).

Create a new abstraction Move a set of siblings down in a class hierarchy, creating a new superclass.

Remove an abstraction Delete a class, moving its subclasses to become subclasses of its superclass.

Split a class Split a class into two or more sibling classes.

Merge classes Merge two or more siblings into a single class.

3 User Interface

We have developed PROMPTDIFF, a tool for tracking changes between ontology versions [Noy and Musen, 2002]. It is a plugin to the Protégé ontology environment [Protege, 2002].

Figure 1 shows how PROMPTDIFF presents the result of comparing two versions of the UNSPSC ontology, which is a standardized hierarchy of products and services that enables users to consistently classify the products and services they buy and sell. User input results in regular updates, consisting, for example, of additions of new products, or re-classifications of existing products. In the PROMPTDIFF result, the classes that were deleted are crossed out, the added classes are underlined, and classes that were renamed or changed are in bold. We use color coding to make the changes even more apparent. The warning icon (⚠) overlayed with the class icon indicates that the subtree rooted at the class has undergone some changes.

Figure 2 shows *complex changes* in these two versions of the UNSPSC ontology: The addition of several classes rooted at $Distribution\_and\_Control\_centers\_and\_accessories$ is in fact a tree addition. The icon at the root of the added

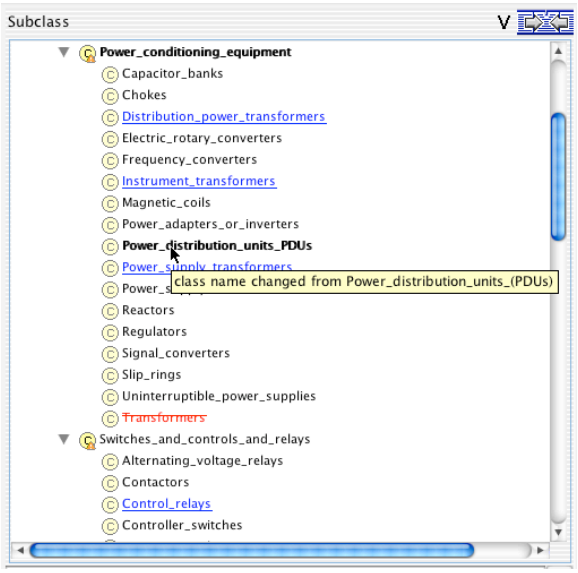


Figure 1: Comparison of two versions of the UNSPSC ontology in PROMPTDIFF. The classes that were deleted are crossed out and the added classes are underlined.

subtree has an overlayed add icon (⊕) indicating that all classes in this subtree have the same status—they were all added in this version. If a whole tree is deleted, an overlayed delete icon (⊗) identified the tree-level operation. The class *Electrical_equipment_and_components_and_supplies* was moved to this location from another position in the tree. The tooltip indicates where it was moved from.

Figure 3 shows the moved class in its old position in the hierarchy: The class appears in grey and the tooltip indicates where the class was moved to.

To summarize, we visualize two types of changes: (1) class-level changes and (2) tree-level changes. For class-level changes, the class-name appearance indicates whether the class was added, deleted, moved to a new location, moved from a different location, or its name or definition has

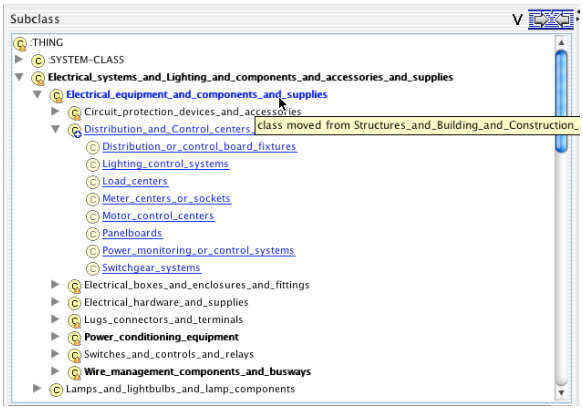


Figure 2: A comparison thats shows a moved class (in bold) and the addition of a subtree.

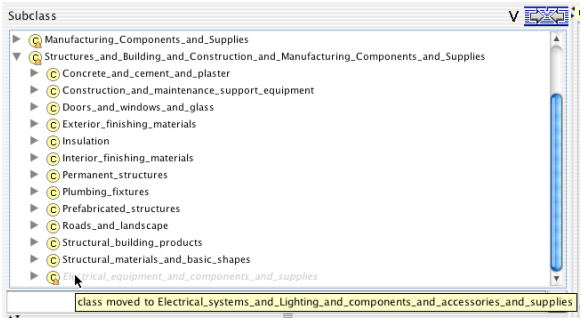


Figure 3: The old position of the moved class (see Figure 2).

changed. If all classes in a subtree have changed in the same way (e.g., were all added or deleted), then the changed icon at the subtree root indicates that the tree-level operation.

4 Outlook

We have presented a tool for examining changes between ontology versions and identified a set of complex changes between ontology versions. Currently, PROMPTDIFF does not display all the changes presented in Section 2, although internally it identifies all of them. We plan to experiment with additional visual metaphors for displaying all complex changes and to evaluate whether using too many different visual clues puts too much of a cognitive load on the user.

Another natural extension of the current tool would be enabling users to accept and reject changes. the default We can also consider using logs of changes if they are available (perhaps grouping together some basic changes in the log into single complex changes) to determine the differences between versions. comparing ontology concepts in likely have

Finally, as we gain more experience with ontology versioning, we will be able to identify more complex changes between versions, and, more important, find automatic ways of determining that such changes have occurred.

Acknowledgments

This research was supported in part by a contract from the U.S. National Cancer Institute.

References

[Klein, 2001] M. Klein. Combining and relating ontologies: an analysis of problems and solutions. In *IJCAI-2001 Workshop on Ontologies and Information Sharing*, pages 53–62, Seattle, WA, 2001.

[Noy and Klein, 2003] Natalya F. Noy and Michel Klein. Ontology evolution: Not the same as schema evolution. *Knowledge and Information Systems*, 5, 2003. in press.

[Noy and Musen, 2002] N. F. Noy and M. A. Musen. PromptDiff: A fixed-point algorithm for comparing ontology versions. In *Eighteenth National Conference on Artificial Intelligence (AAAI-2002)*, Edmonton, Alberta, 2002.

[Protege, 2002] Protege. The Protégé project, http://protege.stanford.edu, 2002.

Capabilities: describing what services do *

Phillipa Oaks, Arthur H.M. ter Hofstede and David Edmond
Centre for Information Technology Innovation - Faculty of Information Technology
Queensland University of Technology
GPO Box 2434, Brisbane, QLD 4001, Australia

1 Introduction

In recent times the Semantic Web, and Web Services have converged into the notion of self-describing semantic web services. These are web services that provide and use semantic descriptions of the concepts in their domain over and above the information provided by WSDL[1].

In this paper we are concerned with advertising web service capabilities in such a way that services can be dynamically discovered based on the functionality they provide. Although the other phases of service interaction, such as evaluation, selection, negotiation, execution and monitoring are important, the discovery phase is the crucial first step.

2 Requirements for capability descriptions

A set of criteria for evaluating capability description languages was described in [Sycara *et al.*, 1999] in reference to agent capabilities. These requirements include expressiveness, abstraction, support for inferences, ease of use, application on the web, and avoiding reliance on keyword extraction and comparison. We believe these high level criteria are relevant in the context of semantic web services but they do not address the specific requirements of dynamic web service discovery. The following requirements are derived from the literature and our observations. A capability description language should provide:

1. The ability to declare what action a service performs.
2. The ability to allow different sets of inputs [Sabou *et al.*, 2003].
3. The ability to declare preconditions and effects in some named rule definition language [Gil and Blythe, 2000].
4. The ability to describe objects that are not input but are used or affected by the capability [Wroe *et al.*, 2003].
5. The ability to refer to ontological descriptions of the terms used in the description and thus place the use of the terms in context [Gil and Blythe, 2000; Sycara *et al.*, 1999].
6. The ability to make explicit the domain or context in which the service operates.
7. The ability to classify capabilities based on aspects of the description enabling exact or partial matches between required and provided capability descriptions [Gil and Blythe, 2000; Sycara *et al.*, 1999].

We refer to these requirements in the next section using the notation (1), with the number representing the requirement.

3 A model of capability

In this section we introduce a model of capability with edited examples from an ontology rendered by the Protege tool[2] in OWL[3].

At the top level we have the class *CapabilityOrParameter*. This superclass allows us to share several informational properties or cases between the classes *Capability* and *Parameter*. These informational properties include the location, source, destination, duration, date or time, manner and topic of a capability, for example:

```
<owl:Class rdf:ID="Capability">
<rdfs:subClassOf rdf:resource="#CapabilityOrParameter"
rdf:type="http://www.w3.org/2002/07/owl#Class"/>
</owl:Class>
<owl:FunctionalProperty rdf:ID="location"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty">
<rdfs:domain rdf:resource="#CapabilityOrParameter"/>
<rdfs:range rdf:resource="#CaseDescription"/>
</owl:FunctionalProperty>
```

A *CaseDescription* is *describedIn* an *OntologicalSource* (e.g. dictionary, thesaurus, ontology, specification or standard). An OntologicalSource *belongsTo* an *Ontology* and is *specifiedBy* a *Fragment* within that ontology.

```
<owl:Class rdf:ID="OntologicalSource"/>
<owl:Class rdf:ID="Ontology"/>
<owl:Class rdf:ID="Fragment"/>

<owl:FunctionalProperty rdf:ID="belongsTo"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty">
<rdfs:domain rdf:resource="#OntologicalSource"/>
<rdfs:range rdf:resource="#Ontology"/>
</owl:FunctionalProperty>

<owl:FunctionalProperty rdf:ID="specifiedBy"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty">
```

*This work is supported by the Australian Research Council SPIRT Grant "Self-describing transactions operating in a large, open, heterogeneous and distributed environment" involving QUT, UNSW and GBST Holdings Pty Ltd.

[1] http://www.w3.org/TR/wsdl12/

[2] http://protege.stanford.edu/

[3] http://www.w3.org/2001/sw/WebOnt/

<rdfs:domain rdf:resource="#OntologicalSource"/>
<rdfs:range rdf:resource="#Fragment"/>
</owl:FunctionalProperty>

The next class in the model is *Capability* which has several properties. The most important is the mandatory *action* represented as a *Verb* that describes the activity the capability performs (1). To allow for the fact that different verbs may be used to express the same action, a reference to a definition in an *OntologicalSource* can be provided (5). Other *semanticRelations* such as synonyms, hypernyms and hyponyms may be used to further elaborate the *Verb*. The ability to provide definitions and alternative meanings to the primary verb assists similarity matching of capabilities (7).

