The 6th International Semantic Web Conference and the 2nd Asian Semantic Web Conference

ISWC+ASWC 2007

Workshop 4

First Industrial Results of Semantic Technologies

Workshop Organizers:

Roberta Cuel, Lyndon Nixon, Claudio Bergamini

11 Nov. 2007
BEXCO, Busan KOREA

ISWC 2007 Sponsor

Emerald Sponsor

Gold Sponsor

Silver Sponsor

We would like to express our special thanks to all sponsors

ISWC 2007 Organizing Committee

General Chairs

Riichiro Mizoguchi (Osaka University, Japan)

Guus Schreiber (Free University Amsterdam, Netherlands)

Local Chair

Sung-Kook Han (Wonkwang University, Korea)

Program Chairs

Karl Aberer (EPFL, Switzerland)

Key-Sun Choi (Korea Advanced Institute of Science and Technology)

Natasha Noy (Stanford University, USA)

Workshop Chairs

Harith Alani (University of Southampton, United Kingdom)

Geert-Jan Houben (Vrije Universiteit Brussel, Belgium)

Tutorial Chairs

John Domingue (Knowledge Media Institute, The Open University)

David Martin (SRI, USA)

Semantic Web in Use Chairs

Dean Allemang (TopQuadrant, USA)

Kyung-Il Lee (Saltlux Inc., Korea)

Lyndon Nixon (Free University Berlin, Germany)

Semantic Web Challenge Chairs

Jennifer Golbeck (University of Maryland, USA)

Peter Mika (Yahoo! Research Barcelona, Spain)

Poster & Demos Chairs

Young-Tack, Park (Sonngsil University, Korea)

Mike Dean (BBN, USA)

Doctoral Consortium Chair

Diana Maynard (University of Sheffield, United Kingdom)

Sponsor Chairs

Young-Sik Jeong (Wonkwang University, Korea)

York Sure (University of Karlsruhe, German)

Exhibition Chairs

Myung-Hwan Koo (Korea Telecom, Korea)

Noboru Shimizu (Keio Research Institute, Japan)

Publicity Chair: Masahiro Hori (Kansai University, Japan)

Proceedings Chair: Philippe Cudré-Mauroux (EPFL, Switzerland

Metadata Chairs

Tom Heath (KMi, OpenUniversity, UK)

Knud Möller (DERI, National University of Ireland, Galway)

Organising Committee

Lyndon Nixon (Free University Berlin, Germany)

Roberta Cuel (University of Trento, Italy)

Claudio Bergamini (Imola Informatica, Italy)

Program Committee

Chris van Aart, Sogeti, The Netherlands

Richard Benjamin, ISOCO, Spain

Elmar Dorner, SAP AG, Germany

Roberta Ferrario, ISTC-CNR, Italy

Christian Fillies, Semtation Gmbh, Germany

Tim Geisler, webXcerpt Software GmbH, Germany

Ruben Lara, Tecnologia, Informacion and Finanzas, Spain

Andreas Persidis, Biovista, Greece

Jean Rohmer, Thales group, France

Hans-Peter Schnurr, Ontoprise and Customer, Germany

Paul Warren, British Telecom, UK

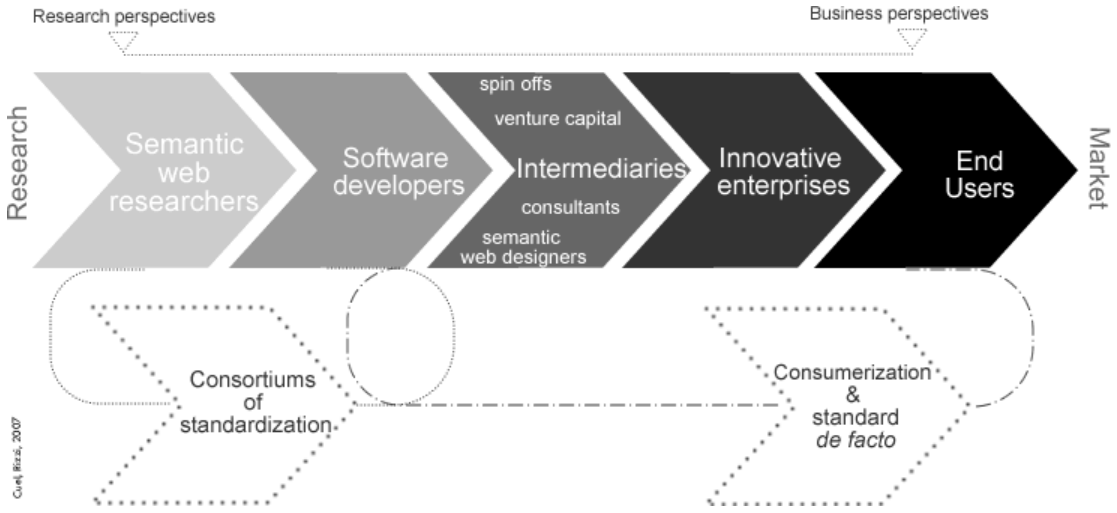
David Wood, Zepheira LLC, USA

Foreword

The goal of the FIRST workshop is to demonstrate the early adoption of semantic technologies by enterprises, and to promote further technology transfer from Semantic Web research to industry.

Important issues of technology maturity, accessibility, integration into existing enterprise IT structures and maintenance throughout the entire ontology and metadata lifecycle must be resolved before semantic technologies can become part of mainstream enterprise IT solutions. However, we fully expect that when this is the case, more flexible, efficient and dynamic business processes can be enabled, leading to higher market visibility, competitiveness and cost reductions for semantic technology adopters.

Semantic Web researchers are becoming more aware of the need to direct their research towards business needs and industry is becoming more aware of the value of these technologies for their activities. There are many primary actors that are interested in the Semantic Web and all of them can influence its development both from the research and business perspectives, at various positions in the Semantic Web value chain.



Stakeholders and value chain of the Semantic Web.
Source: Knowledge Web Technology Roadmap 2007.

The figure shows the value chain of semantic web evolution, connecting researchers (who mainly produce semantic web theories and methods), computer science firms (which mainly produce solutions), firms and end-users, which develop, adopt, make use of semantic web technologies:

- Semantic Web researchers are directly involved in European and international projects, are developing and innovating theories and methods in the Semantic Web (i.e. Semantic Web languages, Semantic Web services or algorithms to deal with reasoning, scalability, heterogeneity, and dynamics). In order to validate the resulting theories, researchers often develop application prototypes which are experimented within innovative firms.
- Consortiums of standardization are interested in sorting out new recommendations and standards on Semantic Web technologies, providing the basis for innovative tools.
- Software developers are interested in developing Semantic Web solutions and applications. They can directly sell the latter to firms and, at the same time, test innovative theories and methods.
- Intermediaries are interested in transferring technology and knowledge from researchers and developers to practitioners. Intermediaries can assume the form of Semantic Web designers, consultants, venture capitals, spin offs, etc.
- Innovative enterprises are interested in catching new opportunities from Semantic Web, also developing new business models.
- End-users are interested in obtaining useful and effective solutions. One of the most important requisites is to use a transparent technology - “No matter what is behind, if it works”. Although end-users are positioned at the end of the value chain, their needs are very relevant for all the Semantic Web stakeholders.

- Standardization de-facto. End-users should be considered as a central element in the production of social and semantic based technology, thus should be strongly connected with all the other value chain actors. Also, the consumerization phenomenon is unveiling new standards, which are de-facto substitution consortiums of standardization.

Nowadays, thanks to several European projects developed in different fields and semantic technologies conferences, a lot of companies are considering Semantic Web technologies as very challenging, and are starting to test these technologies into real applications.

This workshop is part of these efforts to bring industry and research closer together and facilitate increasing uptake of semantic technologies in the enterprise. We hope you will find the FIRST workshop useful and profitable!

Yours,

FIRST workshop organising committee

Acknowledgements

The organisers wish to thank the EU Network of Excellence KnowledgeWeb for the support given to this workshop.

More about the KnowledgeWeb Industry activities can be found at

http://knowledgeweb.semanticweb.org/o2i

Table of Contents (papers in order of workshop presentation)

A Case Study of an Ontology-Driven Dynamic Data Integration in a Telecommunications Supply Chain.

Aidan Boran[1], Declan O'Sullivan[2], Vincent Wade[2]

[1] Bell Labs Ireland, Alcatel-Lucent, Dublin, Ireland
[2] Centre for Telecommunications Value-Chain Research, Knowledge and Data Engineering Group, Trinity College, Dublin, Ireland.
{aboran@alcatel-lucent.com, declan.osullivan@cs.tcd.ie, vincent.wade@cs.tcd.ie}

Abstract. Data Integration refers to the problem of combining data residing at autonomous and heterogeneous sources, and providing users with a unified global view. Ontology-based integration solutions have been advocated but for the case to be made for real deployment of such solutions, the integration effort and performance needs to be characterized. In this paper, we measure the performance of a generalised ontology based integration system using the THALIA integration benchmark. The ontology based integration solution is used to integrate data dynamically across a real telecommunications value chain. An extension of the THALIA benchmark, to take account of the integration effort required, is introduced. We identify the issues impacting the ontology based integration approach and propose further experiments.

Keywords: Data Integration, Ontology, Semantic Integration, Interoperability, Information Integration

1 Introduction

Data Integration refers to the problem of combining data residing at autonomous and heterogeneous sources, and providing users with a unified global view [1]. Due to the widespread adoption of database systems within the supply chains of large enterprises, many businesses are now faced with the problem of islands of disconnected information. This problem has arisen since different (often multiple) systems are employed across different functional areas of the supply chain (e.g. sales, production, finance, HR, logistics).

Consolidation within the telecommunications industry has also driven the need for fast and agile integration of the supply chains since all consolidations are expected to undertake efficiencies in common areas. In this environment data and information integration has become a key differentiator.

While existing data integration solutions (e.g. consolidation, federation and replication systems) are capable of resolving structural heterogeneities in the underlying sources, they are not capable of semantic integration [13]. Additionally,

our telecoms supply chain demands that our integration solution be able to cope with change in the underlying data sources in a flexible and automatic way.

The objectives of this work are (i) identify the generalised ontology based integration approach (ii) measure the integration performance of this approach using supply chain use case (iii) identify the issues which would impact an industrial deployment.

Our generalised ontology based approach consists of upper and lower ontologies connected via ontology mappings. The THALIA [2] integration benchmark system, supplemented with a classification of effort required to implement the various benchmark tests, was used to measure the integration performance of the system. Our findings shows that our initial ontology based approach although feasible does not in its current form offer significant improvements over schema based approaches. However, based on this initial experience we believe that ontology based approach holds greater promise in the long term, and we identify in our conclusions key issues that need to be addressed in order for an enhanced ontology based approach to emerge.

We conclude by highlighting the key issues and discuss future work to conduct a set of experiments to validate our solutions to provide significant value add using an ontology approach.

2 Problem Domain

Supply chains of large companies are typically comprised of many IT systems which have developed over time to support various supply chain functions (e.g. Customer Relationship Management, Demand Forecasting, Production, and Logistics). Each stage of a product's life is managed by one or more IT systems. While these systems have introduced many productivity improvements in their individual areas, they have also contributed to the creation of separate islands of data in the enterprise.

An important part of many supply chains is Product Lifecycle Management (PLM). Product Lifecycle Management is a supply chain process which manages an enterprises' products through all stages of their life from initial sales opportunity, demand forecasting, product realisation, manufacturing, delivery to customer and support to end of life. It is within this area of our supply chain, we have identified data consistency and visibility issues between the systems which manage the Sales and Forecasting part of the product lifecycle. Lack of consistency can lead to failure to deliver on time or excess inventory.

To migitate any risk associated with lack of consistency between sales and forecasting views of the PLM, organisations attempt to balance forecasting and sales opportunities [3]. In our supply chain, these risks are managed using a manual integration of financial information from each system. The report that is produced by this manual integration supplements the financial information with an integrated view of the customers and products. This involves a lot of manual steps to export data from

the databases and rework with a spreadsheet where the various heterogeneities are resolved manually. This serves as our use case in this paper.

From [4], this PLM use case presents the following integration challenges:

Structural heterogeneities: The data sources contain concepts and properties which have different granularity levels and require transformation or aggregation to create the integrated view.

Semantic heterogeneities: The data sources contain concepts and properties which have both same name with different meanings or different names with same meanings.

Additionally, we add the following challenge from our domain.

Data source changes: It is expected that new data sources can be added and existing data sources can be changed.

3 Related Work

Current industrial data integration system fall into three categories: federation systems, replication systems and consolidation systems. While each of these serve a market need, they tend to operate at the syntactic level and thus do not deal with semantic heterogeneities in the underlying sources.

Research effort is now focused on the semantic integration problem. From [5], semantic integration has three dimensions: mapping discovery, formal representations of mappings and reasoning with mappings.

Mapping representations have been created such an INRIA [6], MAFRA [7]. Mapping tools (CMS [8], FCA-Merge [9]) have been created which allow mappings to be manually or semi-automatically created.

Some current commercial data integration systems provide some level of semantic information modeling and use mapping to provide connectivity to the data sources. Contivo [10] provides an enterprise integration modeling server (EIM) which contains various enterprise vocabularies. The Unicorn workbench [11] [1] provides a schema import capability which allows different schema to be mapped to a central enterprise model. Software AG [12] develops Information Integrator which provides an upper level ontology which is mapped manually to lower data source ontologies. Mappings in the Information Integration system support both semantic and structural conversions of data.

Our research differs from the above since we carry out a benchmark of the ontology approach using real industrial data. We focus on scalability and adaptivity issues with

[1] Unicorn is now part of IBM and is to be integrated in IBM's Websphere product.

the approaches which depend on mappings. Furthermore we are researching techniques which will reduce the dependence on mappings by supplementing the metadata available in the upper and lower ontologies and therefore providing better inference capabilities.

4 Ontology Based Solution

The use of ontologies to tackle data integration problems holds promise [5,13]. It has been shown that an ontology being a formal and explicit specification of a shared conceptualization [14] is ideal for allowing automated and semi-automated reasoning over disparate and dispersed information sources. In the context of data integration, ontologies support data integration in five areas (i) representation of source schemas, (ii) global conceptualization, (iii) support for high level queries, (iv) declarative mediation and (v) mapping support [15].

4.1 Integration Implementation

We adopt a hybrid ontology approach[15, 19] for our generalised ontology based approach to integration (see Fig. 1). This consists of an upper ontology which contains a high level definition of the business concepts used by the sales and forecasting professionals, lower ontologies which lifts the database schema to a resource description framework (RDF) format. The upper and lower ontologies are connected using mappings based on the INRIA [6] mapping format.

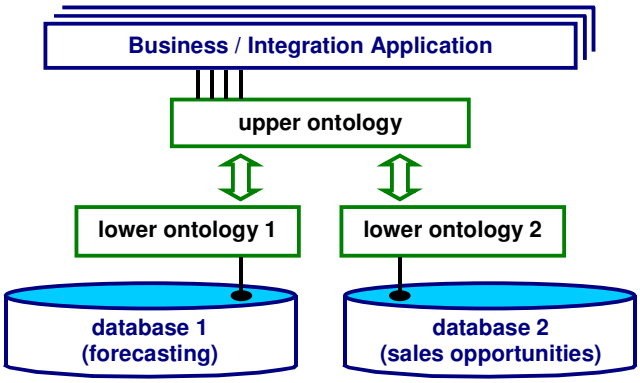


Figure 1 - Integration System Architecture

The hybrid approach was selected since it offers improvements in implementation effort, support for semantic heterogeneities and adding and removing of source over the single or multiple ontology approaches [19].

Upper Ontology

The upper ontology (figure 2) was developed by gathering information about each domain from three supply chain professionals, one working on forecasting, one working on sales and one working on the current manual integration of the systems. Each professional summarised their domain understanding in a short précis. These descriptions were used to create a common view of the sales and forecasting area. By extracting the concepts and relations described in the précis an ontology was developed in Web Ontology Language (OWL) using The Protégé development kit [16]. Ontologies are instantiated in the integration application using the Jena API [17]. The ontology contains 8 classes, 20 datatype properties and 5 object properties.