A capability can be performed within a specific domain or context, and an explicit domain or context identifier such as UNSPSC and NAICS (6) is provided by the property *has classification*.

<owl:ObjectProperty rdf:ID="classification">
<rdfs:domain rdf:resource="#Capability"/>
<rdfs:range rdf:resource="#OntologicalSource"/>
</owl:ObjectProperty>

We have grouped rest of the properties of a capability according to the ranges of those properties. We distinguish between properties represented by a *Signature* such as *input*, *affects*, *uses* and *output*, and those represented by *Rules* such as *precondition* and *effect*.

A *Signature* represents a set of *Parameters*. A capability can have zero or more *input*, *uses* and *affects* signature sets, including the empty set (2, 4). For example, a service may take as input a name (string) and an age (integer), or nothing at all. Each signature set for a capability should contain a different combination of parameters.

The *output* property is constrained to have only one signature set, as we take the view that different output set would represent a different capability. A capability must have at least one *output* and/or *effect*.

<owl:ObjectProperty rdf:ID="input">
<rdfs:domain rdf:resource="#Capability"/>
<rdfs:range rdf:resource="#Signature"/>
</owl:ObjectProperty>
<owl:FunctionalProperty rdf:ID="output">
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"
<rdfs:domain rdf:resource="#Capability"/>
<rdfs:range rdf:resource="#Signature"/>
</owl:FunctionalProperty>
<owl:Class rdf:ID="Signature"/>
<owl:ObjectProperty rdf:ID="contains">
<rdfs:domain rdf:resource="#Signature"/>
<rdfs:range rdf:resource="#Parameter"/>
</owl:ObjectProperty>

Preconditions and effects are modelled as *Rules*. Each rule is *expressedIn* a named *Rule Language* and a *ruleExpression* (3) is from an *Ontological source*.

<owl:Class rdf:ID="Rule"/>
<owl:FunctionalProperty rdf:ID="expressedIn"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty">
<rdfs:domain rdf:resource="#Rule"/>
<rdfs:range rdf:resource="#RuleLanguage"/>
</owl:FunctionalProperty>
<owl:Class rdf:ID="RuleLanguage"/>
<owl:FunctionalProperty rdf:ID="ruleExpression"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty">
<rdfs:domain rdf:resource="#Rule"/>
<rdfs:range rdf:resource="#OntologicalSource"/>
</owl:FunctionalProperty>

The class *Parameter* and its associated *DataType* are also described in an *OntologicalSource* (5).

<owl:Class rdf:ID="Parameter">
<rdfs:subClassOf rdf:resource="#CapabilityOrParameter"/>
</owl:Class>
<owl:FunctionalProperty rdf:ID="parameterDescribedIn"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty">
<rdfs:range rdf:resource="#OntologicalSource"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="parameterType"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty">
<rdfs:range rdf:resource="#DataType"/>
</owl:FunctionalProperty>
<owl:Class rdf:ID="DataType"/>
<owl:FunctionalProperty rdf:ID="definedIn"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty">
<rdfs:domain rdf:resource="#DataType"/>
<rdfs:range rdf:resource="#OntologicalSource"/>
</owl:FunctionalProperty>

4 Conclusion

The model for capability descriptions we have introduced in this paper can be used to describe many different types of capabilities and the context they operate in. It can be used to advertise the capabilities of atomic, simple and composite services, and it can be used by service composers and planners to structure a description of what they expect services to provide.

We believe this explicit structured description of service capabilities will allow the dynamic discovery of services based on their functionality, consequently improving the efficiency and effectiveness of the discovery process.

References

[Gil and Blythe, 2000] Yolanda Gil and Jim Blythe. How Can a Structured Representation of Capabilities Help in Planning?, July 2000. In AAAI 2000 workshop on Representational Issues for Real-world Planning Systems.

[Sabou *et al.*, 2003] Marta Sabou, Debbie Richards, and Sander van Splunter. An experience report on using DAML-S. In *Proceedings of the Twelfth International World Wide Web Conference Workshop on E-Services and the Semantic Web (ESSW '03)*, Budapest, 2003.

[Sycara *et al.*, 1999] Katia P. Sycara, Matthias Klusch, Seth Widoff, and Jianguo Lu. Dynamic service matchmaking among agents in open information environments. *SIGMOD Record*, 28(1):47–53, 1999.

[Wroe *et al.*, 2003] Chris Wroe, Robert Stevens, Carole Goble, Angus Roberts, and Mark Greenwood. A Suite of DAML+OIL Ontologies to Describe Bioinformatics Web Services and Data. *International Journal of Cooperative Information Systems*, 12(2):197–224, 2003.

An Application Server for the Semantic Web

Daniel Oberle, Raphael Volz, Steffen Staab

University of Karlsruhe
Institute for Applied Informatics and Formal Description Methods
76128 Karlsruhe
Germany
lastname@aifb.uni-karlsruhe.de

Abstract

The Semantic Web relies on the complex interaction of several technologies involving ontologies. Therefore, sophisticated Semantic Web applications typically comprise more than one software module. Instead of coming up with proprietary solutions, developers should be able to rely on a generic infrastructure for application development in this context. We call such an infrastructure Application Server for the Semantic Web. We present design and architecture as well as our implementation KAON SERVER.

1 Introduction

Ontologies serve various needs in the Semantic Web, like storage or exchange of data corresponding to an ontology, ontology-based reasoning or ontology-based navigation. Building a complex Semantic Web application, one may not rely on a single software module to deliver all these different services. The developer of such a system would rather want to easily combine different –preferably existing –software modules.

So far, however, such integration of ontology-based modules had to be done ad-hoc, generating a one-off endeavour, with little possibilities for re-use and future extensibility of individual modules or the overall system.

We present an infrastructure that facilitates plug'n'play engineering of ontology-based modules and, thus, the development and maintenance of comprehensive Semantic Web applications, an infrastructure which we call *Application Server for the Semantic Web (ASSW)*. It facilitates re-use of existing modules, e.g. ontology stores, editors, and inference engines. It combines means to coordinate the information flow between such modules, to define dependencies, to broadcast events between different modules and to translate between ontology-based data formats.

The following sections talk about design decisions leading to the conceptual architecture of an Application Server for the Semantic Web. Finally, we briefly describe our implementation KAON SERVER.

2 Component Management

Extensibility is a major requirement for an Application Server for the Semantic Web. Hence, the Microkernel design pattern is the first choice. The pattern applies to software systems that must be able to adapt to changing system requirements. It separates a minimal functional core from extended functionality and application-specific parts. In our setting, the Microkernel's minimal functionality must take the form of simple management operations, i.e. starting, initializing, monitoring, combining and stopping of software modules as well as dispatching of messages between them.

This approach requires software modules to be uniform so that they can be treated equally by the Microkernel. Hence, in order to use the Microkernel, software modules that shall be managed have to be brought into a certain form. We call this process *making existing software deployable*, i.e. bringing existing software into the particular infrastructure of the Application Server for the Semantic Web, that means wrapping it so that it can be plugged into the Microkernel. Thus, a software module becomes a *deployed component*. We use the word *deployment* as the process of registering, possibly initializing and starting a component to the Microkernel.

3 Component Description

All components are equal as seen from the Microkernel's perspective. Hence, in order to allow a client discovering the components it is in need of, we have to distinguish between them. Thus, there is a need of a registry that stores descriptions of all deployed components. We came up with a management ontology that is primarily used to facilitate component discovery for the application developer. Its taxonomic core is presented in the definitions below.

Component Software entity which is deployed to the Microkernel.

System Component Component providing functionality for the Application Server for the Semantic Web itself, e.g. a connector.

Functional Component Component that is of interest to the client and can be looked up. Ontology-related software modules become functional components by making them deployable, e.g. RDF stores.

External Service An external service cannot be deployed directly as it may be programmed in a different language, live on a different computing platform, uses interfaces unknown, etc. It equals a functional component from a client perspective. This is achieved by having a proxy component deployed that relays communication to the external service.

Proxy Component Special type of component that manages the communication to an external service. Examples are proxy components for inference engines.

4 Conceptual Architecture

The design elements of the architecture are conceptually divided into Connectors, Management Core, Interceptors and Functional Components, like depicted in Figure 1.

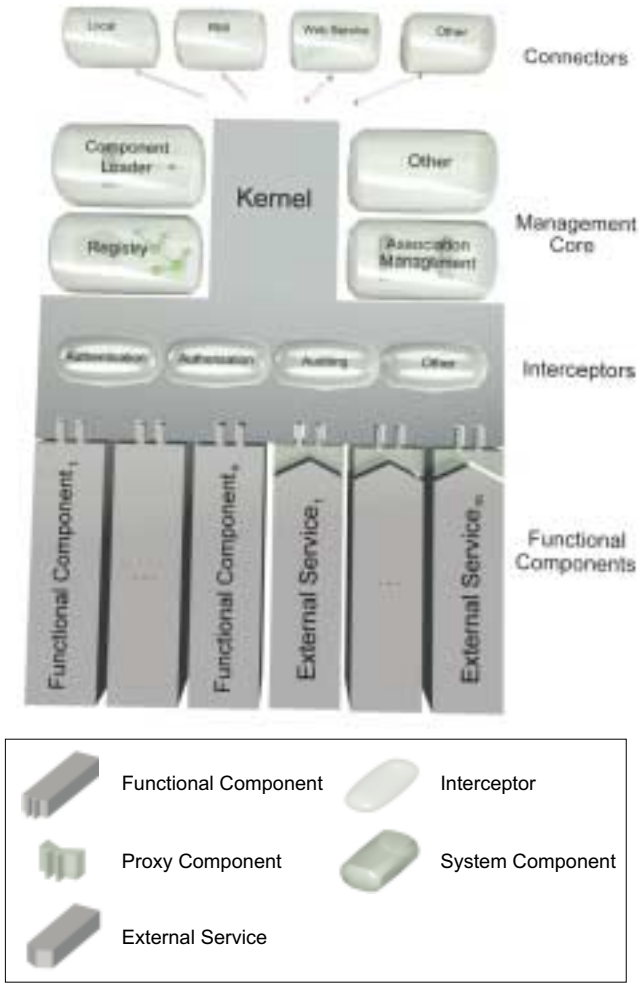


Figure 1: Conceptual Architecture

Connectors
Connectors are system components. They send and receive requests and responses over the network by using some protocol. Apart from the option to connect locally, further connectors are possible for remote connection. Counterparts to a connector on the client side are surrogates for functional components that relieve the application developer of the communication details similar to stubs in CORBA.