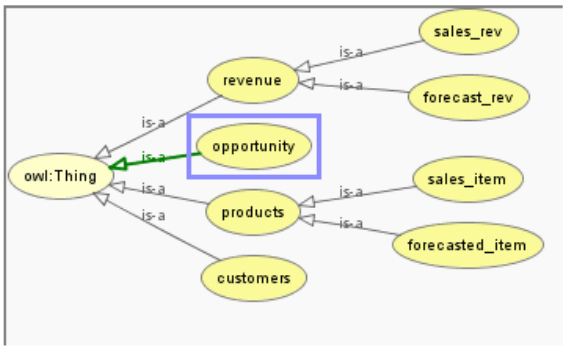


Figure 2 – Class View, Upper Ontology

Lower Ontologies

The lower ontologies lift the basic database schema information into RDF using D2RQ API [18]. This allows for automatic generation of the ontologies from the databases and once instantiated in a JENA model, the lower ontologies can be queried using SPARQL. The D2RQ API automatically converts the SPARQL queries to SQL and returns a set of triples to the caller. The lower ontologies contains classes and properties for each of the underlying database schema items and are accessed through a set of mapping files automatically created by the D2RQ API.

Mappings

A bespoke mapping implementation was created which is based on the INRIA format but additionally allows a Java function to be called to execute a complex mapping. The mappings used in this prototype support simple equivalence mappings (class to class, property to property), union type mappings (propA is the union of propB and propC) and complex conversion mappings (propA can be converted to propB using relation AB). In this prototype, relations are encoded as standalone Java functions. A

complex mapping (to sum three revenue fields into one) with a function specified looks like:

Entity1=http://someUrl/upperontology/#forecast_reveneue_q1
Entity2=http://someUrl/lowerontology/#forecast_revenue_m1,
http://someUrl/lowerontology/#forecast_revenue_m2,
http://someUrl/lowerontology/#forecast_revenue_m3,
Relation=function
FunctionHandle=sum_revenues

Ontology and Database Query

Ontologies are instantiated in the integration application using JENA ontology model. The ARQ (SPARQL) API is used to generate queries on the upper and lower ontologies.

Integration Process

Referring to figure 1, the integration process proceeds as follows:

- **Integration goal**: An integration goal is specified by the user or application. In our test system, the goal is hard coded into the application. The integration goal specifies what the users or applications wish to integrate and contains the concepts to integrate and the data needed to select the information (the key information).
- **Discovery:** Using a SPARQL query on the upper ontology, each concept in the goal is supplemented with the properties available for that concept. (e.g. customer_info concept 'becomes' customer_name, customer_id, customer_region etc…)
- **Mapping:** Mappings are now applied to the concept and property names to generate SPARQL queries on the lower ontologies.
- **Data Query:** Output from the mappings step is a sequence of SPARQL queries which are run against the lower ontology. These queries are in turn converted to SQL by the D2RQ API.
- **Results:** Each requested property and the properties value is returned to the application. In our test system we have no semantics to help us construct a formatted report so a simple list of attribute names and values are returned.

5 Experimental setup and results

This work has adopted an experimental approach to help identify the issues that would impact the deployment of an ontology based integration system (or service) in a large enterprise consisting of many heterogeneous data systems which are subject to change. We use the THALIA queries as a proxy for the type of changes we might need to accommodate.

Having identified the general ontology based approach (section 4), the remaining objectives of the experiment were:

- Measure the integration performance of the system using the THALIA queries.
- Using a generalised approach identify the primary issues which would impact an industrial deployment.

5.1 Benchmark

THALIA (**T**est **H**arness for the **A**ssessment of **L**egacy information **I**ntegration **A**pproaches) is a publicly available and independently developed test bed and benchmark for testing and evaluating integration technologies. The system provides researchers and practitioners with downloadable data sources that provide a rich source of syntactic and semantic heterogeneities. In addition, the system provides a set of twelve benchmark queries for ranking the performance of an integration system [2]. A simple score out of twelve can be assigned to an integration system based on how many of the 12 THALIA tests the system can integrate successfully. In this work, we extended the THALIA system by introducing a simple effort classification system so that each query result in THALIA could be assigned an effort estimate based on how automatic the solution is. From a maintenance viewpoint, we feel this is an important factor since it defines how well the system will perform in a changing industrial environment. We have summarised the 12 queries below:

Query1: Synonyms: Attributes with different names that convey the same meaning

Query2: Simple Mapping: Related attributes in different schemas differ by a mathematical transformation of their values. (E.g. Euros to Dollars)

Query3: Union Types: Attributes in different schemas use different data types to represent the same information.

Query4: Complex Mapping: Related attributes differ by a complex transformation of their values.

Query5: Language Expression: Names or values of identical attributes are expressed in different languages.

Query6: Nulls: The attribute value does not exist in one schema but exists in the other

Query7: Virtual Columns: Information that explicitly provided in one schema is only implicitly available in the other schema.

Query8: Semantic Incompatibility: A real-world concept that is modeled by an attribute does not exist in the other schema

Query9: Same Attributes exist in different structures: The same or related attributes may be located in different position in different schemas.

Query10: Handling Sets: A set of values is represented using a single, set-valued attribute in one schema vs. a collection of single-valued hierarchical attributes in another schema

Query11: Attribute name does not reflect semantics: The name does not adequately describe the meaning of the value that is stored.

Query12: Attribute composition: The same information can be represented either by a single attribute or by a set of attributes.

5.2 Experimental Setup

This work focused on two databases in the supply chain. The first is an Oracle based system which manages sales opportunities. It contains high level product and financial information and detailed customer information. This system has 58 tables and over 1200 attributes. The second system is a Sybase based system which manages product forecasting. It contains high level customer information but detailed product and financial information. This system has 50 tables with over 1500 attributes.
Since these systems are so large, each database schema was examined to extract the tables and data that were relevant to the integration use case and this reduced data set was recreated in two mySQL databases. The integration use case allowed us to reduce that original dataset (tables and properties) to only that data used in the use case. For example, one database also contains multiple levels of customer contact detail which is not relevant to the integration use case This reduced the data sizes to 8 tables for each database. All schema and real data from the original databases were preserved in the mySQL versions. To allow the full THALIA to be run, the databases needed to be supplemented by additional complexity in three areas (language expression and virtual columns, nulls – see table 1).

This use case involves the integration of financial information from each system by opportunity and supplementing this financial information with an integrated view of the customers and products. Real customer data was loaded into the mySQL database to run the use case.

Here is a sample of the key heterogeneities that exist in the underlying data:

- Structural – Simple conversions
  - Example 1: currency units in one schema need to be converted to a different unit in the second schema.

- Structural – 1-n relations

  A single product (high level description) in one schema is represented by a list of parts (low level description) in the second schema. For example a product at the sales database is defined as “ADSL Access Platform”, in the forecasting database this is broken down into many parts (frames, cards, cabinets)

- Structural - complex conversions

- Example 1: Revenue figures in one schema are stored monthly compared with quarterly revenue in other schema. The upper ontology deals with quarterly revenue and a conversion (summing) of monthly to quarterly revenue needs to occur.
- Example 2: "Long codes" used in one schema are comprised of three subfields in the second schema

- Semantic - Different class and property names conveying same information
  - Example 1: Upper ontology has a class called "customers" with properties "name", "id" and "region". Lower ontologies have classes "custs", "account" and properties "name", "id" and "FTS-Tier"

- Semantic - Same property name conveys different information
  - Example: product_id is used in both the lower schemas but conveys different information with different granularity

5.3 THALIA Benchmark results

This section contains the results related to the objective to measure the performance of our approach using our supply chain use case.

With respect to the THALIA integration system using our generalised approach, we can achieve 50% automated integration (6/12 tests passed). A test is deemed to have passed if the integration system can perform the integration in at least a semi-automatic way. Table 1 below shows the detailed results:

Table 1: THALIA Integration Benchmark Results

Test	***Result***	***Effort***
1. Synonyms	PASS	Semi Automatic
2. Simple Mapping	FAIL	Manual
3. Union Types	PASS	Semi Automatic
4. Complex Mapping	FAIL	Manual
5. Language Expression	PASS	Semi Automatic
6. Nulls	PASS	Fully Automatic
7. Virtual Columns	FAIL	Manual
8. Semantic Incompatibility	PASS	Semi Automatic
9. Same Attribute in different Structure	FAIL	Manual
10. Handling Sets	FAIL	Fail
11. Attribute names does not define semantics	PASS	Semi Automatic
12. Attribute Composition	FAIL	Manual

Efforts are categorised as follows:

- Fully Automatic: no code, mapping or ontology changes needed.
- Automatic: Automatic regeneration of ontology or other non code artefact
- Semi Automatic: A mapping or other non code artefact needs to be changed manually

- Manual: Non core code artefact needs to be changed/added manually
- Fail: core code changes needed.

(Note: this is an extended method of classification that is not part of the core THALIA system)

In total, 31 mappings were needed to implement the use case. Of these, 21 mappings were simple (e.g. point to point relations) between ontologies and the remaining 10 were complex mappings requiring supporting 'function code' to be written.

As table 1 indicates, tests 2,4,7,9 and 12 fail. This was because they required conversions to be constructed which in turn required some mapping code to be produced. Examples of these are:

-In one schema, product id is encoded in a longer representation called "longcode" and the product-id needs to be extracted (test 7).

Tests 1,3,5,8 and 11 require a mapping to be created which does not require any mapping conversion function to be written. Examples of these are:

- customer_name in one ontology is mapped to cust_name in another (test 1)
-product_description in the upper ontology are the union of product information in the lower ontologies (test 3).
-customer_region in one ontology is mapped to "client" (test 5)

Test 10 fails outright since it would require changes to the integration system code itself.

5.4 Findings

Complex mappings create tight coupling.

It was found that a third of the heterogeneities in our database required complex mappings to be created (tests 2, 4, 7, 9, 12). Unfortunately these complex mappings create a tighter coupling between the upper and lower ontologies than is desirable since the complex conversion functions that need to be written tend to require information from both the upper and lower ontologies. For example, a complex mapping is needed to sum the revenue for three months from the lower ontology into a quarterly value for the upper ontology; however the function specification for this summation needs to know which lower ontology resource to obtain the monthly value from.

Furthermore, the number of mappings required will grow as different integration use cases are implemented since different data properties may need to be mapped between the lower and upper ontologies.

The abstraction level of the upper and lower ontologies also negatively impacts the coupling. At the lower ontology, we have very low abstraction (few semantics) ontology and at the upper ontology we have a high abstraction (domain conceptualization). This forced some aspects of the integration to be resolved in the

application and not in the ontologies or mappings. For example, there are a number of cases where a property could be used to find other properties (opportunity id allows us to find a customer id which allows us to find a customer name). However, given the opportunity id, we currently do not encode this linkage in the ontology or in the mappings.

We believe therefore that this generalised ontology approach does not offer significant improvements over a schema based approach.

Limited Reasoning in the upper ontology
The current upper ontology is essentially a high level schema and thus provides little opportunity to engage in inference. Its primary purpose is to provide the global conceptualization. We wish to reason about the following items:

1) Is the integration goal resolvable using the current mappings and ontologies.
2) Given a property, we wish to infer what other properties are accessible. This will reduce the number of complex mappings needed.

Workflow
Given the current architecture, we have very limited semantics to allow the decomposition of an integration goal into a sequence of queries and/or inferences. For example, given an opportunity id, and wishing to retrieve product, customer and financial information for that opportunity provides us with the following steps in an integration workflow:

Integration Workflow(i) :

1) *test if product, customer and financial information are accessible using "opportunity id"*
2) *Discover what " properties" are available for product, customer and financial information*
3) *Invoke mappings to retrieve "properties" from the data sources through the lower ontology (data source ontologies) and carry out any conversions required by the mappings*
4) *Structure the returned information based on the integration goal(e.g. 1-n relations between product descriptions in different databases)*

6 Conclusions

Data integration between two different databases in a telecommunications supply chain was identified as a use case for the application of ontology based data integration. The paper describes an experimental investigation of key aspects of such a data integration process, applied to real-world datasets, and based on a measurement of the integration performance using the THALIA framework. The implemented integration service allowed automatic integration in 6 of the 12 tests supported by

THALIA. The THALIA system was enhanced to incorporate a simple effort estimate (effort column table 1).

Using RDF for the lower ontologies allowed schema changes in the databases to be automatically propagated to lower ontologies using the D2RQ api. These changes can then be incorporated into the system using mappings. The system can also cope with semantic heterogeneities as defined by the THALIA system (test 8). In spite of these benefits, the test results illustrate that in the generalised architecture, the mappings create a coupling between the upper and lower ontology that impacts scalability and provides little improvement over what would be possible with a traditional schema based approach. The results show the importance of encoding extra semantics (metametadata) in the upper ontology since this metadata can be used to resolve heterogeneities and move what are currently manually created (complex) mappings to semi-automatic mappings.

In order to improve the automation level of the generalised integration system, future research should enhance the presented approach in the following directions:

a) To help reduce the dependence on mappings, a fundamental change is needed in the integration (ontology) so that it contains 'integration metadata' and is not simply an upper data definition --- that is information that will support integration and is not just a high level schema.

b) We wish to run inference over the upper ontology to 'decide' if the integration goal is achievable or not, and if it is achievable we need to compose a set of steps to carry out the integration. For this problem, we propose to integrate a lightweight workflow vocabulary into the application which will allow explicit and external definition of the steps required to run any integration.

In our next experiments, we propose to enhance our existing system to conduct experiments in both of the areas above.

Acknowledgements

This work has received funding from the Industrial Development Authority, Ireland. This work is partially supported by Science Foundation Ireland under Grant No. 03/CE3/I405 as part of the Centre for Telecommunications Value-Chain Research (CTVR) at Trinity College Dublin, Ireland.

References

[1] A. Y. Halevy, "Answering Queries using views: A Survey," *The VLDB Journal*, vol. 10(4), pp. 270-294, 2001.

[2] M. Stonebraker, *THALIA - Integration Benchmark*, Presentation at ICDE 2005, April 6, 2005.

[3] M. Gilliland, "Is Forecasting a waste of time?" *Supply Chain Management Review*, July/August 2002.

[4] A.P. Sheth, "Changing focus on interoperability in information systems: From system, syntax, structure to semantics. M. F. Goodchild, M. J. Egenhofer, R. Fegeas, and C. A. Kottman (eds.).

[5] N. F. Noy, "Semantic Integration: A survey of Ontology Based Approaches," *SIGMOD Record*, vol. 33(4), Dec 2004.

[6] INRIA, A format for ontology alignment. http://alignapi.gforge.inria.fr/format.html

[7] MAFRA: http://mafra-toolkit.sourceforge.net/

[8] Crosi Mapping System: http://www.aktors.org/crosi/deliverables/summary/cms.html

[9] G. Stumme, A. Madche, "FCA-Merge: Bottom-up merging of ontologies," *7th Int. Conf. on Artificial Intelligence (IJCAI '01)*, Seattle, WA, pp. 225-230, 2001.

[10] Contivo, Semantic Integration : How you can deliver business value, today. http://www.contivo.com/infocenter/SI.pdf

[11] J. de Bruijn, H. Lausen, "Active ontologies for data source queries," *Proc. first European Semantic Web Symposium (ESWS2004)*, LNCS no. 3053, Springer, Heidelberg, 2004.

[12] J. Angele, M. Gesmann, *Data integration using semantic technology: a use case*, Software AG.

[13] M. Uschold, M. Gruninger, "Ontologies and Semantics for Seamless connectivity," *SIGMOD Record*, Vol 33(4), Dec 2004.

[14] T. R. Gruber, "A Translation Approach to Portable Ontology Specification," *Knowledge Acquisition*, vol. 5(2), pp. 199-220, 1993.