Management Core
The Management Core comprises the Microkernel. The Management Core is required to deal with the discovery, allocation and loading of components. The registry, a system component, manages descriptions of the components and facilitates the discovery of a functional component. Another system component called association management allows to express and manage relations between components. Event listeners can be put in charge so that a component A is notified when B issues an event or a component may only be undeployed if others don't rely on it. When provided a deployment description, the component loader facilitates the deployment process for a client. System components can be deployed and undeployed ad hoc, so extensibility is also given for the Management Core.

Interceptors
Interceptors are software entities that monitor and modify it before the request is sent to the component. Security is realized by interceptors which guarantee that operations offered by functional components (including data update and query operations) in the server are only available to appropriately authenticated and authorized clients. Transactions, modularization and evolution spanning several ontology stores may also be realized by interceptors.

Functional Components
RDF stores, ontology stores etc., are finally deployed to the management kernel as functional components. In combination with the component loader, the registry can start functional components dynamically on client requests.

5 KAON SERVER - An implementation

Our implementation of an Application Server for the Semantic Web, called KAON SERVER, offers a uniform infrastructure to host functional components, in particular those provided by the KAON Tool suite[1]. The latter includes tools allowing easy ontology creation and management, as well as building ontology-based applications in Java. The KAON SERVER architecture reflects the conceptual architecture presented in the previous section.

In the case of the KAON SERVER, we use the Java Management Extensions (JMX[2]) as it is an open technology and currently the state-of-the-art for component management. Basically, JMX defines interfaces of managed beans, or *MBeans* for short, which are JavaBeansthat represent JMX manageable resources. MBeans are hosted by an *MBeanServer* which allows their manipulation. All management operations performed on the MBeans are done through interfaces on the MBeanServer. In our setting, the MBeanServer realizes the kernel and MBeans realize components.

[1] Karlsruhe Ontology and Semantic Web Tool suite, http://kaon.semanticweb.org

[2] http://java.sun.com/products/JavaManagement/

Semantic Annotation and Matchmaking of Web Services

Joachim Peer
Institute for Media and Communications Management, University of St. Gallen
Blumenbergplatz 9, 9000 St. Gallen, Switzerland
joachim.peer@unisg.ch

1 Introduction

Automatic retrieval, evaluation and execution of Web Services is a potential "enabler technology" for innovative applications like dynamic personal information management systems (PIMs). To enable dynamic and intelligent service usage, semantically rich description of services and their operations is required. However, the current standard for the description of Web Services, WSDL, is following the tradition of interface description languages (IDLs), focusing on syntactic descriptions of operation names and input/output types rather than on the semantic meaning of these data structures. This paper presents a proposal for a more satisfying way of Web Service markup and matchmaking.

2 Related Work

The Software Engineering community has invested great efforts into the proper description of software components to enable automatic or semi-automatic software component retrieval and automatic programming. Examples are Larch [Guttag and Horning, 1993], Meyer's work on "design by contract" and the work by [Fischer *et al.*, 1995] on NORA/HAMMR. The basic idea was to express the semantics of components and operations by means of logical expressions and to use theorem provers to test for "matching conditions" [Zaremski and Wing, 1995]. The serious disadvantage of this approach – exponential response times – has been addressed by several papers, and several heuristics to minimize the problem have been proposed. One of the most promising ways to minimize that problem is to restrict the expressiveness of the underlying logical markup language in order to gain algorithmic efficiency. [Li and Horrocks, 2003] describe service matchmaking as variants of subsumption checks for description logic concepts, with concepts representing services. DAML-S is an effort to describe services by means of description logics, namely DAML+OIL. The DAML-S ontology is a set of standard terms to be used for service descriptions by means of description logic constructs. Among the disadvantages of DAML-S are some usability issues arising from RDF encoded service descriptions [Ankolekar *et al.*, 2002], its incompatibility with WSDL and its restrictions on the specification of pre- and post-conditions. The concept sketched here aims to minimize these problems by building directly on top of WSDL and by incorporating a Horn-style rule language to express pre- and post-conditions.

3 Semantically Rich Web Service Descriptions

The approach to Semantic Web Service annotation presented in this paper has to provide a certain standard of usability, reasonable computational complexity, and compatibility with WSDL. The following sections briefly describes a possible way to achieve these goals.

3.1 Enriching WSDL with Semantics

We can embed semantic information into the data model of WSDL by several means, e.g. by introducing a new namespace to be used by qualified attributes or by facilitating the recently introduced *substitution framework* of WSDL 1.2.

3.2 Modeling Input and Output Concept Descriptions

To enhance interoperability between different vocabularies of description, we can map WSDL message parts (XML types) to ontological concepts (cf. [Peer, 2002]). This will increase the usefulness of signature matching. For complex XML and description logic structures, additional mapping information for the relations between DAML-S ontologies and XML grammars must be provided [1].

3.3 Modelling Pre- and Post-Conditions

As described in Sect. 2, modelling of pre- and post-conditions is a central concern of the description of the semantics of software components. Each operation may have a different set of pre- and post-conditions.

We propose to use a subset of First Order Predicate Logic (with equivalence and sorts) to model pre- and post-conditions for Web Service operations. As a consequence of predicate logic's known problems of undecidability and incompleteness, we need to abstain from certain features of FOL to ensure the requirement of computational tractability. To this end, we impose several restrictions upon the language supported by our concept. Firstly, we do not permit the use of

[1] Some preliminary tool support can be found online at $http://sws.mcm.unisg.ch/work.html\sharp mapper$

functions of arity > 0. Therefore, the set of terms $\mathcal{T}(\mathcal{X}, \mathcal{F})$ is restricted to functions $\mathcal{F}$ with arity of zero and a set of variables $\mathcal{X}$. Further, we require pre- and post-conditions to adhere to the Horn subset of FOL.

To incorporate pre- and post-conditions into WSDL documents, we propose to use an XML grammar derived from RuleML. We require that all predicates and sorts used in conditions are identified by an URI. This enables the creation of ontologies of predicates and the application of Semantic Web operations like concept subsumption checking.

4 Prototypical Implementation

We have implemented the concept proposed in this paper in Java. The implementation consists of two main components:

- A **registry component** which manages service advertisements in the form of WSDL documents, annotated using the techniques described in this paper. Providers of services can upload, edit and remove annotated WSDL documents and related ontologies. During the upload of annotated WSDL documents, the container parses the file and tests if the description logics concepts used in the document are already registered. In order to ensure the quality of the service, the registry component refuses to store WSDL documents that contain yet unregistered description logic constructs.
- A **matchmaking component** which accepts requests for service operations and returns a list of fitting candidates. The matchmaking component follows a *wrapper* approach: it is designed around two Java interfaces which define methods to be implemented by components for (i) description logics concept subsumption and (ii) clause subsumption. Description logic subsumption operations are required for all filtering phases, while clause subsumption is used exclusively for pre- and post-condition matching. The wrapper architecture enables us to easily plug in external components, without changing the essential algorithms of the tool. Currently we provide an interface to the tableaux based description logic engine RACER and to the saturation based theorem prover SPASS.

We have conducted a series of scalability tests. Our measurements suggest that the runtime behavior is linear. We came to the preliminary conclusion that the the conceptual restrictions (e.g. in the rule language) and our technical design decisions pay off.

The prototype and its source code are freely available for download at *http://sws.mcm.unisg.ch*. The results of the scalability tests can be also be found there.

5 Current Limitations and Future Work

A central limitation of the work as presented in this paper is that we have not undertaken any formal investigation of the consequences of the logical and architectural design we propose. Although some early scalability tests have been performed, additional tests, involving more complex services, need to be carried out. Another limitation of our current batch of tests is that we focused primarily on performance, and we did not explicitly look for scenarios which might be negatively affected by the restrictions we imposed on our rule language.

Another essential problem left for future work is to extend the matchmaking process from *atomic* operations to whole *processes*, which combine several Web Service operations to achieve a specific goal. Work on automatic planning, conducted by the AI community, may be leveraged to the area of Web Services to achieve this task. Among potentially useful approaches are Situation Calculus (the application to Web Services was demonstrated by [McIlraith and Son, 2002]), Hierarchical Task Networks (HTN's) (as demonstrated by [Hendler *et al.*, 2003]), Graphplan [Blum and Furst, 1995], and Constraint Logic Programming.

References

[Ankolekar *et al.*, 2002] A. Ankolekar, F. Huch, and K. Sycara. Concurrent execution semantics of DAML-S with subtypes. In *Proceedings of The First International Semantic Web Conference (ISWC)*, 2002.

[Blum and Furst, 1995] A. Blum and M. Furst. Fast planning through planning graph analysis. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI 95)*, 1995.

[Fischer *et al.*, 1995] B. Fischer, M. Kievernagel, and G. Snelting. Deduction-based software component retrieval. In *Proc. IJCAI-95 Workshop on Formal Approaches to the Reuse of Plans, Proofs, and Programs, Montreal, August 1995*, 1995.

[Guttag and Horning, 1993] J. Guttag and J. Horning. *Larch: Languages and Tools for Formal Specification*. Springer Verlag, 1993.

[Hendler *et al.*, 2003] J. Hendler, D. Wu, E. Sirin, D. Nau, and B. Parsia. Automatic web services composition using SHOP2. In *Proceedings of The Second International Semantic Web Conference(ISWC)*, 2003.

[Li and Horrocks, 2003] L. Li and I. Horrocks. A software framework for matchmaking based on semantic web technology. In *Proc. of the Twelfth International World Wide Web Conference (WWW 2003)*, 2003.

[McIlraith and Son, 2002] S. McIlraith and T. Son. Adapting Golog for composition of semantic web services. In *Proceedings of the Eighth International Conference on Knowledge Representation and Reasoning (KR2002)Toulouse, France, April 2002*, 2002.

[Peer, 2002] J. Peer. Bringing together semantic web and web services. In Ian Horrocks and James Hendler, editors, *Proceedings of The First International Semantic Web Conference (ISWC)*, 2002.

[Zaremski and Wing, 1995] A. M. Zaremski and J. M. Wing. Specification matching of software components. In *Proceedings of 3rd ACM SIGSOFT Symposium on the Foundations of Software Engineering*, 1995.