[15] I. F. Cruz, H. Xiao, "The Role of Ontologies in Data Integration," *Journal of Engineering Intelligent Systems*, vol. 13(4), pp. 245-252, 2005.

[16] Protégé Ontology Development tool. http://protege.stanford.edu/overview/

[17] Jena Semantic Web Framework: http://jena.sourceforge.net/

[18] D2RQ API: http://sites.wiwiss.fu-berlin.de/suhl/bizer/D2RQ/

[19] H. Wache et al. Ontology-Based Integration of Information – A survey of existing approaches. In Proceedings of the IJCAI-01 Workshop on Ontologies and Information Sharing, 2001.

A Semantic Data Grid for Satellite Mission Quality Analysis

Reuben Wright[1], Manuel Sánchez-Gestido[1], Asunción Gómez-Pérez[2], María S. Pérez-Hernández[2], Rafael González-Cabero[2], Oscar Corcho[3]

[1] Deimos Space, Ronda de Poniente, 19, 28760 Tres Cantos (Madrid), Spain
{reuben.wright, manuel.sanchez}@deimos-space.com
[2] Facultad de Informática, Universidad Politécnica de Madrid, Spain
{asun, mperez}@fi.upm.es, rgonza@delicias.dia.fi.upm.es
[3] School of Computer Science, University of Manchester, England
oscar.corcho@manchester.ac.uk

Abstract. The use of Semantic Grid architecture eases the development of complex, flexible applications, in which several organisations are involved and where resources of diverse nature (data and computing elements) are shared. This is the situation in the Space domain, with an extensive and heterogeneous network of facilities and institutions. There is a strong need to share both data and computational resources for complex processing tasks. One such is monitoring and data analysis for Satellite Missions and this paper presents the Satellite Mission Grid, built in the OntoGrid project as an alternative to the current systems used. Flexibility, scalability, interoperability, extensibility and efficient development were the main advantages found in using a common framework for data sharing and creating a Semantic Data Grid.

Keywords: S-OGSA, WS-DAIOnt-RDF(S), OWL, Satellite Mission, Grid, Space Domain, CCSDS

1 Introduction

We describe here the development of a system to enable flexible monitoring and investigation of the operation of satellite missions. We look briefly at the industry and its general operation, then in more detail at the requirements for such a system. The technical issues encountered, and solved, are given in detail and then a summary of the key advantages of the semantic approach taken. The paper concludes by considering the next steps to be taken in uptake of this system, and semantic technologies more generally in the space domain.

2 Requirements

The scope was to replicate some features of an existing Satellite Mission Quality Analysis program and to demonstrate some extensions. The analysis centres on the comparison of planned activity against the production of data by the satellite and the

further processing of that data in ground systems. The features that interest us are in the specific files, but these files are great in number, and size, and the metadata is locked into implicit forms. The approach was to extract this metadata to build an explicit semantic representation of the data which allows both the existing analysis and analysis not yet possible with that system. We will describe in this section the industry, the existing system, and use cases which describe the behaviour of this semantic system.

Earth Observation Satellite Systems Earth Observation can be defined as the science of getting data from our planet by placing in orbit a Hardware/Software element with several observation instruments, whose main goal is to obtain measurements from the Earth surface or the atmosphere. These scientific data are sent to Ground Stations and then processed in order to get meaningful information.

The working of an Earth Observation Satellite System consists of a simple process repeated over time. The instruments on board the satellite act like cameras that can be programmed taking "pictures" (images) of specific parts of the Earth at predefined times. Parameters for the instrument operations and the general satellite configuration constitute the Mission Plans issued by the Mission Planning System which is sent to the Flight Operation Segment (FOS). This, in turn, sends equivalent information to a Ground Station and from there to the satellite antenna of the spacecraft.

A computer on board the satellite will store the list of MCMD (MacroCommands) that ask an instrument or any other part of the satellite to perform an action. These include loading a table, triggering an operation and getting internal status information.

Images from each of the instruments are stored onboard (in the satellite computer memory) as raw data and when the satellite over-flies the Ground station that data is sent to the Ground Station antenna (Data downlink). Conversion from the raw data to “products” is performed in a sequence, at the Ground Station and various Payload Data Segment facilities. These add such things as identification labels and geolocation data to each of the images. Fig. 1 shows graphically the overall scenario. A more detailed explanation of the whole system can be found in [1].

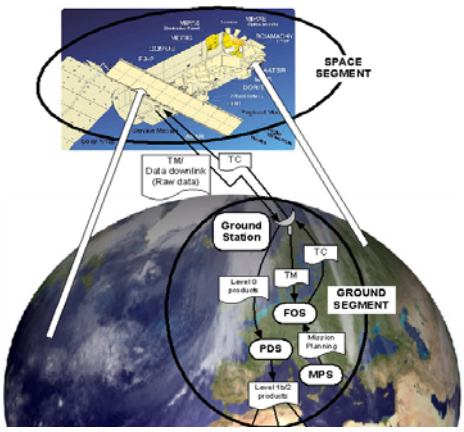


Fig. 1. General overview of an Earth Observation Satellite system (Envisat)

The Envisat satellite was launched in March 2002 with an ambitious, combined payload that ensures the continuity of the data measurements of the European Space Agency (ESA) ERS satellites. ENVISAT data supports Earth Science research and allows monitoring of the evolution of environmental and climatic changes. Furthermore, the data facilitates the development of operational and commercial applications. The list of instruments for this mission is composed of 10 instruments: RA-2, MWR, MIPAS, MERIS, GOMOS, ASAR, AATSR, DORIS, SCIAMACHY and LRR. Reference [2] contains extensive information describing the Envisat mission.

Mission planning for the instruments and for the Satellite operations is issued regularly, nominally on a weekly basis, and can be later modified, before it is frozen for sending to the satellite. If FOS receives requests for some instruments with High Rate modes (e.g. MERIS and ASAR) they have to be accommodated in the previous valid planning. A catastrophic event (earthquakes, volcanic eruptions, hurricanes, …) or a specific demand from the scientific community are examples of events that can cause last minute re-planning.

Existing System for Analysis of Satellite products: QUARC

Data circulates within the system as various Plan and Product Files, with well-defined structures. QUARC is a system that checks off-line the overall data circulation process and in particular the quality of the instrument product files. This process needs as input the product files, the MCMD and the mission planning, provided to the system by other facilities. Apart from the product files in the PDS it needs to retrieve information from other parts of the system to crosscheck that the planning has been transformed properly into MCMD´s, the instrument has performed successfully the measurements (taking images of the Earth), that these images have been stored onboard and transmitted as Raw Data to the Ground station and the processing from Raw Data to Level 0 and then to Level 1B and Level 2 was correct

QUARC returns reports and plots, which help in the production of new plans. Additionally, the QUARC system is designed to assist when taking decisions in the situation where an instrument or the whole system begins to malfunction and to detect, in a semi-automated fashion that something incorrect has occurred in one part of the product generation or data circulation.

The operational QUARC system is located in a single location (ESA-ESRIN, in Italy) that communicates with the archive containing all the products generated from the beginning of the mission and with all the other facilities. The Data Ingestion Modules, one per file type, read the files and convert their contents into parameters that are meaningful to the QUARC data model. The system has been built specifically for this purpose and has bespoke user interfaces. It took several years to build and there are significant ongoing maintenance and development costs as new reports are required and new missions are launched.

Use Cases

In addition to functional and non-functional requirements from the existing system we produced Use Cases to support incremental, distributed development. These translated directly into Test Cases for evaluation of the system.

Use Case 1: Instrument unavailability This is a Use Case to ensure our new system is capable of replicating the core functionalities of the existing system. A user needs to find out what planned events and generated products exist for a given time period and instrument, and to plot these results against each other in a timeline. A simple interface is needed, with no underlying complexity exposed to the user.

Use Case 2: Check for the quality of the products in Nominal mode Certain sorts of products have internal parameters giving a measure of quality of data. The specific situation for this use case at present would be extraction of one of these quality parameters, over a period of time, for an instrument in a particular mode, being "Nominal". The product files we were to work with didn't include this quality data so we expanded this to the more general requirement to be able to extract any piece of metadata from a set of product files.

Use Case 3: Update of functionalities with no software update A crucial perceived advantage of the semantic approach was the flexibility with which the system could be adapted. A mission may last 10/15 years and since we are largely investigating anomalous behaviour not all useful queries will be known ahead of time. We needed to know how easily we could develop new queries over our data, and parameterise them for use by ordinary users.

Use Case 4: Data lifecycle The satellite plans are not static and the system needed to be able to remove or update metadata from its stores. This needed to be done automatically, and only in the correct circumstances of a new plan covering the same time period and from the provider of the original plan. When querying, the user must be given information about the final, executed, plan.

Use Case 5: Modularity of metadata service The desire to be able to change the metadata store comes from wanting flexibility in extending the system. The approach was to design and build a loosely-coupled, service-orientated architecture. In particular we would ensure we could change the metadata store and query engine, but more generally we use modular components defined by their interfaces. Choices between components can be made on various characteristics including cost, scalability, reliability, and performance. Crucially the user shouldn't have to worry about implementation details.

3 Technical Issues and Solutions

The following diagram shows the geographical deployment and component breakdown of the developed system. Software was deployed at 3 sites – Manchester, Madrid and Athens, and Atlas further uses the Everlab cluster of machines throughout Europe. The number actions 1-5 and 6-8 show the activity flow for annotating and querying data respectively.

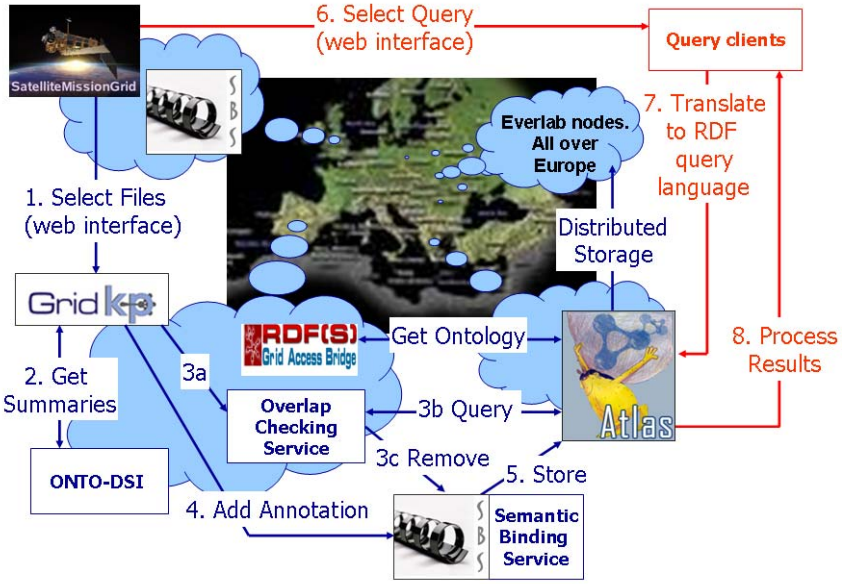


Fig. 2. System architecture

Infrastructure: a distributed, scalable architecture

S-OGSA [3] is the reference Semantic Grid architecture developed in Ontogrid and used for the Satellite Mission Grid. The basis of S-OGSA is an information model of semantic resources, which extends the OGSA model [4]. S-OGSA also anticipates that a set of capabilities will be required from Grid middleware in order to address the new services with varying degrees of semantic capabilities. For this, S-OGSA includes two service categories called Semantic Provisioning Services and Semantically Aware Grid Services. Below are details of the Information Model and the Service Categories, and their use in the Satellite Mission Grid.

S-OGSA Information Model The S-OGSA model identifies three types of entities:
Grid Entities - anything that carries an identity on the Grid, including resources and services [5]. In this system they include planning systems, planning files, satellite in-

struments, product files and product processing facilities. These entities would be considered similarly in any non-semantic implementation of the process.
Knowledge Entities - special types of Grid Entities that represent or could operate with some form of knowledge. Examples of Knowledge Entities are ontologies, rules, knowledge bases or even free text descriptions that encapsulate knowledge that can be shared. In this system we had a single ontology including classes for times, planning systems, macrocommands, satellite instruments, and the details of the various plan and product file metadata. Ultimately there needs to be a set of ontologies to cover the whole satellite mission domain. In the Satellite Mission Grid an annotation process creates knowledge entities (sets of RDF statements) for the different types of files.
Semantic Bindings - Knowledge Entities that represent the association of a Grid Entity with one or more Knowledge Entities (that is, they represent semantic metadata of a Grid Entity). Existence of such an association transforms the subject Grid entity into a Semantic Grid Entity. Semantic Bindings are first class citizens as they are modeled as Grid resources with an identity and manageability features as well as their own metadata. Grid Entities can acquire and discard associations with knowledge entities through their lifetime. In our system the files are made into Semantic Grid Entities by attaching the created annotations.

Semantic Provisioning Services These are Grid Services that provision semantic entities. Two major classes of services are identified:
Knowledge provisioning services - these can produce (and in some cases store and manage) Knowledge Entities. Examples of these services are ontology services and reasoning services. In this system the ontology services are implemented using RDF(S) Grid Access Bridge [6], an implementation of WS-DAIOnt [7].
Semantic Binding provisioning services - these can produce (and in some cases store and manage) Semantic Bindings. The system includes an annotation service that generates Semantic Bindings from planning and product files. This annotation service is implemented using the Grid-KP [8][9] tool. It also uses a storage service for the Semantic Bindings that are generated, so that they can be accessed at any time during their lifetime using a query language. This service [10] was implemented twice, once using the Atlas system [11][12] and once in Sesame [13]. These two services were “swappable” components.

Semantically Aware Grid Services This special type of Grid Services are able to exploit semantic technologies to consume Semantic Bindings in order to deliver their functionality [14]. Their role is complementary to the role of Semantic Provisioning Services since they consume the semantic entities held by Knowledge provisioning services and Semantic Binding provisioning services, and use their services. The user interface for the Satellite Mission Grid is a Semantically Aware Grid Service, making use of all the aforementioned elements in order to deliver its enhanced functionality.

Annotation: making metadata explicit

Data circulates in the existing systems as files with many common generic features. They are slightly different for planning and product files, and the information about these planned events and generated products is usually bound up with the data in-

volved. Standard ASCII formats encode the information in keyword-value pairs, which are stored as headers for the various files. This is a special format defined for the Envisat mission with an enormous amount of software and documentation generated through years of development. This structure can be simply translated to a fairly flat XML structure. Once this is performed on the planning and product files, the system uses XML software tools.

Product files consist of an ASCII header in the above format and a binary part encoded in an ESA proprietary format. This header is just a few Kbs out of an image file size of Gbs. The Onto-DSI [15] component was used to extract and provide just the headers from these files to avoid a large system overhead whilst annotating them.

Much of the metadata was encoded in specific, amalgamated identifiers. For example simple rules had to be created to process product filenames such as "RA2_MW__1PNPDK20060201_120535_000000062044_00424_20518_0349.N1". This is decomposed into an Event type (RA2_MW), Processing level (1P) and centre (PDK), a Sensing start time (2006-02-01:12.05.33) and so on. Generic metadata (applied across all captured metadata) and the ontology further add, for example, that the Event type (RA2_MW) is executed by a particular instrument, the Radar Altimeter. The ontology simplifies this procedure by making it a simple matter of a domain expert stating where the pieces of information are found. A parser extension was written in Grid-KP to carry out the extraction of the relevant properties from the files and this task was separated from other work such as the creation of query interfaces. While it was not used formally or automatically in this implementation, it is easy to force verification of the metadata against the ontology and this was done manually on occasion.

Another issue was conversion of units. One example of this was converting from date formats, as given above (and given to the users in the webforms) to another standard time format used in space missions, MJD2000. It is the number of seconds (and milliseconds) to have passed since the year 2000, including leap seconds. The conversion routine was wrapped as a webservice using SOAPLAB [16].

It is anticipated that migration of other data to the system would be much simplified by this process and these tools being in place. In addition the annotation services were deployed in different locations, which supported the distributed nature of the data sources.