I-X: Task Support on the Semantic Web

Stephen Potter, Austin Tate and Jeff Dalton
Artificial Intelligence Applications Institute,
School of Informatics, The University of Edinburgh,
Crichton Street, Edinburgh, EH8 9LE, UK
{s.potter,a.tate,j.dalton}@ed.ac.uk

1 Introduction

The coordination of resource and activity to achieve some common objective is a key task within modern virtual organisations. The Semantic Web initiative promises to increase the number of knowledge and information resources available, presenting more (and more varied) opportunities for interaction. However, as the number and complexity of these interactions increases, so too does the need for task support tools. This extended abstract describes our research into support for mixed-initiative (that is, involving both human and computer agents) collaborative tasks in distributed environments. At the heart of this approach is the *I-X* technology. This is introduced in section 2, while section 3 illustrates the nature of the task support it offers through the description of two applications. Section 4 outlines some future directions that this work will pursue and the final section provides a summary and some conclusions.

2 I-X: A Task Support Architecture

The I-X[1] technology is intended to provide a well-founded approach to allow humans and computer systems to cooperate in the creation or modification of some product, be it a document, plan, design or physical entity [Tate *et al.*, 2003]. The I-X tools support users in selecting and performing processes and creating or modifying process products. A set of *issues* is associated with the process or product, representing unsatisfied requirements, problems arising from critique and so on. Both processes and process products are considered, in the abstract, to consist of (perhaps hierarchically composed) *nodes*: these correspond to activities in the process or parts of the product. The relationships between nodes are defined by a set of *constraints*. Finally, *annotations* can be associated with these elements to capture other, perhaps less formal, information surrounding the collaboration. Together, these elements constitute the <I-N-C-A> (<Issues-Nodes-Constraints-Annotations>) model and provide a unifying framework that allows the communication — using an XML encoding — of elements from one agent to another.

[1]The 'I' of I-X is meant to convey all of 'intelligent', 'intelligible', 'integrated' and 'issue-based', with the 'X' being the uninstantiated variable. See i-x.info for more about I-X.

2.1 The I-X Tool Suite

The principal interface to these tools, the *I-P*[2] (I-X Process Panel) can be seen, at its simplest, as a 'to-do' list for its user; however, when used in conjunction with other I-X agents, it can become a sophisticated workflow and messaging tool. A panel corresponds to its user's 'view', in <I-N-C-A> terms, of the current activity, and the current state of the collaboration is used to generate dynamically the support options the tool provides. For example, associated with a particular activity node might be suggestions for performing it using known procedural decompositions, for invoking an agent offering a corresponding capability, or for delegating the activity to some other agent.

The other tools in the suite include messaging tools and information viewers and editors, used, for example, to allow the user to specify relationships with other agents in the environment, and to create and publish Standard Operating Procedures (SOPs), generic approaches to archetypal activities. Particularly relevant to this discussion is the *I-Q* (I-Query) tool. I-Q is a generic I-X agent shell which, when embodied with the appropriate mechanisms, presents an interface to a particular Semantic Web information resource, providing seamless integration with other I-X agents.

3 Demonstration Applications

In this section we illustrate the use of I-X to support activity involving Semantic Web resources through the brief description of two demonstrations that have been developed.

3.1 Workshop Organisation

This application involves the following scenario: an official of a UK technology research funding body is charged with organising a workshop concerning some particular area of computer science so as to get an overview of its current state.[2] Accordingly, from a set of published SOPs, she selects *Organise workshop*. Now shown on her I-P[2] are the sub-tasks needed to achieve this goal, involving selecting attendees, choosing a location and date, fixing the agenda, and so on.

Further decomposing the *select attendees* task, the initial sub-task is *identify steering committee* for the workshop. An available I-Q agent is known to be capable of performing this

[2]Developed as part of the AKT Project: see www.aktors.org.

task for topics drawn from the ACM classification of computer science.[3] This agent constructs appropriate RDQL[4] queries and sends them via http to an RDQL interface onto an RDF triple store. This database describes the current state of UK research in (predominantly) computer science through some millions of triples extracted from various sources by various techniques, the triples being described according to a number of published ontologies.[5] The RDQL formed by the I-Q agent refers to these ontologies and implicitly contains knowledge of the contents of the triple store, and the agent 'knows' how to communicate with the store and process its responses. However, this is opaque to the I-P^2 user, who need know nothing about this transaction, and, having selected the appropriate topic from the ACM classification and parameterised her message to the I-Q agent, receives a message naming the suggested steering committee along with their contact details a few seconds later.

This sub-task completed, the other steps in the SOP are performed by the user (assisted by links to relevant tools and information) or delegated accordingly. Finally, to discuss this workshop and confirm its dates, location and content with the steering committee, she initiates a videoconference; an additional SOP, downloaded from a meeting-support website,[6] provides experience-based assistance with conferencing technology set-up.

3.2 Search and Rescue

This application involves more complicated interactions with Semantic Web resources. The scenario surrounds the coordination of resources to rescue and care for a downed aviator.[7]

On being alerted about the emergency, the SAR (Search And Rescue) coordinator, through his I-P^2, selects an appropriate SOP containing a number of sequential steps such as *select hospital* and *select SAR resource*. In this environment, the SAR domain and the infrastructures — including medical facilities — of the countries in the locale are encoded according to DAML-O ontologies, with both ontologies and knowledge bases available as web resources.[8]

A particular I-Q agent in this domain has the ability to access and reason with the appropriate ontologies, and so can extract from the knowledge bases information about hospitals offering specialist care facilities (for example, burns units). So, once the nature of the injuries to the airman has been established, this agent can be invoked to suggest the closest appropriate hospitals.

SAR resources — helicopters, patrol boats, etc — are described as DAML-S services, and advertised to a matchmaking service.[9] For the purposes of selecting amongst these resources, a second I-Q agent is able to construct and send to the matchmaker an appropriate DAML-S request, instantiated with the location of the airman and the location of the selected hospital. When selecting an appropriate resource, then, this agent can be invoked to act as an intermediary to the matchmaker, constructing appropriate requests and parsing the returned results.

4 Future Directions

With particular reference to operating on the Semantic Web, there are a number of areas of work that would enhance the I-X support environment and encourage interoperability, and which we hope to address in the near future. For instance, publishing <I-N-C-A> information according to OWL ontologies would make resources such as SOPs more readily available to a wider community, while describing the capabilities of I-X agents using OWL-S would make these more visible externally, and position I-X more centrally within the developing ideas of web service description and invocation. More generally, some consideration of the whole notion of task support within the Semantic Web is needed: What sort of tasks will be performed? What sort of support is necessary/possible? How might this support best be delivered?

5 Summary and Conclusions

The intention of this extended abstract has been to describe the I-X environment for collaborative task support, with particular reference to placing this in the context of the Semantic Web and its emerging standards, concepts and resources. The potential benefits are mutual: on the one hand, I-X task support is greatly enhanced by exploiting Semantic Web information resources, as illustrated by the applications described above; on the other hand, as the Semantic Web moves towards its goal of empowering users to achieve more than information browsing, the need for integrated intelligent task support of the sort provided by I-X becomes more evident.

Acknowledgments

The work described in this extended abstract is supported, in part, by the AKT IRC and by the DARPA DAML program.

The University of Edinburgh and research sponsors are authorised to reproduce and distribute reprints and on-line copies for their purposes notwithstanding any copyright annotation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of other parties.

References

[Tate *et al.*, 2003] A. Tate, J. Levine, J. Dalton, and A. Nixon. Task Achieving Agents on the World Wide Web In *Spinning the Semantic Web*, Fensel, D., Hendler, J., Liebermann, H. and Wahlster, W. (eds.), MIT Press, 2003.

[3] See `www.acm.org/class/1998/overview.html`.

[4] RDQL is an SQL-like query language for RDF; see: `www.hpl.hp.com/semweb/rdql.htm`.

[5] For more about the triple store see, see `triple-store.aktors.org`.

[6] `i-me.info/resources/coakting`.

[7] Developed in the course of the CoSAR-TS project: see `www.aiai.ed.ac.uk/project/cosar-ts`.

[8] See, for example, the infrastructure ontology at: `www.daml.org/experiment/ontology/infrastructure-elements-ont`, and the knowledge base about a (fictitious) country at: `sonat.daml.org/DAMLdemo/instances/enp/nc-BINNI.daml`.

[9] `www-2.cs.cmu.edu/~softagents/daml_Mmaker/daml-s_matchmaker.htm`.

SEMAGEN: A Semantic Markup Generation Framework

James Starz

ISX Corporation
4301 North Fairfax Drive, Suite 370
Arlington, VA, 22203
jstarz@isx.com

I. Introduction

It is necessary to have ontologies and semantically-grounded markup to enable the power of the Semantic Web. Although standards exist for extensible markup language (XML) schemas and database interfaces, these data sources contain only syntactic data, not explicit semantic information. There is a need to bridge the gap between structured data sources and semantic data. Ontology creation is difficult because it is best done by those familiar with the ontology's domain and the field of knowledge representation. The problem of using publicly available ontologies is they may lead to terminology that is inconsistent with that of the organization using the ontology. Generating markup has similar difficulties. It can be created manually using text editors or graphical interfaces. The second option is to automatically generate markup through translation. Unfortunately, the different options for creating markup are either labor intensive or produce data of marginal quality. There is often a prohibitive tradeoff between cost of creating markup and the perceived value [Bosak, 2001]. This poster presents a technique to easily translate structured data into semantically rich ontology-based markup.

II. Markup Generation Architecture

The Semantic Web consists of an emerging landscape of technologies. In designing a toolkit for the Semantic Web, an architecture is needed that is not tied to specific data representations.

To achieve this goal, the problem of converting syntactic markup to semantic markup is broken into subtasks.

1. Convert schema representation into an ontology
2. Map ontology representation to customized ontology representation
3. Markup conversion using mapping representation

For the implementation of this process, all incoming markup was XML adhering to an XML Schema. The Ontology Web Language (OWL) was used to represent mappings, as well as the ontologies.

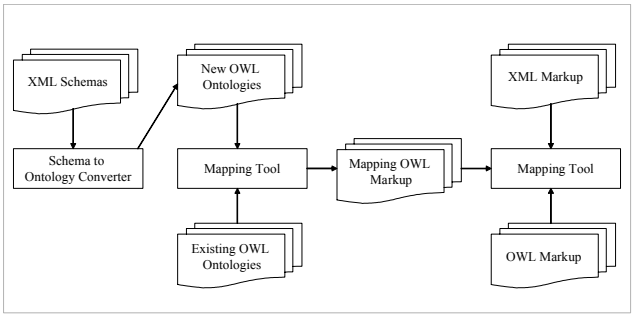


Figure 1. Implementation Architecture

III. Ontology Generation

This process will take a syntactically defined language and create a candidate ontology based on it. For our purposes, I consider a simplified view of ontologies that uses classes and their properties. I consider the following language constructs for the process.