Storage: managing a (meta)data lifecycle

The Annotation Service was able to use exactly the same mechanisms as the user interface to communicate with the Semantic Binding Service to ask if its current file overlapped with any existing Plan files. This design has two advantages; firstly, no new specific code needed to be written as we already had the query interfaces. Secondly, although the logic needed here was quite simple, we have allowed ourselves full access to the flexibility of RDF querying, which means that if more complex rules are needed in future we will be able to accurately encode them. Having established if an update was needed the RDF was updated (deleted and replaced), or not, using the standard mechanisms provided by the metadata stores.

Managing our RDF in identifiable, separate Semantic Bindings allows us to better manage the overlaps, and the lifetime of the metadata when several annotations may

be created. This also gives us confidence in the ability to incrementally migrate data into such a system.

Scalability was not explicitly investigated as part of this use case, but tests have been carried out separately on each of the separate components making up the system. This is the most sensible approach in this type of architecture, where components are combined without a need for large amounts of integration work.

Querying: exploring the data

Having created semantic bindings from our files to RDF statements about them, we had gained in flexibility in our querying. We used SPARQL and SeRQL when Sesame was the metadata store and RQL when Atlas was the store. The ability to choose which language we wanted was another reason for considering interchangeable metadata stores. We worked with a flexible "Free Querying" interface as we considered how the system would be incrementally improved and developed. This interface simply allowed the user to create queries (in the language of their choice) and get the results back in a tabular form.

As an example we looked at how we could abstract our queries over the data to a level where new sorts of data could be added to the system and still be identified by our queries. An initial implementation of one of the queries for Use Case 1 was looking for all planned events (DMOP event records) which were using the Radar Altimeter. We matched at the low level of Event identifiers using the implicit metadata that "events with identifiers containing RA are carried out by the Radar Altimeter instrument". The nature of RDF as a web of statements and the existence of an ontology to formalise the existence of different properties made it easy to move these queries to an improved, semantic level.

We were initially searching on the event_id property of the DMOP_er class (DMOP event records), which look like `"RA2_IE_00000000002372"`. It matches `REGEX(?EVENT_ID,".*RA.*")` in the SPARQL regular expression syntax. This query works, but we were able to see in the ontology that a better level was possible.

The individual data items about planned events use the event ids, but our system was able to augment that with the knowledge about which event types use which instruments. This was enabled by having an ontology which included instruments and types of events as objects independent of the individual events which they classify. The following diagram showing part of the Satellite Ontology shows that the `DMOP_er` class (top left) is related to the `Plan_Event` class by `represents_plan_event` property, and that `Plan_Event` instances have their own identifiers – `plan_event_id`. These look like `"RA2_CAL"` or `"RA2_IE"` and we could match them explicitly. They represent the different types of events that can be planned. We then looked one level further - moving up to the level of the `Instrument` class and see the Radar Altimeter is identified as one such, with `instrument_id` of `"RA"`.

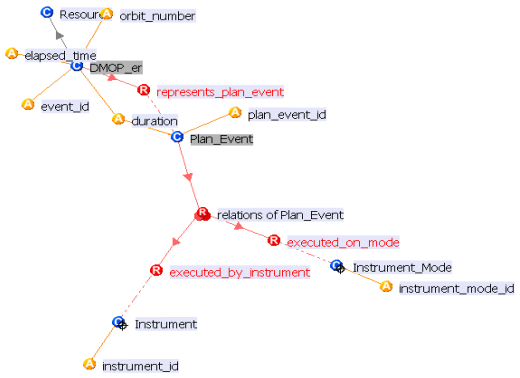


Fig. 3. A section of the Satellite Ontology showing the specific events (DMOP-er), their types (Plan_Event) and the Instruments which carry out those event types. Diagram from NeOn [17]

We moved in our SPARQL query from

```
?EVENT event_id ?EVENT_ID ;
FILTER ( REGEX(?EVENT_ID,".*RA.*"))
```

to

```
 ?EVENT event_id ?EVENT_ID ;
        represents_plan_event ?PLAN_EVENT_TYPE .
?PLAN_EVENT_TYPE executed_by_instrument ?INSTRUMENT .
     ?INSTRUMENT instrument_id "RA"
```

While this is longer it is both clearer to understand and to implement as a webform where the user will select an instrument. It is also likely to execute more quickly as it is looking for an exact match of `instrument_id` rather than having to rely on regular expression parsing of a string value.

The biggest gain is that it is much more robust in the face of changing data. We can continue to use these "semantic level" queries about instruments even if we add new event types which use this instrument or change the unique identifiers for individual DMOP event records. If further data in the system contained new sorts of events planned and carried out by the Radar Altimeter then our queries would automatically match them. In any of these extended cases a simple statement associates the new event type with an existing instrument or new events with an existing event type. The exact same query (for use of an instrument) will then also report about these new events. We shifted from talking about details of identifiers to the actual objects which the user is concerned about. i.e. we moved to a more semantic level. This process is shown in more detail in an OntoGrid demonstration video [18].

Accessibility: providing tools for end users

Having developed the queries in a language such as SPARQL we very easily built webforms to allow users to provide the values for any such queries. A web application (running on Tomcat) was developed to serve Java Server Pages which presented interfaces and results to users, and converted to RDF queries, and back from RDF results sets. The results were provided in a simple XML structure and we generated from that either tables or graphs.

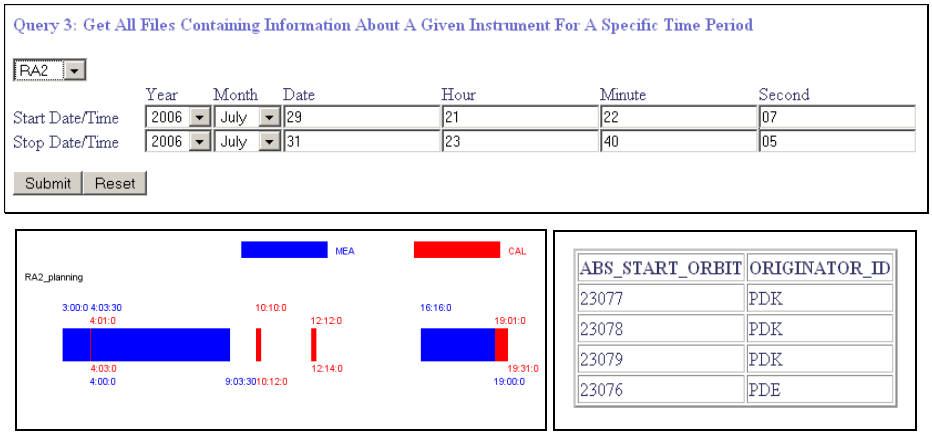


Fig. 4. Webform for user input and timeline and tabular outputs

4 Evaluation

The implementation shows several advantages with respect to a conventional approach, in terms of flexibility, reduction of software running costs, maintainability, expandability, and interoperability. A testing phase allowed us to evaluate in more detail the various aspects identified in the Use Cases.

Access to Data and Information

Use of Envisat data products or product related information is granted for the product metadata (in product headers) but not generally the science data itself, unless approved under the CAT-1 regulation [19]. This imposes a condition on the system, that it may sometimes be able to retrieve information for some parts within the files of the Envisat system, but this access has to be explicitly forbidden for the general users. These access concerns were taken care of in the system by making sure annotation components only processed the correct, public parts of the files.

The Grid infrastructure made possible the management of data in several locations.

Legacy formats/systems

One of the key issues to bear in mind when implementing a completely new design in a system of this size and complexity is how to manage migration of systems. Some parts of the system (probably belonging to other organisations or facilities) are not changed at all, or only partially updated and there is no simultaneous modernisation of all parts in the system. It is therefore necessary to gradually adapt the un-changed elements to the new development and to incrementally migrate functionality. Legacy systems can be considered in terms of data formats and software elements.

Data formats Although current data files are structured and in well documented formats there remain hundreds of different file types. Much of the metadata is only "implicit", such as the information stored within filenames or specialised code systems. For these files we have made this metadata "explicit", and much more easily accessible. This helps query developers, and also systems that must manage the lifecycle of metadata.

The use of the ontology to store information about properties will make migration of future data simpler. The mapping simply has to be done between the data and the ontology once. This should be especially easy as these existing systems have very strictly defined inputs and outputs and increasingly the formats used are XML based. The process of writing the specific code to extract the information from the datafile and re-encode it as RDF is much simplified, and can be more easily managed.

Software In the Envisat mission, complex software elements with well-defined interfaces are used in both planning and product generation. Some of these functionalities were incorporated in the prototype (e.g. time conversion utilities) by enabling them to be accessed as Web Services.

In a distributed architecture, such as that used in this Semantic Grid, encapsulation allows external systems to interact with individual parts of the system. For example, during a transitional period they might simply make use of a query interface to extract information or an RDF interface to provide some data to an RDF store or annotation component.

Standardised approach

The metadata format and query language defined purposely for the current QUARC implementation, although powerful, cannot be exported and directly used in other systems.

The Satellite Mission Grid uses the RDF standard for the storage of the metadata, and the specifics of the format are described by OWL and RDF(S) schemas. RDF allows us to use standard query languages like SPARQL or RQL which are incorporated in tools which have been already tested and proved adequate for re-use in other systems. The use of standard formats also allows different metadata stores to be used, depending on circumstances. For example, we used Atlas and Sesame. Another advantage is in not having to train developers in specific new skills but to be able to use what they already know. However, the existence of several query languages can add an overhead in terms of the required skills for developers.

Flexibility

The system allows a user who is familiar with one of the supported query languages to develop queries directly and iteratively. The process of creating forms which allow other users to provide the specific values for these queries is simple, and made even simpler by the existence of 'common' methods for converting between time formats. In this way we have been able to demonstrate how the system can be enhanced without any significant technical changes being required. This is crucial in an analysis system, where not all relevant queries are known or defined at the beginning of the use of the system.

It is also possible to add in some new relationships or properties without having to change the existing data at all. If a new way of grouping some part of the existing information was required by operators of the system then it could be added into the RDF directly. For example, a set of subtypes of instrument types could be created to describe their power consumption. Then each of the instruments could be labelled as belonging to one or other of the subtypes. This would take just a few statements in RDF and the millions of pieces of event information organised by which instrument generates them would now also be connected to this set of subtypes, allowing a user to make queries based on the power consumption.

Semantic Technologies

An Earth observation mission may last 10/15 years and the completely flexibly query interface allows exploring of data and development of new queries. This is crucial as anomalies are discovered and examined. Furthermore, new missions may provide information which is useful to combine with this existing data.

The developed Satellite Ontology can be extended and enlarged as required by the complexity of functionalities involved in the system. The structure developed so far has been sufficient for the data managed so far. The common ontology has enabled communication, and rapid extension of functionalities as demonstrated through Use Cases 2 and 3. The addition of "generic" information (such as the linking of event types to instruments) allows us to have semantically rich annotations. This semantic approach to queries where they are moved up to greater levels of abstraction gives us much more flexibility and robustness over time, as we are querying what the users need to know (usage of the instrument) rather than what we traditionally have recorded (the list of codes used for the event types). This shows the general technique of modelling what it is that users wish to search and discuss. As well as making development more efficient it reduces the time to acquaint new users with the system.

Data life cycle

We have shown that controlled updates can be made to stored data. These are automated but only allowed from authorised data sources. This ability supports the fact that data is not static, and that having access to the correct information can involve removing out-of-date statements as well as adding new ones.

More generally, we hope to be able to integrate data from many sites and missions and reuse the same system across these missions. As such we have created the methodology and tools for adding new data sources. Lightweight annotation components can convert from a legacy system to common RDF which is made available (via the semantic binding service) to the query engines.

There are high volumes of data being produced by Envisat – anticipated to reach a petabyte of data in 10 years of operation [20]. The extraction and management of just metadata (rather than all the data itself) is necessary for any ongoing analytical system. In a distributed system such as the Satellite Mission Grid, we create a single amalgamated dataset from many geographically dispersed data silos. Crucially, all the access is virtualised from the user perspective, i.e. they don't have to know about it at all. The resources and computation are distributed but the user has a simple, local browser interface for annotation and querying.

Modularity and Extensibility

The abstraction of components in a loosely coupled system means we have all the advantages of modularity. Interchangeable components can be selected depending on the particular application of the system. It allows the users to enjoy a single interface into whichever systems are determined to be best suited to the current scale of data and complexity of query. We also gain in extensibility; it opens up any further development to using "best-of-breed" components, be they new versions of existing ones, or completely new development. Separate testing of the various components will provide better metrics for deciding between them. We encountered the usual problems of distributed deployment but none proved insoluble and with experience these will be further reduced.

5 Conclusions and Next Steps

A semantic approach, where metadata is created explicitly and managed as a "first class object" gives advantages of flexibility and extensibility. A Grid approach where data and computations are distributed across many sites and machines gives improvements of scalability and robustness. The prototype system has shown itself capable of carrying out the current functionality of mission analysis systems, but across a geographically distributed dataset. It has also shown itself to be easy to extend in capability without significant development effort.

In the Semantic Data Grid community we have helped focus on lightweight protocols and making components more easy to integrate with existing systems. This vision supports a movement towards SOKU – Service Oriented Knowledge Utilities. The next industry steps include the incremental updating and extension of existing systems, where we will add to the metadata we store and make explicit what was formerly implicit. There can also be new approaches to work on both internal tools and for external development work within ESA projects. The experience means we can actively seek out which new projects to build from the ground up in a semantically aware manner.

6 Acknowledgments

The authors would like to thank the other members of the OntoGrid consortium. They are also very grateful to the European Space Agency (ESA) representatives Olivier Colin (ESA-ESRIN) and Pierre Viau (ESA-ESTEC) for providing access to the actual products and auxiliary tools from the Envisat mission.

7 References

1. Sánchez Gestido, M.: OntoGrid Business Case and User Requirements Analysis and Test Set Definition For Quality Analysis of Satellite Missions. Deliverable D8.1, OntoGrid, http://www.ontogrid.net (2005)
2. ESA bulletin number 106, "EnviSat special issue", http://www.esa.int/esapub/pi/bulletinPI.htm
3. Corcho, O., Alper, P., Kotsiopoulos, I., Missier, P., Bechhofer, S., Kuo, D., Goble, C.: An overview of s-ogsa: a reference semantic grid architecture. Journal of Web Semantics 4 (2006)
4. Foster, I., Kishimoto, H., Savva, A., Berry, D., Djaoui, A., Grimshaw, A., Horn, B., Maciel, F., Siebenlist, F., Subramaniam, R., Treadwell, J., Reich, J. V.: The open grid services architecture, version 1.0. Technical report, Open Grid Services Architecture WG, Global Grid Forum (2005)
5. Treadwell, J.: Open grid services architecture glossary of terms. Technical report, Open Grid Services Architecture WG, Global Grid Forum (2005)
6. RDF(S) Grid Access Bridge, OntoGrid, http://www.ontogrid.net
7. Estebán-Gutierrez, M., Gómez-Pérez, A., Corcho, O., Muñoz-García, O.: WS-DAIOnt-RDF(s): Ontology Access Provision in Grids, The 8th IEEE/ACM International Conference on Grid Computing, http://www.grid2007.org/?m_b_c=papers#2b (2007)
8. Grid-KP, OntoGrid, http://www.ontogrid.net
9. http://www.isoco.com/innovacion_aplicaciones_kp.htm
10. Corcho, O., Alper, P., Missier, P., Bechhofer, S., Goble, C.: Grid Metadata Management: Requirements and Architecture, The 8th IEEE/ACM International Conference on Grid Computing, http://www.grid2007.org/?m_b_c=papers#2b (2007)
11. Atlas, OntoGrid, http://www.ontogrid.net
12. Miliaraki, I., Koubarakis, M., Kaoudi, Z.: Semantic Grid Service Discovery using DHTs, 1st CoreGrid WP2 Workshop on Knowledge and Data Management (2005)
13. Sesame, Aduna Open Source project, http://www.openrdf.org
14. Alper, P., Corcho, O., Goble, C.: Understanding Semantic-Aware Grid Middleware for e-Science., Journal of Computing and Informatics (To be published)
15. ONTO-DSI, OntoGrid, http://www.ontogrid.net
16. Senger, M., Rice, P., Oinn, T.: Soaplab - a unified Sesame door to analysis tools. In: Proceedings of UK e-Science, All Hands Meeting, 2-4th September, Nottingham, UK (2003)
17. NeOn toolkit, http://www.neon-project.org/web-content/
18. Wright, R: Ontogrid Satellite Use Case demonstration video, http://www.youtube.com/watch?v=TSbb_8vmKvk
19. ESA Earth Observation Principal Investigator, http://eopi.esa.int/esa/esa
20. European Space Agency Information Note 13, http://www.esa.int/esaCP/ESA0MDZ84UC_Protecting_0.html

Reusing Human Resources Management Standards for Employment Services

Asunción Gómez-Pérez[1], Jaime Ramírez[1] and Boris Villazón-Terrazas[1]

[1] Facultad de Informática, Universidad Politécnica de Madrid, Campus Montegancedo s/n
28860, Boadilla del Monte, Madrid, Spain
{asun, jramirez, bvillazon}@fi.upm.es

Abstract. Employment Services (ESs) are becoming more and more important for Public Administrations where their social implications on sustainability, workforce mobility and equal opportunities play a fundamental strategic importance for any central or local Government. The EU SEEMP project aims at improving facilitate workers mobility in Europe. Ontologies are used to model descriptions of job offers and curricula; and for facilitating the process of exchanging job offer data and CV data between ES. In this paper we present the methodological approach we followed for reusing existing human resources management standards in the SEEMP project, in order to build a common "language" called Reference Ontology.