Expression Operator	Examples	Rules
"Nesting" or non-terminal	$A \rightarrow B$ $B \rightarrow C$	A is a class. B is a property relating A to C
Literal	$A \rightarrow a$	A is a property with a literal value of a
Concatenation	$A \rightarrow a\ b$	A has two property values. Though a and b are ordered, the ordering is lost using this approach
Or	$A \rightarrow a \mid b$	A is a property for a, and A' is a property for b. If a and b are the same "type" only 1 properties may be necessary
Kleene Star	$A \rightarrow a^*$	A is a property. The cardinality for the property is between 0 and infinity
One or more	$A \rightarrow a+$	A is a property. The cardinality for the property is between 1 and infinity

Table 1. Expression Rules

In my implementation, an XML schema is fed into a parser detecting various expressions. In the table above, "nesting" refers to OWL object properties. The other examples refer to datatype properties with the literal corresponding to an XML Schema built-in datatype. In my implementation, I choose to ignore cardinality restrictions for simplicity. The approach assumes the language syntax provides implicit information about relationships of objects. I believe this assumption will work in most cases and provide a straightforward ontology that most users could understand and extend. The following example shows a simple schema for representing people along with their address.

```
<xs:schema … >
 <xs:element name="Address">
   <xs:complexType>
   <xs:sequence>
     <xs:element name="City" type="xs:string"/>
     <xs:element name="State" type="xs:string"/>
   </xs:sequence>
   </xs:complexType>
 </xs:element>
 <xs:element name="Person">
   <xs:complexType>
   <xs:sequence>
     <xs:element name="Name" type="xs:string"/>
     <xs:element name="BirDay" type="xs:date"/>
     <xs:element ref="Address"/>
   </xs:sequence>
   </xs:complexType>
 </xs:element>
</xs:schema>
```

Figure 2. An Example XML Schema

```
<rdf:RDF … >
<owl:Ontology rdf:about="" />
 <owl:Class rdf:ID="Person"/>
 <owl:Class rdf:ID="Address"/>
 <owl:ObjectProperty rdf:ID="PersonAddress">
  <owl:domain rdf:resource="#Person"/>
  <owl:range rdf:resource="#Address"/>
 </owl:ObjectProperty>
 <owl:DatatypeProperty rdf:ID="BirDate">
  <owl:domain rdf:resource="#Person"/>
  <owl:range rdf:resource="&xsd;date"/>
 </owl:DatatypeProperty>
 <owl:DatatypeProperty rdf:ID="City">
  <owl:domain rdf:resource="#Address"/>
  <owl:range rdf:resource="&xsd;string"/>
 </owl:DatatypeProperty>
 <owl:DatatypeProperty rdf:ID="State">
  <owl:domain rdf:resource="#Address"/>
  <owl:range rdf:resource="&xsd;string"/>
 </owl:DatatypeProperty>
 <owl:DatatypeProperty rdf:ID="LastName">
  <owl:domain rdf:resource="#Person"/>
  <owl:range rdf:resource="&xsd;string"/>
 </owl:DatatypeProperty>
</rdf:RDF>
```

Figure 3. Example schema translated into OWL

IV. Mapping Generation

There is a desire to allow end users to customize the output of the conversion process. Instead of basing markup conversion solely on the default ontology, an intermediate step is introduced using OWL markup. The markup translation will only use this mapping representation when doing conversion. Therefore, the markup translation can occur without knowledge of the ontology representation.

A few mapping problems can occur with this approach. A property can refer to a class that no longer exists. If this occurs, the class is replaced with the class higher in the nesting. If no such class exists, the relation can be with a generic object, such as the OWL "Thing". This provides some limited ability to change the ontology semantics.

V. Mapping Translation

The final stage of the technique involves processing some representation using the mapping. I assume that incoming markup could have a directed graph structure. In my implementation, I assume markup will be in XML. The process is similar to the ontology creation step. However, now the mapping file will be the basis of what the output should be. As I find nodes of the graph, I create new objects for each class we find in the mapping file. Each new object is assigned an unambiguous identifier. All properties that are found will have the last instance created as their domain.

VI. Related Work

The XML to DAML Translator [Aube and Post, 2001] has similar aspirations to my technique, but the two approaches are very different. Their tool provides a more formal ontology based on a schema. My approach differs because it is much simpler and intended for broader use and ontology customization. Currently the XML to DAML Translator does not support XML markup transformation, just DAML Ontology creation.

VII. Conclusions

Limiting factors to widespread acceptance of the Semantic Web are the high cost and effort to produce the necessary semantically-rich data. This poster demonstrates a simple process to form semantic markup from structured data sources. This methodology allows immediate creation of ontologies and markup to support the Semantic Web.

VIII. Acknowledgements

I would like to thank Jim Hendler, Sara Conners, as well as Brian Kettler, Terry Padgett, Pete Haglich, and Brian Barthel at ISX Corporation.

IX. References

Aube, M., Post, S., XML to DAML Translator, http://www.davincinetbook.com:8080/daml/xmltodaml/xmltodaml.html, 2001.

Bosak, J., Text Markup and the Cost of Access, http://www.nature.com/nature/debates/e-access/Articles/bosak.html, 2001.

Semantic Phone: A Semantic Web Application for Semantically Augmented Communication

Akira Sugiyama, Jun-ichi Akahani, Tetsuji Satoh
NTT Communication Science Laboratories, NTT Corporation
2-4 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0237, Japan
{sugiyama, akahani}@cslab.kecl.ntt.co.jp, satoh.tetsuji@lab.ntt.co.jp

1 Introduction

The Semantic Web will make various ontologies and Web documents annotated with metadata based on the ontologies available on the Internet. These ontologies and Web documents with metadata will enable various services for our daily activities. Aiming to enrich communication between people with Semantic Web technologies, we have been researching *Semantically Augmented Communication*. In this paper, we propose a Semantic Web application, called Semantic Phone, which provides adequate information for the contents of human-human conversation by utilizing ontologies and Web documents with metadata on the Semantic Web.

To provide adequate information on conversations between people, a system has to understand human conversation. For that purpose, it is necessary to prepare the knowledge for understanding every topic in a conversation. In traditional speech dialog systems, system designers have to prepare domain knowledge. In order to understand conversations that are not limited to a small range of topics, knowledge of various topics is required. However, it is difficult for system designers to prepare knowledge for various domains. The method of understanding conversations using the statistical technique is also proposed. However, it is difficult to build a dialog corpus of various topics.

Our basic idea is to utilize ontologies offered by Semantic Web as domain knowledge. If the Semantic Web fully spreads, we can expect that knowledge of various domains will become available. In our method, ontologies are used for knowledge of the topic to understand conversations.

In the following, we first provide an outline of Semantic Phone and propose a method for understanding conversation with the Semantic Web.

2 Semantic Phone

Semantic Phone is a Semantic Web application that provide timely adequate information according to human-human conversation (Figure 1). The application understands human-human conversation, retrieves information suitable for the contents of conversation from the Semantic Web, and presents the Web document on a browser in a timely manner. The application aims at supporting and activating communication by showing suitable information in suitable timing.

Figure 2. shows the processing flow which offers the information according to a conversation. First, speech recognition and a morphological analysis are performed on the speech to obtain the word sequence. Note that our method does not carry out any syntactic analysis nor a semantic analysis of utterances. This method is aimed at natural human-human conversation. There is much unclear pronouciation in natural conversation, so accuracy of speech recognition is not expectable. Moreover, natural conversation consists of many fragmented utterances. Therefore, syntactic analysis and semantic analysis are difficult.

The process of understanting a conversation is performed in a conversation understanding module by considering a word sequence as an input, and a conversation understanding result is outputted as an ontological instance. Section 3 explains this method briefly. The information retrieval module generates a reference formula from the instance outputted as a conversation understanding result, the reference is then carried out from Web, and a result is displayed on a browser.

By showing suitable information with sufficient promptness according to the contents of conversation, a topic may swell or it may influence further discussions.

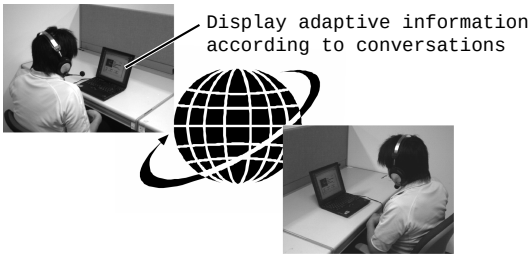


Figure 1: Application Image

3 Conversation Understanding

As mentioned above, to understand conversations, knowledge is required to understand conversations on any topic. For many speech dialog systems, the domain knowledge which understands conversation is prepared with the frame[Chu-Carroll, 1999][Nakano *et al.*, 1999]. For example, for a speech dialog system relating to a hotel reservation, a frame is made that describes the conversation's properties, such as

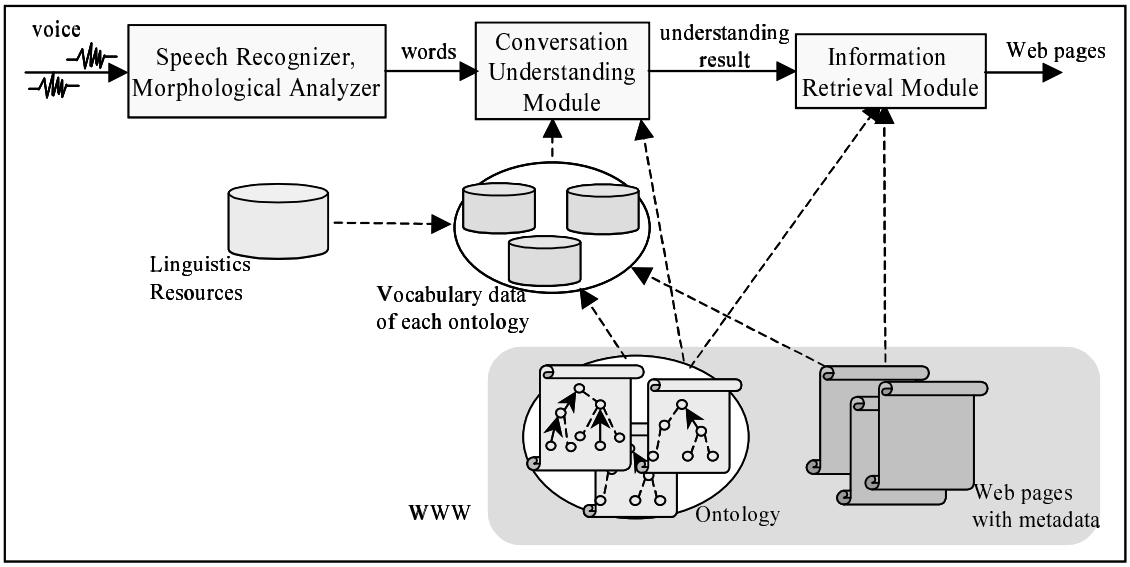


Figure 2: System Overview

the stay schedule, the number of people, and the price. In order to understand conversations that are not limited to a small range of topics, system designers have to prepare knowledge of various topics. In this research, ontology is used for topic knowledge. Although ontologies were not designed for understanding conversation, we think that they act effectively as knowledge sources for understanding conversations.