Keywords: Human Resources Management Standard, Human Resources Ontologies.

1 Introduction

Nowadays there is an important amount of investment in human capital for economic development. Human resources management refers to the effective use of human resources in order to enhance organisational performance [13]. The human resources management function consists in tracking innumerable data points of each employee, from personal records (data, skills, capabilities) and experiences to payroll records [13]. Human resources management has discovered the Web as an effective communication channel. Although most businesses rely on recruiting channels such as newspaper advertisements, online job exchange services, trade fairs, co-worker recommendations and human resources advisors, online personnel marketing is increasingly used with cost cutting results and efficacy.

Employment Services (ESs) are becoming more and more important for Public Administrations where their social implications on sustainability, workforce mobility and equal opportunities play a fundamental, strategic importance for any central or local Government. The goal of the SEEMP[1] (Single European Employment Market-Place) project is to design and implement an interoperability architecture for public e-Employment services which encompasses cross-governmental business and decisional

[1] http://www.seemp.org/

processes, interoperability and reconciliation of local professional profiles and taxonomies, semantically enabled web services for distributed knowledge access and sharing. The SEEMP project relies on WSMO [4] that permits to semantically describe Web Services, ontologies and mediators. WSML [3] is the concrete language used in SEEMP for encoding those descriptions. For this purpose, the resultant architecture will consist of: a Reference Ontology, the core component of the system, that acts as a common "language" in the form of a set of controlled vocabularies to describe the details of a job posting or a CV (Curriculum Vitae); a set of local ontologies, so that each ES uses its own local ontology, which describes the employment market in its own terms; a set of mappings between each local ontology and the Reference Ontology; and a set of mappings between the ES schema sources and the local ontologies [5].

A major bottleneck towards e-Employment applications of Semantic Web technology and machine reasoning is the lack of industry-strength ontologies that go beyond academic prototypes. The design of such ontologies from scratch in a textbook-style ontology engineering process is in many cases unattractive for two reasons. First, it would require significant effort. Second, because the resulting ontologies could not build on top of existing community commitment. Since there are several human resources management standards, our goal is not to design human resources ontologies from scratch, but to reuse the most appropriate ones for public e-Employment services developed on the framework of the SEEMP project. In this paper we present the methodological approach we followed for reusing existing human resources management standards like NACE[2] (Statistical Classification of Economic Activities in the European Community), ISCO-88 (COM)[2] (International Standard Classification of Occupations, for European Union purposes) and FOET[2] (Classification of fields of education and training), among others.

This paper is organized as follows: Section 2 presents some related work. Next section 3 explains the adopted methodological approach to build the SEEMP Reference Ontology from standards/classifications and already existing ontologies, and then in section 4 an overall perspective of the resultant SEEMP Reference Ontology is shown. Then section 5 describes some considerations with respect to the building process of the local ontologies taking as starting point the Reference Ontology and the ES data sources. Finally, section 6 offers some final conclusions, and poses the future work that, among other things, will serve to validate the ideas proposed in this paper.

2 Related Work

Currently the Human Resource Semantic Web applications are still in an experimental phase, but their potential impact over social, economical and political issues is extremely significant.

COKE is described in [9], a three-level ontology containing a top-level Human Resources ontology, a middle-level Business Process ontology and a lower-level Knowledge Objects ontology. PROTON (PROTO-Ontology), a 4-level ontology

[2] Available through RAMON Eurostat's Classifications Server at http://ec.europa.eu/comm/eurostat/ramon/

which specializes in coverage of concrete and/or named entities (i.e. people, organizations, numbers) and is used for HR applications [10]. Bizer et al present in [1] a scenario for supporting recruitment process with Semantic Web technologies but just within German Government. Mochol et al depict in [15] a brief overview of a Semantic Web application scenario in the Human Resources sector by way of describing the process of ontology development, but its final goal is to merge ontologies. In [2] it is described a competency model and a process dedicated to the management of the competencies underlying a resource related to e-recruitment (mainly CV or a Job Offer).

Regarding main standardization initiatives in the HR sector, the HR-XML consortium has built up a library of more than 75 interdependent XML schemas which define the data elements for particular HR transactions, as well as options and constraints governing the use of those elements [11].

Finally there is an effort described in [12] which mission is to promote technology into HR/e- learning standards and applications. Its current focus topics includes: semantic interoperability, semantic of HR-XML[11], etc.

3 Methodological approach for Reusing Human Resources Management Standards

In this section we describe the adopted approach to build the SEEMP Reference Ontology; a preliminary version is described in [7]. This methodological approach follows and extends some of the identified tasks of the ontology development methodology METHONTOLOGY [6]; this methodological approach consists of:

1. Ontology specification; in this activity we specify, using competency questions, the necessities that the ontology has to satisfy in the new application.
2. Standards selection; in this activity we select the standards and existing ontologies that cover most of the identified necessities.
3. Semantic enrichment of the chosen standard; this activity states how we enrich semantically the chosen standard.
4. Ontology evaluation; in this activity we evaluate the ontology content.

3.1 Ontology specification.

This activity states why the ontology is being built, what its intended uses are, and who the end-users are. For specifying the ontology requirements we used the competency questions techniques proposed in [8].

- *Intended uses of the ontology*. The purpose of building the Reference Ontology is to provide a consensual knowledge model of the employment domain that could be used by ESs, more specifically within the ICT (Information and Communication Technology) domain.
- *Intended users of the ontology*. We have identified the following intended users of the ontology: candidates, employers, public or private employment search

service, national and local governments; and European commission and the governments of EU countries.

- *Competency Questions*. These questions and their answers are both used to extract the main concepts and their properties, relations and formal axioms of the ontology. We have identified sixty competency questions; they are described in detail in [14]. An example of the competency questions is: *Given the personal information (name, nationality, birth date, contact information) and the objectives (desired contract type, desired job, desired working conditions, desired salary) of the job seeker, what job offers are the most appropriate?.*
- *Terminology*. From the competency questions, we extracted the terminology that will be formally represented in the ontology by means of concepts, attributes and relations. We have identified the terms (also known as predicates) and the objects in the universe of discourse (instances); they are described in detail in [14].

3.2 Standards selection.

In order to choose the most suitable human resources management standards for modeling CVs and job offers, the following aspects have been considered: *The degree of coverage of the objects identified in the previous task*, this aspect has been evaluated taking into account the scope and size of the standard. However, a too wide coverage may move us further away from the European context; therefore we have tried to find a tradeoff between this aspect and the following one: *the current european needs*, it is important that standard focuses on the current European reality, because the user partners involved in SEEMP are European, and the out coming prototype will be validated in European scenarios; and the *user partners recommendations,* in order to asses the quality of the standards, the opinion of the user partners is crucial since they have a deep knowledge of the employment market.

Besides, when choosing the standards, we also took into account that the user partners of SEEMP selected the ICT domain for the prototype to be developed in SEEMP. Hence, the chosen standards should cover the ICT domain with an acceptable degree. In the case of the occupation taxonomy, as it will be shown, we have chosen one standard, but then we have taken some concepts coming from other classifications, in order to obtain a richer classification for the ICT domain.

When specifying job offers and CVs, it is also necessary to refer to general purpose international codes such as country codes, currency codes, etc. For this aim, the chosen codes have been the ISO codes, enriched in some cases with user partners' classification.

Finally, the representation of job offers and CVs also require temporal concepts such as interval or instant. So, in order to represent these concepts in the final Reference Ontology, the DAML time ontology[3] was chosen.

[3] http://cs.yale.edu/homes/dvm/daml/time-page.html

3.3 Semantic enrichment of the chosen standard.

This activity states how we enrich the human resources management standards. In order to make possible the enrichment of the standards, it was necessary to import them into the ontology engineering tool WebODE [6]. This process consists of implementing the necessary conversions mechanisms for transforming the standards into WebODE's knowledge model. For this purpose we have developed for each standard/classification an *ad hoc* translator (wrapper) that transformed all the data stored in external resources into WebODE's knowledge model.

Once we transformed the standards into ontologies, the next step is to enrich them introducing concept attributes and *ad hoc* relationships between ontology concepts of the same or different taxonomies. We perform this task, doing the following. We created from scratch the Job Seeker Ontology (models the job seeker and his/her CV information), and the Job Offer Ontology (models the job vacancy, job offer and employer information); following some HR-XML[11] recommendations. Moreover, we defined relationships between the concepts of the Job Seeker and Job Offer Ontologies and the concepts defined on the standard (classification) based ontology.

3.4 Ontology Evaluation.

The evaluation activity makes a technical judgment of the ontology, of its associated software environments, and of the documentation. We will evaluate the Reference Ontology using the competency questions identified in the first task.

4 SEEMP Reference Ontology

The Reference Ontology[4], described in this section, will act as a common "language" in the form of a set of controlled vocabularies to describe the details of a job posting and the CV of a job seeker. The Reference Ontology was developed following the process described in detail in section 3 and with the ontology engineering tool WebODE [6]. The Reference Ontology is composed of thirteen modular ontologies: *Competence, Compensation, Driving License, Economic Activity, Education, Geography, Job Offer, Job Seeker, Labour Regulatory, Language, Occupation, Skill* and *Time*. The main subontologies are the Job Offer and Job Seeker, which are intended to represent the structure of a job posting and a CV respectively. While these two subontologies were built taking as a starting point some HR-XML [11] recommendations, the other subontologies were derived from the available international standards (like NACE, ISCO-88 (COM), FOET, etc.) and ES classifications and international codes (like ISO 3166, ISO 6392, etc.) that best fit the European requirements. Figure 1 presents:

- These thirteen modular ontologies (each ontology is represented by a triangle). Ten of them were obtained after wrapping the original format of the

[4] The Reference Ontology is available at: http://droz.dia.fi.upm.es/seemp/ (Username: seemp and password: employer)

standard/classification, using *ad hoc* translator or wrapper for each standard/classification.

- The connections between the ontologies by means of *ad hoc* relationships.

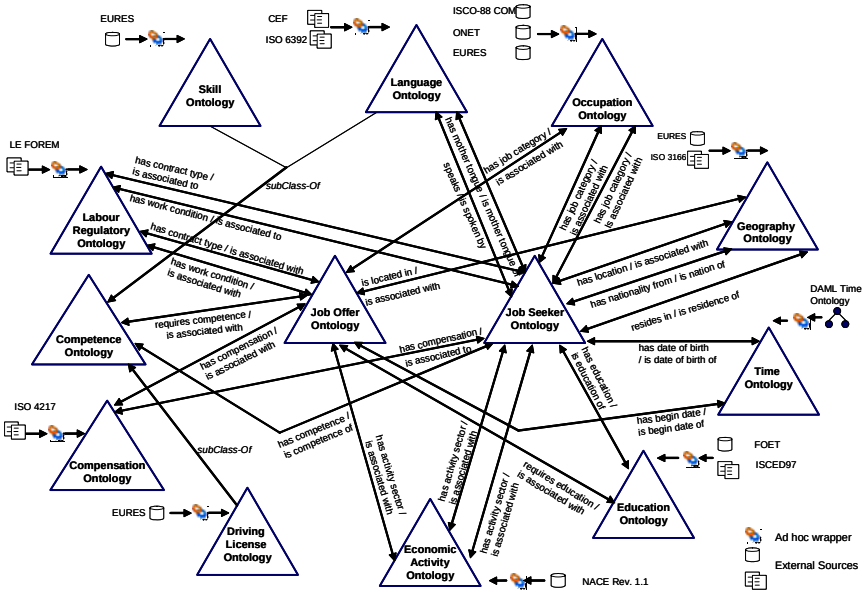


Fig. 1. Main ad-hoc relationships between the modular ontologies.

Next we provide the conceptualization of the mentioned ontologies.

- *Job Seeker Ontology*. This ontology models knowledge of job seeker and CV information used in the SEEMP Project. The Job Seeker Ontology imports concepts from the education ontology, language ontology, economic activity ontology, compensation ontology, geography ontology, driving license ontology, labour regulatory ontology and skill ontology, and these imported concepts are used to connect the Job Seeker Ontology with the other ontologies. **Error! Reference source not found.** shows all the *ad hoc* relationships whose domain is a concept belonging to the Job Seeker Ontology (concepts of the Job Seeker Ontology are drawn in yellow). Examples of the relationships can be: '`Job Seeker` *has driving license* `Driving License`' (with the `Driving License` concept from the Driving License Ontology), '`Job Seeker` *has education* `Education`' (with the `Education` concept from the Education Ontology), '`Job Seeker` *has mother tongue* `Language`' (with the `Language` concept from the Language Ontology), etc.
- *Job Offer Ontology*. This ontology models knowledge of job vacancy, employer and job offer information used in the SEEMP Project. The Job Offer Ontology imports concepts from the education ontology, language ontology, economic activity ontology, compensation ontology, geography ontology, driving license ontology, labour regulatory ontology and skill ontology, and these imported concepts are used to connect the Job Offer Ontology with the other ontologies.

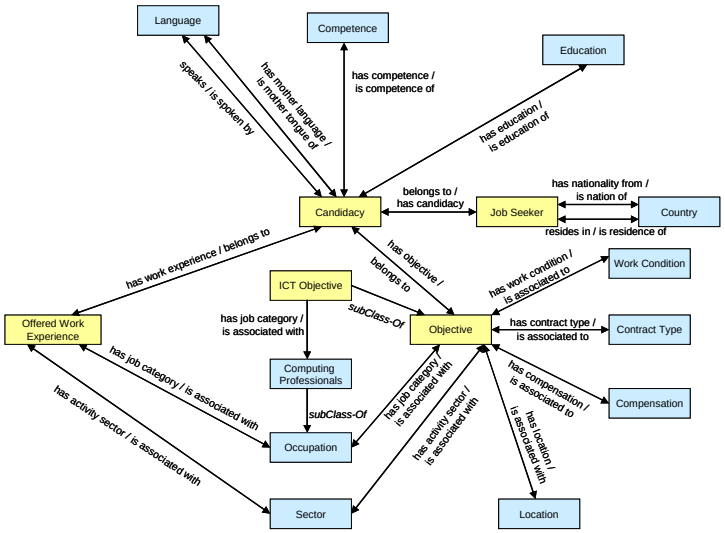


Fig. 2. The ad-hoc relationships of the Job Seeker Ontology

Figure 3 shows all the *ad hoc* relationships whose domain is a concept belonging to the Job Offer Ontology (concepts of the Job Seeker Ontology are drawn in green). Examples of the relationships can be: '`Job Vacancy` *requires driving license* `Driving License`' (with the `Driving License` concept from the Driving License Ontology), etc.

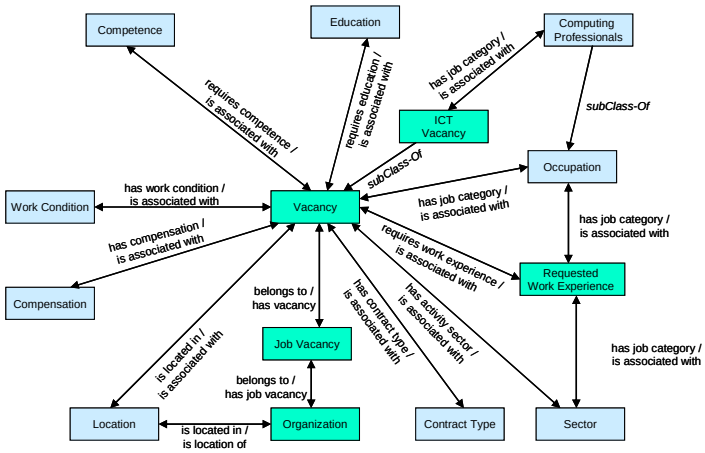


Fig. 3. The ad-hoc relationships of the Job Offer Ontology

- *Compensation Ontology.* This ontology models knowledge of wages and salaries used in the SEEMP Project. It is based on the ISO 4217[5]. The ISO 4217 is expressed in HTML format. It is a list of 254 currency names and codes. The

[5] http://www.iso.org/iso/en/prods-services/popstds/currencycodeslist.html

resultant Compensation Ontology has 2 concepts: `Currency` and `Salary`. For every currency element specified in the ISO 4217 a different instance of the `Currency` concept is defined. So, the `Currency` concept has 254 instances. An example instance of the `Currency` concept is `UNITED STATES - US Dollar`. The *ad hoc* relations defined in this ontology are *has currency* between the `Salary` and `Currency` concepts, and its inverse one.

- *Driving License Ontology.* This ontology models knowledge of Driving License domain. It is based on the levels recognized by the European Legislation[6]. This classification is expressed in HTML format and it is a list of 12 kinds of driving licenses. The resultant Driving License Ontology just has the `Driving License` concept; and for every kind of driving license specified in the European Legislation a different instance of the `Driving License` concept is defined. An example instance of the `Driving License` concept is `A1 - Light weight motorcycle.` The `Driving License` concept has two relations: '`Driving License` *is associated with* `Job Vacancy`' (with the `Job Vacancy` concept from the Job Offer Ontology) and '`Driving License` *is driving license of* `Job Seeker`' (with the `Job Seeker` concept from the Job Seeker Ontology).
- *Economic Activity Ontology.* This ontology models knowledge of economic activities and sectors. It is based on the NACE Rev. 1.1[7]. This standard is expressed in MS Access database format and it is a classification of 849 economic activities. The resultant Economic Activity Ontology has 849 concepts. In this case we have defined a concept for every element of the NACE taxonomy in order to preserve the hierarchy. In the Economic Activity Ontology, the most general concept is the `Activity` concept. This concept is organized in the taxonomy based on the NACE. The `Activity` concept has four relations: '`Activity` *is associated with* `Objective`' (with the `Objective` concept from Job Seeker Ontology), '`Activity` *is associated with* `Offered Work Experience`' (with the `Offered Work Experience` concept from Job Seeker Ontology), '`Activity` *is associated with* `Job Vacancy`' (with the `Job Vacancy` concept from Job Offer Ontology) and '`Activity` *is associated with* `Requested Work Experience`' (with the `Requested Work Experience` concept from Job Offer Ontology).
- *Occupation Ontology.* This ontology models knowledge of occupations and job categories. It is based on the ISCO-88 (COM)[8], ONET[8] and European Dynamics classification of occupations. ISCO-88 (COM) and ONET are expressed in MS Access database format; European Dynamics classification of occupations is stored in an ORACLE database table. ISCO-88 (COM) is a classification of 520 occupations; ONET is a classification of 1167 occupations and the European Dynamics classification has 84 occupations. The resultant Occupation Ontology has 609 concepts. For this ontology we have extended manually the ISCO-88 (COM) classification with European Dynamics and ONET classifications for ICT

[6] http://ec.europa.eu/transport/home/drivinglicence/

[7] Available through RAMON Eurostat's Classifications Server at http://ec.europa.eu/comm/eurostat/ramon/

[8] http://online.onetcenter.org/

occupations. In this case we have defined a concept for every element of the resulting extended taxonomy in order to preserve the hierarchy. This ontology defines a concept taxonomy based on the aforementioned standards, in which the most general concept is the `Occupation` concept. The `Occupation` concept has four binary relations: '`Occupation` *is associated with* `Objective`' (with the `Objective` concept from the Job Seeker Ontology), '`Occupation` *is associated with* `Offered Work Experience`' (with the `Offered Work Experience` concept from the Job Seeker Ontology), '`Occupation` *is associated with* `Job Vacancy`' (with the `Job Vacancy` concept from the Job Offer Ontology) and '`Occupation` *is associated with* `Requested Work Experience`' (with the `Requested Work Experience` concept from the Job Offer Ontology).

- *Education Ontology*. This ontology models knowledge of education level and education fields. The education fields are based on the FOET[9] and the education levels are based on the ISCED97[10] (International Standard Classification of Education); both of them are expressed in MS Access database format. FOET has 127 education fields and ISCED97 has 7 education levels. The resultant Education Ontology has 130 concepts. For the education levels we have defined the `Education Level` concept; and for every education level specified in ISCED97 a different instance of the `Education Level` concept is defined. For the education fields we have defined a concept for every element of the FOET taxonomy in order to preserve the hierarchy. The concept `Education` has four binary relations: '`Education` *has education level* `Education Level`', '`Education` *has education field* `Education Field`', '`Education` *is associated with* `Job Vacancy`' (with the `Job Vacancy` concept from the Job Offer Ontology) and '`Education` *is education of* `Job Seeker`' (with the `Job Seeker` concept from the Job Seeker Ontology).
- *Geography Ontology.* This ontology is based on the ISO 3166[10] country codes and the European Dynamics classifications: Continent and Region. The ISO 3166 is expressed in XML format; Continent and Region classifications are stored in ORACLE database. The ISO 3166 has 244 country codes and names; Region classification has 367 regions and Continent classification has 9 continents. The resultant Geography Ontology has four concepts, a `Location` as main concept, which is split into three subclasses: `Continent`, `Region` and `Country`. For every country element specified in the ISO 3166 a different instance of the `Country` concept is defined, so the `Country` concept has 244 instances. For every region element specified in the Region classification a different instance of the `Region` concept is defined, so the `Region` concept has 367 regions. Finally for every continent element specified in the Continent classification a different instance of the `Continent` concept is defined. An example instance of the `Continent` concept is `EU – Europe`. An example instance of the `Country` concept is `SPAIN – ES`. An example instance of the `Region` concept is

[9] Available through RAMON Eurostat's Classifications Server at http://ec.europa.eu/comm/eurostat/ramon/
[10] http://www.iso.org/iso/en/prods-services/iso3166ma/index.html

Galicia. The Country concept has four binary relations: 'Country *is nation of* Job Seeker' (with the Job Seeker concept from the Job Seeker Ontology), 'Country *is residence of* Job Seeker' (with the Job Seeker concept from the Job Seeker Ontology), 'Country *is located in* Continent' and 'Country *has region* Region'.

- *Labour Regulatory Ontology.* This ontology is based on the LE FOREM[11] classifications ContracTypes and WorkRuleTypes, both of them expressed in XML format. ContractTypes classification has ten contract types and WorkRuleTypes has 9 work rule types. The resultant Labour Regulatory Ontology has 2 concepts. For every type of work condition or contract type considered by LE FOREM, a different instance of one of these two concepts (Contract Type or Work Condition) is included in the ontology. An example instance of the Contract Type concept is Autonomous. An example instance of the Work Condition concept is Partial time. The Work Condition concept has 2 binary relations: 'Work Condition *is associated to* Objective' (with the Objective concept from the Job Seeker Ontology) and 'Work Condition *is associated with* Job Vacancy' (with the Job Vacancy concept from the Job Offer Ontology). The Contract Type concept has two binary relations: 'Contract Type *is associated to* Objective' (with the Objective concept from the Job Seeker Ontology) and 'Contract Type *is associated with* Job Vacancy' (with the Job Vacancy concept from the Job Offer Ontology).
- *Language Ontology.* This ontology is based on the ISO 6392[12] and the Common European Framework of Reference (CEF)[13]. The ISO 6392 is expressed in HTML format and CEF is a description in PDF format. The ISO 6392 has 490 language codes and CEF has 6 language levels. The resultant Language Ontology has 3 concepts: Language, Language Level and Language Proficiency. For every language element specified in the ISO 6392 a different instance of the Language concept is defined, so the Language concept has 490 instances. For every language level element specified in the CEF a different instance of the Language Level concept is defined, so the Language Level concept has 6 instances. An example instance of the Language concept is eng – English. An example instance of the Language Level concept is A2 – Basic User. The Language concept has three relations: 'Language *is mother tongue of* Job Seeker' (with the Job Seeker concept from the Job Seeker Ontology), 'Language *is spoken by* Job Seeker' (with the Job Seeker concept from the Job Seeker Ontology) and 'Language *is evaluated by* Language Proficiency' (with the Language Proficiency concept from the Language Ontology).
- *Skill Ontology*. This ontology models knowledge of Skills and abilities. It is based on European Dynamics Skill classification. This classification has 291 skills and

[11] LE FOREM is an user partner of the SEEMP project, http://www.leforem.be/

[12] http://www.iso.org/iso/en/prods-services/popstds/languagecodes.html

[13] http://www.cambridgeesol.org/exams/cef.htm

it is stored in an ORACLE database table. The resultant Skill Ontology has 2 concepts: Skill `concept` with its subclass `ICT Skill`. For every skill element specified in the European Dynamic classification a different instance of the `ICT Skill` concept is defined. An example instance of the `ICT Skill` concept is `Hardware programming.` The `Skill` concept has two relations: '`Skill` *is associated with* `Job Vacancy`' (with the `Job Vacancy` concept from the Job Offer Ontology) and '`Skill` *is skill of* `Job Seeker`' (with the `Job Seeker` concept from the Job Seeker Ontology).

- *Competence Ontology*. This ontology defines a concept called `Competence` as a super class of the imported concepts `Skill`, `Language Proficiency` and `Driving License`. The `Competence` concept has three binary relations: '`Competence` *is associated with* `Vacancy`' (with the `Vacancy` concept from the Job Offer Ontology); '`Competence` *is competence of* `Candidacy`' (with the `Candidacy` concept from the Job Seeker Ontology) and '`Competence` *requires* `Education`' (with the `Education` concept from the Education Ontology).
- *Time Ontology.* This ontology is based on DAML ontology[14] and it is expressed in OWL format. The main concepts of this ontology are `Instant` and `Interval`, which are subclasses of `Temporal Entity`. `Instant` is linked to `Interval` through the properties of `begins`, `ends`, `inside` and `begins or in`. `Instant` is also linked to an instant temporal description, which is a concept with the properties of `second`, `minute`, `hour`, `day`, `month`, `year` and `time zone`. `Interval` has the subclass `proper interval`, which is related with itself through the relations '*interval equals*', '*interval before*', '*interval starts or finishes*', etc. Proper intervals can be concatenated through the relation '*concatenation*'.

Finally we present the Reference Ontology statistics. The Reference Ontology is composed of thirteen modular ontologies. The Reference Ontology has 1609 concepts, 6727 class attributes, 60 instance attributes, 94 *ad hoc* relationships, 1674 instances and 20 axioms.

5 Local ontologies building process

In this section we provide some guidelines for the building process of the local ontologies, each ES uses its own Local Ontology, which describes the employment market in its own terms. Based on the proposed SEEMP architecture, the possible options for building the local ontologies are: building local ontologies from the RO, and building local ontologies as a reverse engineering process from ES schema sources. Next, these options will be explained.

[14] http://cs.yale.edu/homes/dvm/daml/time-page.html

5.1 Building local ontologies from the Reference Ontology

The building process is structured/guided by the architecture of the Reference Ontology and scoped with applications needs. In this sense, we will need to extend some already defined elements, to remove unnecessary elements, or to add new application dependent elements that appear in each ES schema source. The result of this should be a RO friendly "local" ontology. Thanks to this similarity, mappings between local ontologies and RO will not be complex. But on the other hand, mappings between local ontologies and ES schema sources will be complex. Regarding the evolution and change propagation dimension we have:

- Changes in the RO imply a change in the mappings between local and RO as well as probably changes in the mappings between the local ontologies and the ES schema sources.
- Changes in the RO imply a change in the local ontology; in this case, the mappings RO – local ontology would remain as they were. The mappings between the local ontologies and the ES schema sources should be updated.
- Changes in the ES schema sources imply changes in its local ontology (probably the part that is not a mirror of the RO) and the mappings between local ontologies and ES schema sources, and probably minor changes in the mappings between local ontology and the RO. This last consequence is especially interesting because the changes in the ES schema sources will be the most frequent in the scenario posed by SEEMP.

5.2 Building local ontologies as a reverse engineering process from the ES schema sources

In this case, mappings between local ontologies and schema resources should not be complex. On the other hand, complex mappings will appear between the Local and RO. The building process requires more sophistication of knowledge engineering and good acquaintance of all the data and their structures of the application: not easily found skill set in ES or any other operational/research organizations. Regarding the evolution and change propagation dimension we have:

- Changes in the ES schema sources imply a change in the local ontologies and, consequently, in mappings between sources and local ontologies, but not necessarily in mappings between local and the RO.
- Changes in the RO imply changes in the mappings between local ontologies and the RO, but it is not necessary to modify anything at the ES level.

5.3 Approach followed by SEEMP

In SEEMP project we adopt a different option depending on the part of the local ontology to be built. On one hand, we select option 1 (building local ontologies from the RO) for Job Seeker and Job Offer ontologies and other general purpose ontologies like, for example, the Time Ontology. On the other hand, we select option 2 (building local ontologies as a reverse engineering process from ES schema resources) for Occupation, Education, Economic Activity, Language, Compensation, Labour Regulatory, Skill and Driving License ontologies.

The reason of selecting option 1 for Job Seeker and Job Offer ontologies is because there are not significant differences between these ontologies and the way how each ES structures job seeker and job offer information. Consequently mappings between local ontologies and RO will be simple, but mappings between local ontologies and ES schema sources will be complex. Furthermore, for the job seeker and job offer information local ontologies will share the same vocabulary (see [16]).

The reason of selecting option 2 for the ontologies mentioned above is because each ES may have its own classification systems for the related information. Nevertheless, it may happen that the local ontology shares some classification with the RO (as there will happen in the European scope with the driving license classification). In that case, the reverse engineering process for that classification will be skipped, and that part of the RO will be reused. By using option 2, mappings between local ontologies and RO will be complex, but mappings between local ontologies and ES schema sources will be simple.

6 Conclusions and Future Work

In this paper we have presented the methodological approach we followed for reusing existing human resources management standards/classifications in the SEEMP Project. We also described the resultant RO which acts as a common “language” in the form of a set of controlled vocabularies to describe the details of a job posting and the CV of a job seeker. The RO was developed with the proposed methodology and with the ontology engineering tool WebODE. Finally we have provided some guidelines for the building process of the local ontologies, and we conclude that the best option for building the local ontologies is building them following a hybrid approach that employs the best option for each part of the RO.