The Conversation Understanding Module collects ontologies and Web documents with metadata from WWW and constructs vocabulary data of each ontology as a pretreatment. The vocabulary of the property values is collected from a set of metadata, while that of classes and properties is collected from language resources such as a thesaurus.

At the time of execution, the module makes an instance from a word sequence which appeared in conversation based on the collection of vocabularies of ontology. The module refers to the collection of vocabularies of ontology and makes the word a value of an instance if the word is a member of the vocabularies. The module makes an instance set from every ontology, and we treat the instance set as the contents of a conversation.

4 Conclusion

We are currently studying a system that presents the information suitable for human-human conversation as an application of the Semantic Web. Although knowledge on every topic is required to understand conversation, it is difficult to build knowledge about all topics. If various ontologies come to be exhibited by the Semantic Web, an ontology can be used as a form of topic knowledge. The knowledge for every topic required for understanding conversation is built using ontology, metadata and language resources, such as a thesaurus. The result of conversation understanding performed using this knowledge is expressed in the form of an ontological instance.

We have been building a Semantic Phone prototype, and are constructing ontologies of domains, such as sightseeing, restaurants and shopping. We also collecting Web documents on such items as a store and a temple in Kyoto, and building metadata for those documents. We are planning to conduct an experiment using these data in the near future.

As future work we would like to consider how to deal with speech recognition errors, and quick topic changes. We will investigate a method of retrieving a suitable Web documents from the conversation understanding results expressed in the form of an instance and a method of showing it to suitable timing.

We are also considering applying this technology to personal ontologies and personal repositories such as mail and reports[Kamei *et al.*, 2003]. At this stage, presentation of the information that is adapted for the individual has been attained.

References

[Chu-Carroll, 1999] Jennifer Chu-Carroll. Form-Based Reasoning for Mixed-Initiative Dialogue Management in Information-Query Systems In *Proceedings of the Sixth Eurospeech*, pages 1519-1522, 1999.

[Nakano *et al.*, 1999] M. Nakano, K. Dohsaka, N. Miyazaki, J. Hirasawa, M. Tamoto, M. Kawamori, A. Sugiyama, T. Kawabata. Rich Turn-Taking in Spoken Dialogue Systems. In *Proceedings of the Sixth Eurospeech*, pages 1167–1170, 1999.

[Kamei *et al.*, 2003] K. Kamei, S. Yoshida, K. Kuwabara, J. Akahani, T. Satoh. An Agent Framework for Interpersonal Information Sharing with an RDF-based Repository In *Proceedings of ISWC2003*, 2003(to appear).

DAML Reality Check:
A Case Study of KAoS Domain and Policy Services

A. Uszok, J. M. Bradshaw, P. Hayes, R. Jeffers, M. Johnson, S. Kulkarni, M. Breedy, J. Lott, L. Bunch

Institute for Human and Machine Cognition (IHMC), University of West Florida,
40 S. Alcaniz, Pensacola, FL 32501
{auszok, jbradshaw, phayes, rjeffers, mjohnson, skulkarni, mbreedy, jlott, lbunch}@ai.uwf.edu

1 Introduction

DAML, OWL (*http://www.w3.org/2001/sw/WebOnt*), and other increasingly popular description-logic-based representations [Baader *et al.,* 2003] seem to be a natural choice to support the development of the current generation of semantically-rich software services and intelligent systems. The KAoS Policy [Damianou *et al.,* 2000] and Domain Services framework is an interesting example of this trend. By investigating its design, development, and application, we can learn much about the current state of description-logic-based representations, tools, and technology—their strengths, their gaps, and their limitations.

The implementation of the KAoS Policy framework (Fig. 1) proved to be a challenging task and required integration of the scarce existing DAML and description logic tools.

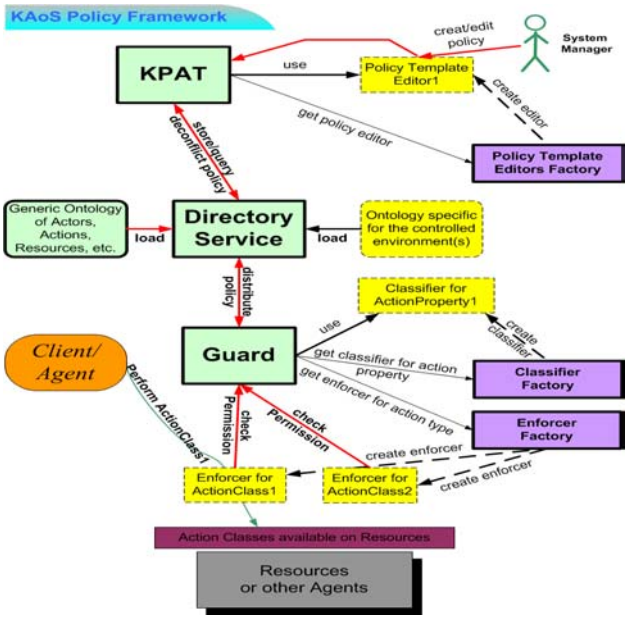


Figure 1. Architecture of the KAoS Policy Framework

The KAoS Policy Framework generic functionality includes:

- Policy ontology management,
- Creating/editing of policies using KPAT,
- Storing, deconflicting and querying policies using the Directory Service,
- Distribution of policies to Guards, which control agents' actions using Enforcers,
- Policy disclosure mechanisms.

The framework can be extended to support a specific environment by:

- Defining new ontologies describing; resources and types of actions which can be performed on them,
- Creating Plug-ins for: Policy Template editors, Enforcers controlling specific actions or with generic enforcement capability, Defining Semantic Matchers to determine if a given instance is in the scope of the given class to support specific actions.

2 Inference Engine Integration

Three inference engines were reviewed for use with KAoS: FaCT [Horrocks *et al.,* 2000], DAMLJess [Kopena *et al.,* 2002], and the Java Theorem Prover (JTP) (*http://www.ksl.stanford.edu/software/JTP*). We were looking at three main criteria: 1. degree of full DAML support, 2. adequacy of the query interface, and 3. likelihood of good support and continued development of the tool. JTP seemed the best choice at the time, and was integrated into KAoS. One problem noted early on with JTP was the time required to assert new ontologies into the inferencing engine. However, the steady improvement of JTP has led to a dramatic increase in its performance, an order of magnitude or more in some cases. Currently, loading of the KAoS core ontologies takes less than 16 seconds on Pentium III 1.20 GHz with 640 MB RAM. Adding the definition of complexity similar to the policy presented on Figure 3 takes less than 340ms.

Some of the most important features of description-logic-based policy representation and reasoning show

their advantages as part of policy analysis. Among others, these include subsumption-based reasoning, determination of disjointness, and instance classification [Baader *et al.,* 2003]. The first two features are used mainly during the kinds of analysis associated with policy administration. Instance classification is especially valuable for policy exploration, disclosure, and distribution—it is used, for instance, to determine which entities belong to a given domain or if a resource that is being accessed by a given action is within a range constrained by policy.

3 Ontology-driven System Architecture

In this section we consider the benefits and problems of using ontologies as a central aspect of system design. An ontology allows for great flexibility in design and deployment, however careful attention to performance-sensitive aspects of the system is essential. Additional problems arise at two boundaries: where the reasoning system meets the human world and where it meets the systems being governed by policy. Our approach to addressing these issues is described in this section.

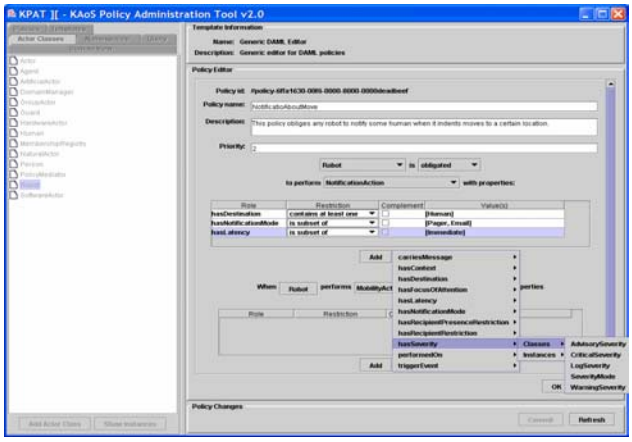


Figure 2. Graphical interface of the DAML policy editor

The KPAT graphical interface hides the complexity of the DAML representation from users and uses the Jena toolkit to build new DAML structures of policies. On the other hand, its unique user experience is achieved through the use of ontology. The user is always presented with a complete set of choices, which are valid in the given context.

The framework nature of KAoS means that the installation configuration can vary. Since the role of each software component is related to concepts defined in specialized ontologies it is relatively easy to associate these components (enforcers, classifiers, policy editors, etc.) with an appropriate ontology definition. Such mappings are registered in proper software factories, creating a new Java component on demand (see Figure 1). KAoS always checks if particular factory consists of a specialized component for handling the given ontology concept and if so, uses it instead of the generic functionality.

When a policy leaves the Directory Service it typically has to be translated from DAML into some format, which is compatible with the integrated legacy systems. KAoS communicates to the outside world using a map relating ontology properties to the name of the class defining its range as well as a list of current cached instances of that class. A particular system can use the given cached instances when assessing policies or it can refresh them by contacting the Directory Service and providing the name of the range.