An important conclusion of the work that we have carried out is that we can reuse human resource management standards in new applications following a systematic approach. Moreover, it is clear such a reuse can save time during the development of the whole system. However, it is not always possible to reuse a standard in a straightforward way, because sometimes the ideal standard does not exist for different reasons (different scope, outdated, etc.), and it is necessary to extend some “imperfect” standard with additional terminology coming from other standards or *ad hoc* classifications. As future work, we will complete the development of the local ontologies for the SEEMP project. This work allows us to confirm that proposed systematic approach not only supports the creation of a RO from

standards/classifications properly, but it also facilitates the creation of the local ontologies related to this RO, since the building process of these local ontologies can take advantage of the already existing RO.

Acknowledgments. This work has been partially supported by the FP6 EU SEEMP Project (FP6-027347).

References

1. Bizer, C., Heese R., Mochol, M., Oldakowski, R., Tolksdorf, R., Eckstein, R.: The Impact of Semantic Web Technologies on Job Recruitment Processes; 7th International Conference Wirtschaftsinformatik (2005).
2. Bourse, M., Leclère, M., Morin, E., Trichet, F.: Human Resource Management and Semantic Web Technologies;1st International Conference on Information Communication Technologies: from Theory to Applications (ICTTA), (2004).
3. de Bruijn, J., Lausen, H., Polleres, A., Fensel, D.: The web service modeling language: An overview. In: Proceedings of the 3rd European Semantic Web Conference (ESWC2006), Budva, Montenegro, Springer-Verlag (2006).
4. Fensel, D., Lausen, H., Polleres, A., de Bruijn, J., Stollberg, M., Roman, D., Domingue, J.: Enabling Semantic Web Services - The Web Service Modeling Ontology. Springer (2006).
5. FOREM, UniMiB, Cefriel, ARL, SOC, MAR, PEP: User Requirement Definition D.1.SEEMP Deliverable (2006).
6. Gómez-Pérez, A., Fernández-López, M, Corcho, O.: Ontological Engineering. Springer Verlag. (2003).
7. Gómez-Pérez, A., Ramírez, J., Villazón-Terrazas, B.: Methodology for Reusing Human Resources Management Standards. In Nineteenth International Conference on Software Engineering and Knowledge Engineering (SEKE'07), Boston, USA, July, (2007)
8. Grüninger, M, Fox, MS.: Methodology for the design and evaluation of ontologies In Skuce D (ed) IJCAI95 Workshop on Basic Ontological Issues in Knowledge Sharing, (1995) pp 6.1-6.10
9. Gualtieri, A., Ruffolo, M.: An Ontology-Based Framework for Representing Organizational Knowledge. In: Proceedings of I-KNOW '05. Graz, Austria, June, (2005).
10. http://proton.semanticweb.org/. Last visited: August 14, 2007.
11. http://www.hr-xml.org. Last visited: August 14, 2007.
12. Jarrar, M.: Ontology Outreach Advisory - The Human Resources and Employment Domain Chapter. http://www.starlab.vub.ac.be/OOA/OOA-HR/OOA-HR.html
13. Legge, K.: Human Resource Management: Rhetorics and Realities. Anniversary ed. Macmillan. (2005).
14. LFUI, CEFRIEL, ED, TXT, UJF, UniMib, UPM: Supporting State of the Art. D.3.2 SEEMP Deliverable (2006).
15. Mochol, M., Paslaru, E.: Simperl: Practical Guidelines for Building Semantic eRecruitment Applications, International Conference on Knowledge Management (iKnow'06), Special Track: Advanced Semantic Technologies (2006).
16. Swartout, W., Patil, R., Knight, K., Russ, T.: Towards Distributed Use of Large-Scale Ontologies, AAAI-97 Spring Symposium on Ontological Engineering, Stanford University, May, (1997).

Literature-driven, Ontology-centric Knowledge Navigation for Lipidomics

#Rajaraman Kanagasabai[1], #Hong-Sang Low[2], Wee Tiong Ang[1], Anitha Veeramani[1], Markus R. Wenk[2], Christopher J. O. Baker*[1],

[1] Data Mining Department, Institute for Infocomm Research, Singapore.
cbaker@i2r.a-star.ediu.sg
[2] Department of Biochemistry and Department of Biological Sciences, Centre for Life Sciences, Singapore.

As the semantic web vision continues to proliferate a gap still remains in the full scale adoption of such technologies. The exact reasons for this continue to be the subject of ongoing debate, however, it is likely the emergence of reproducible infrastructure and deployments will expedite its adoption. We illustrate the recognizable added value to life science researchers gained through the convergence of existing and customized semantic web technologies (content acquisition pipelines supplying legacy unstructured texts, natural language processing, OWL-DL ontology development and instantiation, reasoning over A-boxes using a visual query tool). The resulting platform allows lipidomic researchers to rapidly navigate large volumes of full-text scientific documents according to recognizable lipid nomenclature, hierarchies and classifications. Specifically we have enabled searches for sentences describing lipid-protein and lipid-disease interactions.

1 Introduction

A series of existing technologies are now recruited along with semantic technologies to build scientific information systems delivering enriched value-added performance [1]. In particular there is an increasing need to link relevant content to semantic web infrastructure either by tagging existing web content and linking it to semantic metadata [2] or by indexing / summarizing legacy formats using algorithms focused on raw text analysis. In this latter case, where NLP approaches are now well established there would appear to be a complementary fit. Specifically the results of text analysis such as marked up text segments, which are typically deposited in relational databases, can be repurposed as instances to precisely defined concepts in ontologies. Likewise the relations between such named entities in text segments can also be instantiated to knowledge-bases. Such knowledgebases can represent a searchable summary of large volumes of literature [3]. Ontologies can provide richly cognitive query models to instantiated knowledgebases and in conjunction with reasoning engines can facilitate instance retrieval for knowledge discovery tasks. Here we focus on a contemporary application domain, Lipidomics, with the goal of

building an ontology-centric navigation platform to facilitate knowledge discovery for life scientists.

In section 2 we describe the architecture supporting the platform. In section 3 we introduce the *status quo* and current challenges in lipid research motivating for the development of the lipid ontology, which we also describe. In section 4 we describe the content acquisition strategy, natural language processing and the lipid-specific ontology instantiation strategy. In section 5 we describe the features of the knowledge navigator interface, discuss user scenario and query paradigms for interrogating the scientific literature.

2 Ontology-centric Content Delivery Platform

The outline of our platform is shown in Figure 1. It comprises of a content acquisition engine that drives the delivery of literature. This engine takes user keywords and retrieves full-text research papers from distributed public repositories and converts them to a custom format ready for text mining. A workflow of natural-language processing algorithms identifies target concepts or keywords and tags individual sentences according to the terms they contain. Sentences are instantiated (as A-boxes) using a custom designed java program to the ontology's literature specification (sentence concept) and relations to instances of each target concept found in the sentence are added into the ontology. The fully instantiated ontology is reasoned over using the reasoning engine RACER and it's A-box query language nRQL [4]. A custom built visual query interface, described in section 5, facilitates query navigation over instantiated object properties and visualization of datatype properties in the ontology.

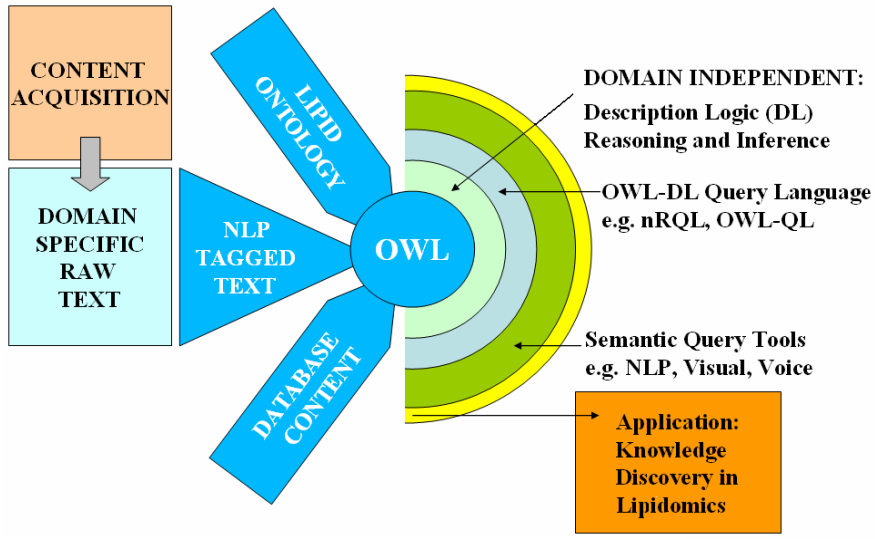


Fig. 1. Ontology-centric knowledge navigation system architecture

3

3 Lipids and Lipidomics

Lipids and their metabolites have a very crucial role in the biology and cellular functions of many living organisms. They are used for energy storage, serve as the structural components of cell membranes, and constitute important signaling molecules. Consequently lipids play diverse and important roles in nutrition and health: Imbalance or abnormality in lipid metabolism often accompanies diseases such as Alzhemer's syndrome, hypercholesterolemia and cancer. Lipidomics [5] is an emerging biomedical research field with important applications in the development of drugs and biomarkers for diseases e.g. cancer and diabetes. In order to attain a better understanding of the role of lipids in physiological processes, scientists use high throughput technology in the analysis of lipid composition of living organisms. Lipidomics generates large amounts of chemical, biological, analytical data that need to be integrated and analyzed in a systematic manner. A major challenge in this regard is the lack of consistent classification for lipids.

3.1 Lipid Classification Challenges

Lipids, unlike their protein counterparts, do not have a systematic classification and nomenclature that is widely adopted by biomedical research community. To address this problem, IUPAC-IUBMB [6] developed a standardized, systematic nomenclature for lipids. The IUPAC nomenclature suffers, however, from several drawbacks. Firstly, it has not gained widespread adoption since the systematic naming of lipids according to their structures can become long and cumbersome. Furthermore the IUPAC naming scheme was often misunderstood by scientists leading to the generation of many pseudo-IUPAC names that are neither chemically or scientifically sound. Given that the IUPAC naming scheme emerged in 1976, the naming scheme has not evolved since then to accommodate the large number of novel lipid classes that have been discovered in the last 3 decades.

In this context different lipid research groups developed their own classifications of lipids which are usually very narrow and only sound for a restricted lipid category. As a result, the same lipid molecule can be classified in many different ways, and be placed under different types of classification hierarchy. A single lipid can be associated with a plethora of synonyms. Furthermore, most of these classification systems are not scientifically sound and hence, create a lot of problems for the systematic analysis of lipids.

The LIPIDMAPS consortium [7] recently developed a scientifically robust and comprehensive chemical representation and classification system that incorporates a consistent nomenclature that is closely aligned to IUPAC nomenclature yet extensible to include new lipids without a systematically defined IUPAC name. Adoption of this standard has been gradual and many research groups still use synonyms or old names. More importantly legacy literature resources predominantly contain instances of lipid synonyms not yet linked to the LIPIDMAPS systematic name or any chemically sound classification.

3.2 Lipid Ontology

It is with the above mentioned problems in mind, we developed the Lipid Ontology. The rationale behind the Lipid Ontology is manifold: (i) it serves to connect the pre-existing/legacy lipid synonyms found in literature or other databases to the LIPIDMAPS classification system; (ii) it serves as a data model to manage information on lipid molecules, define features and declare appropriate relations to other biochemical entities i.e. proteins, diseases, enzymes and pathways; (iii) it serves as an integration and query model for one or more data warehouses of lipids information (iv) it serves as a flexible and accessible format for defining the current systematic classification of lipids and lipid nomenclature, which is particularly relevant to the discovery of new lipids and lipid classes that have yet to be systematically named. The ontology currently has a total of 668 concepts and 74 properties.

The Lipid ontology emerged from a data-warehouse schema developed [8] to house lipid information and lipidomics data. Consequently the ontology inherited certain features of the data model. Information about individual lipid molecules is modeled under the Lipid and Lipid Specification concepts. The Lipid concept is a sub-concept of Small_Molecules, subsumed by the super-concept Biomolecules. Under the Lipid concept are the classes defined in the LIPIDMAPS systematic classification hierarchy. The hierarchy currently consists of 8 major lipid categories and has in total 352 lipid sub-concepts. Instances of these concepts are LIPIDMAPS systematic names of individual lipids.

The Lipid_Specification concept contains information about individual lipids and entails the following sub-concepts; Biological_Origin, Data_Specification (with a focus on high throughput data from Lipidomics), Experimental_Data (mainly mass spectrometry data values of lipids), Properties, Structural_Specification and Lipid_Identifier (that carries within it 2 other sub-concepts; Lipid_Database_ID and Lipid_Name). A Lipid instance (a systematic name) relates to individuals (equivalent to attributes/column data in a database table) from Lipid_Specification via different properties, e.g *has_Mass_Spectra_Data_Values*

Relationship with other non-lipid databases:

In addition, each Lipid instance is related to other databases via the *has_DatabaseIdentifier* property. The *has_DatabaseIdentifier* property links a lipid individual to a database identifier. This ontology is designed to capture database information from the following databases, Swisprot [9], NCBI [10], BRENDA [11], KEGG [12]. The database record identifiers from each database are considered as instances of the respective database record.

Lipid Protein Interactions:

In order to model lipid protein interactions in the ontology, we added a Protein concept. The Protein concept is a descendant of Macromolecules and Biomolecules concepts. The systematic name of a protein from the SwisProt database is modeled as an instance of the Protein concept. A lipid instance is related to a protein instance by the *Interacts_With_Protein* property.

Lipids implicated in Diseases
Information of lipids implicated in disease can also be modeled. We added a primitive concept of Diseases in the ontology. A disease name is considered as a disease instance. A lipid instance is linked to a disease instance currently derived by text mining via a *hasRole_in_Disease* property.

Modelling synonyms
Due to a lack of systematic classification, a lipid molecule can have many synonyms. In the Lipid Ontology, a lipid instance is represented by its LIPIDMAPS systematic name. Synonyms of the lipids need to be modeled into the ontology. Lipid names synonyms are IUPAC names, lipid symbols and other commonly used lipid names, both scientific and un-scientific. Figure 2 shows the conceptualization of the Lipid _Specification which describes lipid names, and lipid databases identifiers. Specifically to address lipid synonyms we introduced 3 sub-concepts, IUPAC, Broad_Lipid_Name, Exact_Lipid_Name. IUPAC is directly subsumed by Lipid_Systematic_Name whereas Broad_Lipid_Name and Exact_Lipid_Name are subconcepts of Lipid_Non_Systematic_Name. For every LIPIDMAPS_systematic name, we anticipate multiple synonyms, an IUPAC name and one or more non-systematic names. The systematic name is related to an IUPAC name via a hasIUPAC_synonym property. This property is also used to relate a non systematic name to IUPAC name. Likewise, the non systematic name and IUPAC name are related to the systematic name via a hasLIPIDMAPS_synonym property.

In our conceptualization we also define a Broad_Lipid_Name as a broad synonym that can describe several lipid molecules. This concept is related to the Lipid concept and other lipid name concepts such as IUPAC, Exact_Lipid_Name via a hasBroad_Lipid_Synonym property. This means that if a non systematic name has one or more, IUPAC names/LIPIDMAPS systematic names/LIPIDMAPS identifiers/KEGG compound identifiers/LipidBank identifiers, it is actually a broad lipid synonym. In contrast, an exact lipid name is a non-systematic name that describe exactly 1 lipid molecule.

To resolve the problem of multiple synonyms in lipid nomenclature, we assembled a list of synonyms for lipids that can be found in the LIPIDMAPS database. These synonyms came from records in the KEGG and LipidBank databases that have an equivalent record found in LIPIDMAPS database. In effect, synonyms were taken from KEGG and LipidBank databases to enrich the lipid name list from LIPIDMAPS. These synonyms were subsequently grounded to their equivalent name in LIPIDMAPS. At present, the list has 36651 unique names, that covers 10103 LIPIDMAPS systematic names, 8468 IUPAC names, 22621 non-systematic names (22494 exact lipid name + 127 broad lipid names).