4 Conclusion

We have shown that the use of description logic provides significant advantages in the design and development of a complex software system. Although some problems arose from the expressive limitations of DAML, we were able to find effective workarounds in practice, and the performance of available DAML technology has improved significantly during the course of this project. We believe that the techniques we have developed for using DAML in an agent-based application are of general utility and can be re-used in other systems. This work provides practical evidence in support of the thesis that the use of ontologies as a central paradigm in an object-oriented programming scenario is an effective design strategy.

References

[Baader *et al.,* 2003] Baader, F., Calvanese, D., McGuinness, D., Nardi, D. & Patel-Schneider, P. (Ed.) (2003). The Description Logic Handbook. Cambridge University Press,

[Damianou *et al.,* 2000] Damianou, N., Dulay, N., Lupu, E. C., & Sloman, M. S. *Ponder: A Language for Specifying Security and Management Policies for Distributed Systems*, Version 2.3., Imperial College, London, 2000

[Horrocks *et al.,* 2000] Horrocks, I., Sattler, U., Tessaris, S. and Tobies., S. How to decide query containment under constraints using description logic. In *Proceedings of LPAR'2000*,

[Kopena *et al.,* 2002] Kopena, J. DAMLJess web site. http://edge.mcs.drexel.edu/assemblies/software/damljesskb/damljesskb.html, 2002

Improving Trust and Privacy in the Semantic Web through Identity Management

Wolfgang Woerndl, Michael Galla
{woerndl,galla}@in.tum.de
Technische Universitaet Muenchen, Germany

1 Introduction

The goal of the Semantic Web activities is to make available the meaning of information to computers. Thereby, software agents or other programs analyze and evaluate semantic annotations of information items to improve services. A combination of personalization and Semantic Web technologies can be beneficial because additional semantic information to data sources can be used to improve customization of search results or other filtering services.

However, personalization also raises issues of privacy and trust. Any personalization application potentially poses privacy problems, because users have to provide information about themselves and want to know how their information is being used. In addition, the privacy of users who provide semantic annotations to information sources is concerned. There is also the problem of trust. In the existing Web, it is more or less up to the user to (manually) decide whether information, e.g. search engine results, might be trustworthy or not. In the Semantic Web, this will not be the case, because agents have to determine the trustworthiness of information themselves.

In this paper, we describe how relationship and identity management can be used as building blocks for trust and privacy in the Semantic Web. We will also briefly introduce the *Personal Information Agents (PINA)* project which tries to integrate ideas of identity management such as pseudonymity into Semantic Web agents.

2 Identity Management, Privacy and Trust

The basic idea of *identity management* is to separate user profiles and identities from the services that are using them. An identity management system allows people to define different identities, roles, associate personal data to it, and decide whom to give the data to and when to act anonymously [Koch and Woerndl, 2001]. An important aspect with regard to user modelling and identity management is to consider different roles and identities of users, for example "work" or "private" identities. Support of pseudonymity and anonymity are features of identity management that can improve user privacy.

Privacy is "the claim of individuals, groups or institutions to determine for themselves, when, how and to what extent information about them is communicated to others." [Westin, 1967]. The aspect of control for the user is essential. Users need to know how, why and what part of their identity is being accessed. It is not reasonable to build user-adaptive systems without considering privacy.

The Semantic Web approach allows "anyone to say anything about anything" [Berners-Lee, 2002] – yet it does not guarantee the truth of such statements. Like in the traditional Web, this leads to *trust* playing an important role in the Semantic Web. Trust is a subjective expectation or assumption about the behavior of another person. Thus, trust can never be imposed automatically without referring to personal evaluations or without referring to a concrete application context. Technically, the digital signature will be a central building block of the "Web of Trust".

Trust has to be addressed in combination with privacy. Therefore, the top layer in the Semantic Web layer cake by Tim Berners-Lee (available at www.w3.org/2002/Talks/09-lcs-sweb-tbl/slide19-0.html in [Berners-Lee, 2002], for example) should rather be named "Trust & Privacy" not just "Trust", because all efforts to improve trust and build a "Web of Trust" potentially decrease the privacy of users. In other words, there is a trade-off between trust and privacy in any personalization system that has to be taken into account when designing the application.

3 Towards Trust and Privacy in the Semantic Web

In the following sections we will discuss how relationship and identity management can foster trust and privacy in the Semantic Web. In our scenario, users annotate Web pages with semantic information and agents use these annotations to provide personalized services to (other) users.

3.1 Using relationship management to improve trust

Trust is a personal evaluation of another person which is mirrored in the *relationship* to that person. Traditionally, people tend to trust those people they have rather strong social relationships with. In part, trust is transitive: often, if a good friend of mine trusts a person A, then I will often tend to trust person A, too. Hence, improving trust is strongly related to social relationship management, which provides a context for trust building.

In the *Interoperable Relationship Management (InReM)* subproject, we address issues of social relationship management by developing an ontology-based formalization of social relationships as well as an agent society exchanging information about social relationships. The InReM subproject is not restricted to trust relationships, but addresses general social relationships occurring in computer supported communication environments.

Social relationships are valuable especially with respect to social capital. Most people do not want their social network to be publicly accessible. Thus the main problems addressed by the agent society are how to define access rights for relationship information and at the same time preserve privacy. Our solution proposes two strategies for resolving access requests:

1. Exploring the social network by exchanging relationship information with other agents.
2. Finding paths via transitive relationships (without exchanging relationship information) to the person requesting the information.

Besides granting or declining access, also the allowed usage of the relationship information must be specified. In our approach, relationship information may either be public (it may be distributed freely), anonymous (it may only be distributed anonymously), or private (it may not be distributed at all).

When information about the trustworthiness of some piece of information in the Semantic Web is needed, an agent can make use of relationship information by trying to derive a trust relationship to the author of the information with respect to the current context. For each context an agent possesses a set of rules defining valid derivations for trust relationships. For crucial information, the derivation may require the existence of a short chain[1] of very strong trust relationships to the author of the information, whereas for less important information, a chain of positive evaluation relationships may be sufficient. By allowing general relationships for derivations of trust relationships, our approach is more general than existing approaches for formalizing the "Web of Trust". These approaches usually use trust in public keys applied for signing information items and disregard privacy protection.

3.2 Combining the trust model with identity management

Unless there is a link of Semantic Web annotations to persons or identities, mechanisms to derive trust can not be implemented. In our approach, this link is provided by integrating identity management into the Semantic Web. Thereby, annotations are stored as part of user profiles in a federated identity management network. A user can define and control different pseudonyms to mark Semantic Web annotations. The real identity of the user does not have to be disclosed. For example, a user can provide annotations under a pseudonym "mgalla" or "foo23". Agents then derive the trustworthiness of annotations as explained above by using these pseudonyms instead of digital signatures of users. The authenticity of pseudonyms is proven by the identity management network.

Users can also control the conditions under which annotations may be accessed by agents or the linkability of pseudonyms. For example, the user can reveal that "mgalla" and "foo23" really are the same person. This information can then be used by agents to improve the results of the relationship analysis but must not be made available to other users.

In [Woerndl and Koch, 2003; Koch and Woerndl, 2001] we explain a concept for authorization of user profile accesses in this scenario. Thereby, authorization is done by combining access control with privacy enhancing technologies. User profile agents negotiate access right to user profiles (such as relationship information of the user) with service agents using privacy policies of services and preferences and access rules of users.

4 The Personal Information Agents (PINA) Project

The briefly presented solution towards trust and privacy in the Semantic Web is part of the *Personal Information Agents (PINA)* project. The goal of PINA is to bring together identity management on the one hand, and Semantic Web and agent technologies on the other hand. The purpose is to support semantic personalization of information sources and improve adaption of information to user profiles. The fundamental idea is to store references to Semantic Web annotations as part of user profiles in an identity management framework. These references can then be used to improve trust in the Semantic Web through relationship management as explained above without necessarily worsen user privacy.

In [Koch and Woerndl, 2001; Woerndl and Koch, 2003] we describe an identity management infrastructure in the domain of community support systems that can be used in our scenario. Agents can thereby access user identities and profiles via an agent-based interface (FIPA). We are currently implementing the link to Semantic Web annotations. Next steps in PINA also include implementation of more components – such as filter and personalization agents – to test the usability of our approach.

References

[Koch and Woerndl, 2001] Michael Koch and Wolfgang Woerndl: Community Support and Identity Management. *Proc. Europ. Conference on Computer-Supported Cooperative Work (ECSCW2001)*, Bonn, Germany, 2001

[Westin, 1967] Alan Westin: *Privacy and Freedom*. New York, 1967

[Berners-Lee, 2002] Tim Berners-Lee: The Semantic Web – LCS Seminar, 2002. http://www.w3.org/2002/Talks/09-lcs-sweb-tbl/

[Woerndl and Koch, 2003] Wolfgang Woerndl and Michael Koch: Privacy in Distributed User Profile Management. Poster, *The Twelfth International World Wide Web Conference (WWW2003)*, Budapest, Hungary, 2003

[1] A chain of relationships involves a sequence of persons where each person has a relationship of the specific kind with the next one. The length of the chain is determined by the number of persons involved.

Data Migration for Ontology Evolution

Zhuo Zhang, Lei Zhang, ChenXi Lin, Yan Zhao and Yong Yu
Department of Computer Science and Engineering,
Shanghai JiaoTong University, Shanghai, 200030, China.
{martinzhang,tozhanglei}@sjtu.edu.cn, wshlcx@yahoo.com,{francs,yyu}@sjtu.edu.cn

1 Introduction

Ontology is the conceptual backbone that provides meaning to data on the Semantic Web. However, ontology is not a static resource and may evolve over time. Ontology's change often leaves the meaning of data in an undefined or inconsistent state. It is thus very important to have a method to preserve the data and its meaning when ontology changes. In this paper, we propose a general method that solves the problem by migrating the data. We analyze in detail some of the issues in the method including the separation of ontology and data, the migration specification, the migration result and the migration algorithm. The paper also instantiates the general method in RDF(S) as an example. The RDF(S) example itself is a simple but complete method for migrating RDF data when RDFS ontology changes.

2 A General Method

Figure 1 is the overview of the general method. We roughly divide the method into two phases – the design phase and the implementation phase. In the design phase, we need design the separation function, design or choose migration specification language and design migration algorithm. In the implementation phase, we need capture user requirements, obtain the original data and run the migration algorithm.

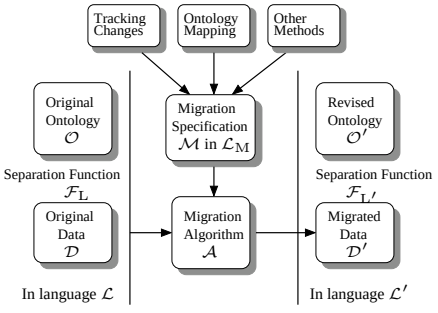


Fig.1 An overview of the general method

3 Ontology and Data

Before the actual migration, we need capture the user's notion of "data" and distinguish it from the ontology that it conforms to.

Definition 1. Given a set of sentences S in some KR language $\mathcal{L}$, the **separation function** $\mathcal{F}_{\mathrm{L}}$ produces another set of sentences $\mathcal{F}_{\mathrm{L}}(S)$ that can be considered as the data part of S. If S itself is entirely data, $\mathcal{F}_{\mathrm{L}}(S) = S$. Formally, $\mathcal{F}_{\mathrm{L}}$ is a function $\mathcal{F}_{\mathrm{L}} : 2^{\Sigma_L} \longrightarrow 2^{\Sigma_L}$ where Σ_L is the set of all sentences of language $\mathcal{L}$ and the data sets are fixpoints of $\mathcal{F}_{\mathrm{L}}$.

Definition 2. Given a set of data $\mathcal{D}$ and an ontology $\mathcal{O}$ expressed in language $\mathcal{L}$, the **conformance function** $\mathcal{K}_{\mathrm{L}}(\mathcal{D}, \mathcal{O})$ returns true iff $\mathcal{D}$ conforms to $\mathcal{O}$.

Let R denote the set of reserved vocabularies[1] defined in the RDF Model Theory document [Hayes, 2002]. For any RDF graph G as a set of triples, let $C(G)$ be the rdfs-closure of G. We can define the set of classes $CL(G)$ and the set of properties $PR(G)$ in a typical ontology layer in G as the following:

$$CL(G) \stackrel{\mathrm{def}}{=} \{\, x \mid (x, \texttt{<rdf:type>}, \texttt{<rdfs:Class>}) \in C(G) \,\wedge\, x \notin R \,\}$$

$$PR(G) \stackrel{\mathrm{def}}{=} \{\, x \mid (x, \texttt{<rdf:type>}, \texttt{<rdf:Property>}) \in C(G) \,\wedge\, x \notin R \,\}$$

and we use the notion $CPR(G)$ to denote the set of all classes, properties and reserved vocabularies:

$$CPR(G) \stackrel{\mathrm{def}}{=} CL(G) \cup PR(G) \cup R \ .$$

The data part of G then consists of triples that has a subject or object that is not in $CPR(G)$. The separation function is defined accordingly.

Definition 3. The separation function for RDF(S) is:

$$\mathcal{F}_{\mathrm{RDFS}}(G) \stackrel{\mathrm{def}}{=} \{\, (x, y, z) \mid (x, y, z) \in G \,\wedge\, (\, x \notin CPR(G) \vee z \notin CPR(G)\,)\,\} \,.$$

Note that this definition is only one possible definition of data in RDF(S) and we belive it appropriately capture the user's notion of "data" in most Semantic Web applications. Nevertheless, there could be other definitions for specific application scenarios. For notation convenience, we also define the ontology layer and language layer of G as

$$OL(G) \stackrel{\mathrm{def}}{=} C(G) - \mathcal{F}_{\mathrm{RDFS}}(C(G)).$$

We observe the following properties of the $\mathcal{F}_{\mathrm{RDFS}}$ separation function:

[1] It includes the rdfV, rdfsV, RDF reification vocabularies and RDF container vocabularies defined in the RDF Model Theory.

Lemma 1. $\mathcal{F}_{\mathrm{RDFS}}\left(\mathcal{F}_{\mathrm{RDFS}}(G)\right) = \mathcal{F}_{\mathrm{RDFS}}(G)$.
Lemma 2. $C(OL(G)) \cap \mathcal{F}_{\mathrm{RDFS}}(G) = \emptyset$.

Lemma 1 indicates that $\mathcal{F}_{\mathrm{RDFS}}(G)$ is a fixpoint for $\mathcal{F}_{\mathrm{RDFS}}$ and Lemma 2 declares that the ontology and language layer and anything inferred from that layer is disjoint with the data layer.

Definition 4. Given RDF data D and RDFS ontology O, the conformance function for RDF(S) can be defined as:

$$\mathcal{K}_{\mathrm{RDFS}}(D, O) \Leftrightarrow CL(D) \subseteq CL(O) \wedge PR(D) \subseteq PR(O) \ .$$

We also observe the following property for $\mathcal{K}_{\mathrm{RDFS}}$ and $\mathcal{F}_{\mathrm{RDFS}}$:

Lemma 3. $\mathcal{K}_{\mathrm{RDFS}}\left(\mathcal{F}_{\mathrm{RDFS}}(G),\, OL(G)\right) = \texttt{true}$.

4 Migration Specification

Migration specification is a formal description of the user's migration requirements. It actually dictates how the data semantics should be preserved.

Definition 5. A **migration specification** $\mathcal{M}$ is a set of sentences (rules) written in some language $\mathcal{L}_{\mathrm{M}}$. The sentences may use constants (e.g. classes, relations, resources) defined in either the original ontology $\mathcal{O}$, the original data $\mathcal{D}$, the revised ontology $\mathcal{O}'$, or the languages $\mathcal{L}$ or $\mathcal{L}'$.

Definition 6. Given the original ontology O and the revised ontology O' both of which are in RDFS, a **simple RDF(S) Migration Specification** M is a set of RDFS tripes $\{(x\ y\ z)\}$ in which each triple $(x\ y\ z)$ satisfies one of the following two requirements:

1. $y = \texttt{<rdfs:subClassOf>} \wedge x \in CL(O) \wedge z \in CL(O')$.
2. $y = \texttt{<rdfs:subPropertyOf>} \wedge x \in PR(O) \wedge z \in PR(O')$.

5 Migration Result

Definition 7. We define $\mathcal{P}$ to be the set of all sentences in $\mathcal{L}'$ that can be proved from the migration specification $\mathcal{M}$ and the original data $\mathcal{D}$:

$$\begin{aligned}\mathcal{P} \ \stackrel{\mathrm{def}}{=}\ & \left\{\phi \,|\, \mathcal{D} \cup \mathcal{M} \cup \mathcal{L} \cup \mathcal{L}' \vdash \phi\right\} - \left\{\phi \,|\, \mathcal{D} \cup \mathcal{L} \cup \mathcal{L}' \vdash \phi\right\} \\ & - \left\{\phi \,|\, \mathcal{M} \cup \mathcal{L} \cup \mathcal{L}' \vdash \phi\right\}\end{aligned}$$

Definition 8. The **migration result** $\mathcal{D}'$ is the largest set that satisfies the condition

$$\mathcal{D}' \subseteq \mathcal{F}_{\mathrm{L}'}\left(\mathcal{P}\right) \wedge \mathcal{K}_{\mathrm{L}'}\left(\mathcal{D}',\, \mathcal{O}'\right) \ .$$

Definition 9. Given a simple RDF(S) migration specification M and the original data D in RDF, we can define the counterpart of $\mathcal{P}$ in RDF(S) as the following function $\mathcal{P}_{\mathrm{RDFS}}$:

$$\mathcal{P}_{\mathrm{RDFS}}(D, M) \stackrel{\mathrm{def}}{=} C(D \cup M) - C(D) - C(M) \ .$$

Lemma 4. $\mathcal{F}_{\mathrm{RDFS}}\left(\mathcal{P}_{\mathrm{RDFS}}(D, M)\right) = \mathcal{P}_{\mathrm{RDFS}}(D, M)$.

Lemma 5. $\mathcal{K}_{\mathrm{RDFS}}\left(\mathcal{P}_{\mathrm{RDFS}}(D, M),\, O'\right) = \texttt{true}$.

Theorem 1. The migration result of the simple RDF(S) migration method is $D' = \mathcal{P}_{\mathrm{RDFS}}(D, M)$.

6 Related Work

Ontology evolution research (e.g. [F.Noy and Klein, 2003; Stojanovic *et al.*, 2002]) focuses on the big picture of the entire life cycle of ontology changes and studies how the evolution process can be managed. Ontology versioning research (e.g. [Klein and Fensel, 2001; Klein *et al.*, 2002]) pays attention to the relations among multiple versions of an ontology. [Heflin and Hendler, 2000] deals with the changes of ontologies in the Web environment. [Kiryakov and Ognyanov, 2002] tracks changes in RDF(S) repositories.

Another line of research that influences our work a lot is ontology mapping. Ontology mapping research (e.g. [Madhavan *et al.*, 2002]) struggles to find ways to (semi-)automatically discover the semantic relations between ontologies. Data semantics can be preserved through the mapping without changing the old data. Our method complements it by migrating the data which may provide better runtime performance and cleaner data.

References

[F.Noy and Klein, 2003] Natalya F.Noy and Michel Klein. Ontology evolution: Not the same as schema evolution. *Knowledge and Information Systems*, 5, 2003.

[Hayes, 2002] Patrick Hayes. RDF model theory. Working draft, W3C, Apr. 2002.

[Heflin and Hendler, 2000] Jeff Heflin and James Hendler. Dynamic ontologies on the web. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000)*, pages 443–449, Menlo Park, CA, USA, 2000. AAAI/MIT Press.

[Kiryakov and Ognyanov, 2002] Atanas Kiryakov and Damyan Ognyanov. Tracking changes in RDF(S) repositories. In *Proceedings of the EKAW 2002*, pages 373–378, Siguenza, Spain, Oct. 2002. Springer.

[Klein and Fensel, 2001] Michel Klein and Dieter Fensel. Ontology versioning for the semantic web. In *Proceedings of the 1st International Semantic Web Working Symposium (SWWS'01)*, pages 75–91. Stanford University, Aug. 2001.

[Klein *et al.*, 2002] Michel Klein, Dieter Fensel, Atanas Kiryakov, and Damyan Ognyanov. Ontology versioning and change detection on the web. In *Proceedings of the EKAW 2002*, pages 197–212, Siguenza, Spain, Oct. 2002. Springer.

[Madhavan *et al.*, 2002] J. Madhavan, P.A.Bernstein, P.Domingos, and A.Y.Halevy. Representing and reasoning about mappings between domain models. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI2002)*, Edmonton,Canada, 2002.

[Stojanovic *et al.*, 2002] Ljiljana Stojanovic, Alexander Maedche, Boris Motik, and Nenad Stojanovic. User-driven ontology evolution management. In *Proceedings of the EKAW 2002*, pages 285–300, Siguenza, Spain, Oct. 2002. Springer.