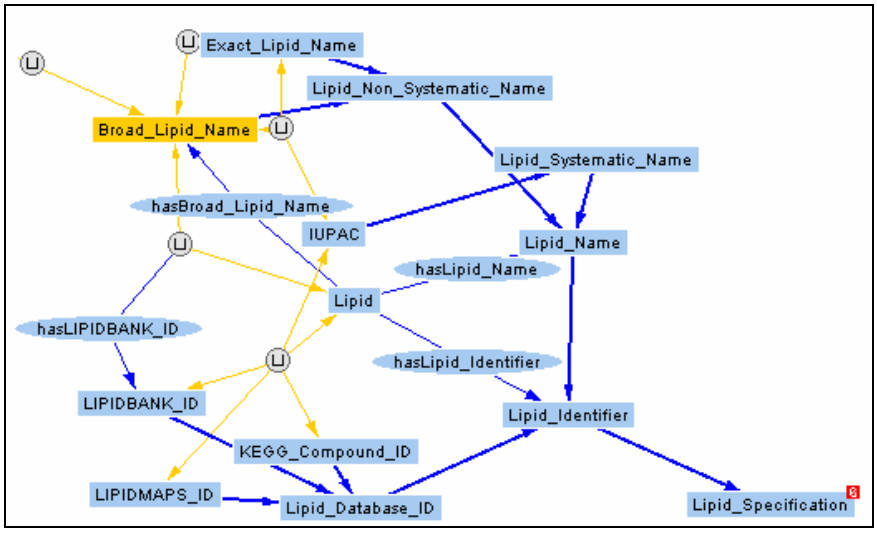


Fig. 2. Conceptualization of Lipid Specification illustrating the categorization of lipid names

Literature Specification
Of particular relevance to the application scenario, in Section 4 – is the provision of a knowledge framework where effective text mining of lipid related information can be carried out. This is supported by the Literature_Specification concept that has 10 sub-concepts, namely; Author, Document, Issue, Journal, Literature_Identifier (with a sub-concept PMID), Sentence, Title, Volume, Year. The Document concept is related to multiple concepts within the Literature_Specification hierarchy via several appropriate properties. The Document concept also has three datatype properties; author_of_Document, journal_of_Document, title_of_Document that are instantiated by author names, journal names and titles of the articles in the form of text strings. The sub-concept Sentence is related to Lipid and Protein via the property hasLipid and hasProtein. It is also related to Document via occursIn_Document property and has a datatype property, 'text_of_Sentence' that becomes instantiated by a text string from a Document found by text mining to have a lipid name and protein name or disease name occurring in the same sentence.

4 Ontology Population Workflow

In this section we describe the content acquisition; natural language processing and ontology instantiation strategy. Primarily ontology instances are generated from full texts using a text mining toolkit called the BioText Suite [13,14,15,16] which performs text processing tasks such as tokenization, part-of-speech tagging, named entity recognition, grounding, relation mining.

Content Acquisition: Our content acquisition engine takes user keywords and retrieves full-text research papers using a Pubmed search, parsing the search results and crawling the publishers' websites. Collections of research papers are converted from their original formats, e.g. pdf, to ascii text and passed to the text mining system.

Named Entity Recognition: The BioText Suite processes retrieved full-text documents and recognizes entities using a gazetteer. The gazetteer matches term lists against the token of a processed text and tags the terms found. It supports rules, e.g. for case-sensitive/case-insensitive matching, or sub/full-string matching. During gazetteer lookup, the ontology class of the term is also added as an attribute, and this is used later during the instantiation process to identify the right ontology class for population.

Separate term lists are employed for detecting lipids, proteins and diseases. The lipid name list was generated from Lipid DataWarehouse [8] containing lipid names from LIPIDMAPS, LipidBank and KEGG [12]. Each lipid name is identified by a LIPIDMAPS systematic name [17], IUPAC name, Common name and optionally other synonyms, along with a database identifier. As of April 2007, LIPIDMAPS contained 10103 entries. There were 2897 LipidBank entries and 749 KEGG entries linked to the corresponding entries in LIPIDMAPS via the database ID. All these linked entries were collapsed and grounded to their respective systematic name (explained in detail in the next paragraph). Term lists were created for each category of names: Systematic, IUPAC, broad and exact synonyms. The manually curated Protein name list from Swiss-Prot (http://au.expasy.org/sprot/) was used for grounding of proteins found in literature and further consolidated by combining all canonical names and synonyms. Grounding used the Swiss-Prot ID. A disease term list was created from the Disease Ontology of Centre for Genetic Medicine (http://diseaseontology.sourceforge.net) and used for grounding disease names.

Normalization and Grounding: Entities recognized in the previous step need to be normalized and grounded to the canonical names, before instantiation. Protein names were normalized to the canonical names entry in Swiss-Prot. The grounding is done via the Swiss-Prot ID. For lipid names, we define the LIPIDMAPS systematic name as the canonical name, and for grounding, LIPIDMAPS database ID is used. Disease names are grounded via the ULMS ID.

Relation Detection: In this step we identify the Lipid-Protein and Lipid-Disease relations, using the grounded entities. We adopt a simple relation mining approach whereby two entities are said to be related if they co-occur in a sentence. Thus, every document is parsed to extract sentences and then co-occurrence detection is invoked. To reduce false positives, we require that the sentence contain one relation keyword. All other sentences are skipped. From the resulting collection, Lipid-Protein or Lipid-Disease pairs are returned along with the respective sentences in which they co-occur. The latter could possibly be used for human validation during the knowledge retrieval step.

Ontology Population: Here we collect all the mined knowledge from the previous steps to instantiate the ontology. The grounded entities are instantiated as class instances into the respective ontology classes (as tagged by the gazetteer), and the relations detected are instantiated as Object Property instances. We wrote a custom script using the JENA API (http://jena.sourceforge.net/) for this purpose.

4.1 Population Performance Analysis

To the best of our knowledge, there is no lipidomics-related corpus for evaluating literature mining and ontology population. We are in the process of building one with biologists from the Lipidomics group at the Centre for Life Sciences, NUS, Singapore. For this paper, we provide a preliminary performance analysis of the text processing and ontology population system by assessing the complete lipid-protein interaction mining task. This started with a PubMed literature search for the query "lipid interact* protein" with our content acquisition engine that identified 495 search results for the time period July 2005 to April 2007. 262 full-text papers were successfully downloaded. The remaining papers were from journals not subscribed to by our organization or had no download-able link to the full paper.

After named entity recognition and relation detection, 121 documents in which no lipid-protein relations were detected were omitted. Ontology instantiation was carried out with the remaining 141 documents. The named entity recognition (NER) component detected 186 lipid names and 528 protein names. After normalization and grounding, there were 92 LIPIDMAPS systematic names, 52 IUPAC names, 412 exact synonyms, 6 broad synonyms and 319 protein names. Cross-links to 59 Lipidbank entries and 41 KEGG entries were also established. The brute-force co-occurrence detection yielded over 1356 sentences. After the relation word filtering, there were only 683 interaction sentences. The 92 LIPIDMAPS names were instantiated into 35 unique classes under the Lipid name hierarchy, at an average of about 2.6 lipids per class. The ontology instantiation process took 22 seconds overall. The experiments have been done on a 3.6 Ghz Xeon Linux workstation with 4 processors and 8GB RAM.

5 Knowledge Navigation for Lipidomics

The development of the ontology-centric knowledge-delivery platform results in a rich knowledge base of instantiated text segments. Typically such an OWL-DL knowledgebase is accessed through highly expressive DL-query languages that have complex syntactic query languages not suitable for domain experts [18]. nRQL is the prominent OWL-DL query language that we used which extends the existing capabilities of RACER with a series of query atoms. While some tools exist which facilitate enhanced end user operability of this query language [19, 20, 21] these implementations are of academic prototype scale and their adoption has yet to be widespread. Here we describe a new tool for the navigation of A-box instances, in our case ‘text segments’ which allows users to build graphical queries which are converted to query language syntax and issued to the reasoner.

9

5.1 Knowlegator

The Knowledge Navigator (Knowlegator) receives OWL-DL ontologies as input and passes them to RACER, after which it enters into a dialogue with RACER and issues a series of commands to query elementary features of the ontology for visual representation in the components panel. The navigator consists of three main panels, a Components panel, the Editor panel and the Output panel (Figure 3). The Components panel renders the ontology as a tree structure of concepts, roles and instances. Concepts are pre-queried to retrieve their respective number of instances and occurrences of object properties. This panel allows drag and drop functionality for query formulation. The Editor Panel is structured as a tabbed pane providing rapid switching between groups of functionalities. The 'Ask a Question' Tab contains the query canvas where questions can be formulated by dragging and dropping an element from the tree structure in the Component panel. Each dropped item is associated with an automatically formulated nRQL query. Dragging a single concept invokes the retrieval of all the individuals of a particular concept. Likewise dragging a named role (object property) queries instances specified in the domain and range of the particular role. In the query canvas a complex query built by extending simpler queries through 'right click' enabled instantiated-object property lookup. A separate window shows a query result specifically in the bottom panel the full text of a sentence is rendered. In addition to facilitating nested role queries through domain-property-range expansion the tool facilitates the identification of (instantiated) relations between any two concepts dragged to the canvas. This provides users with additional entry point to building graphical queries which can be subsequently customized. This is achieved by an exhaustive cascade of nRQL role queries to the ontology.

5.2 Lipidomics Application Domain

The intended user of the system is a researcher who specializes in lipidomics. Lipidomics is a recent research methodology that measures the composition & fluctuation of lipids at the system level of a living system in a high throughput manner. This type of user would like to ascertain the identity of lipids found in his or her experimental work and obtain all other information associated to the lipid in question. In short, they are looking for a, *one stop shop,* knowledge aggregator. Typically, for post-experiment analysis, a user has to visit multiple website or read 5-6 papers to find out the information that they want. Even then, the information that they obtain may be fragmented. Such users are typically not IT savvy and probably only proficient with a Windows environment. When such users do adopt expert or customized software for their work, they can't do without an intuitive GUI interface. Furthermore spending too much learning a new system is not considered useful even if there is a longer term benefit.

10

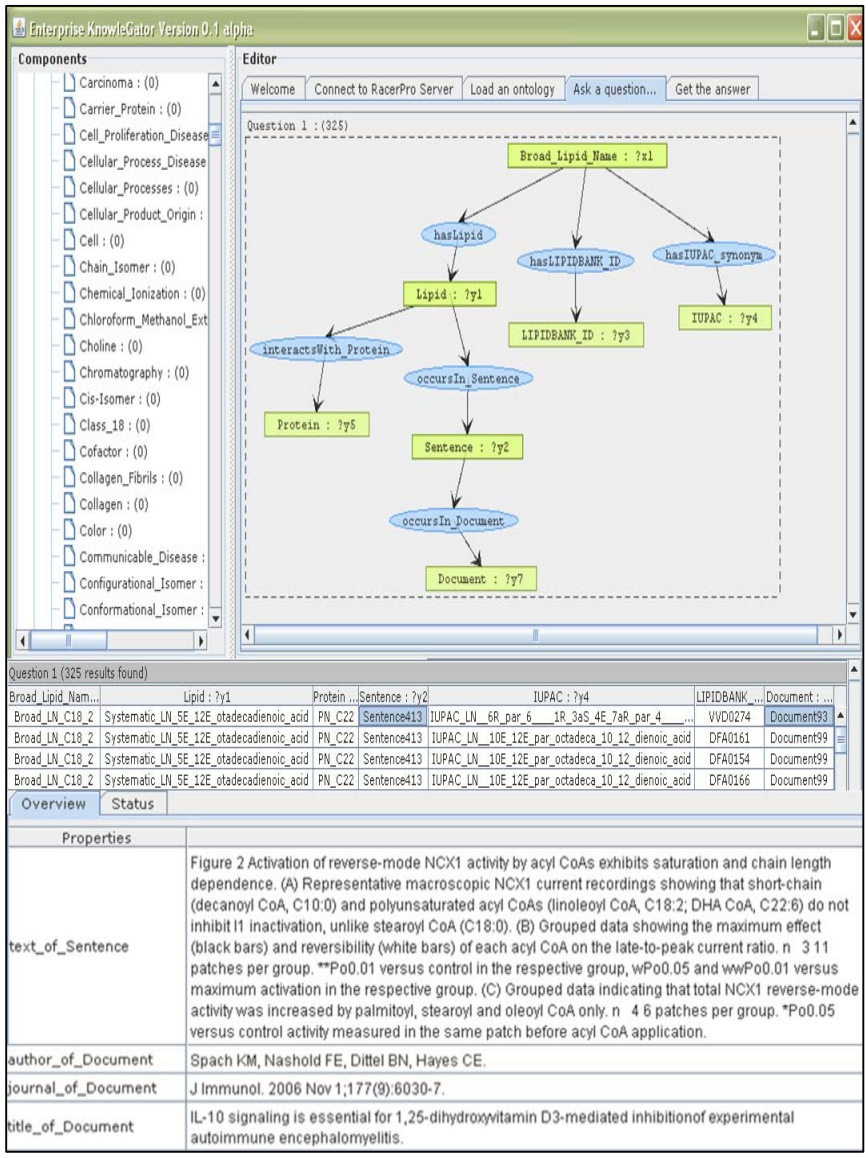


Fig. 3. Query interface of Knowlegator, showing a query for documents that contain sentences describing the interaction of proteins with lipids, and their corresponding lipid synonyms.

11

Lipidomics User Tasks:
The major knowledge-based task of a lipidomics researcher is to resolve the identity of a lipid entity to a given systematic lipid classification. The researcher can have multiple starting-points e.g. raw mass spec data, a common name from the literature or systematic name from an automated annotation pipeline, that must be translated to another classification system based on the users knowledge of lipid synonyms. Using a systematic lipid classification the user can determine or infer the possible functions / biochemical properties of the lipid. Further examination of the relationships in which a particular lipid or class of lipids participates e.g. which types of proteins a lipid interacts with, allows the researcher to make inferences regarding the metabolic process in which it participates or the role of the lipid in a cellular function or disease. Integral to these tasks is the frequent consultation with, and navigation of, the scientific literature using a variety of systematic and non-systematic lipid keywords.

Use Case Description:
The use case scenario of our system initiates with the pre-selection of collection of documents identified by an ad hoc query to a literature database or search engine and identifies relevant abstracts. The user identifies which collection of documents to review and sends them for full-text processing and the creation of a knowledgebase. The user does not require online access to the knowledgebase immediately after document selection and can wait for full text processing to complete. It is relevant to mention that major pharmaceutical corporations regularly make significant financial investments in the manual curation (3 or more months at a time) of scientific literature to generate targeted knowledge bases. This work is often outsourced to smaller companies where labour costs are cheaper. Our approach mirrors this scenario where the decision for a search and the actual navigation of the retrieved documents is decoupled into separate tasks. Once the knowledgebase is created the user has ad-hoc access to the knowledgebase using the concepts and relations provided in the query model of the ontology. The query model has rich domain specific semantics that the lipidomics user is already familiar with i.e. the systematic classification schemes of lipids. In our case the lipid ontology was built by a team (conceptualized by the lipid experts and created by ontology engineers).

5.3 Query Paradigm Comparison

Whereas searching online scientific literature databases provides sufficient ad-hoc access to abstracts it does not facilitate deep search of the full text of the documents. Systematic names of enzymes, lipids and other medical terminologies are rarely included in scientific abstracts. Additionally queries to online literature databases are limited to keyword and Boolean expressions and the traversal of literature resources is frequently based on author supplied keywords. More advanced searches of the scientific literature rely either on browsing manually curated database entries or searching the results of text mining platforms deposited in relational databases. These typically have form based web interfaces limiting the types of queries that can be issued to the database. As a result users may be required to directly interact with the relational database to pose queries that were not perceived necessary or relevant when the web portal to the database was created. This is not untypical. It is at this point where the user loses access to the knowledge resources.

For this reason we further comment on the capabilities of the ontology-centric visual query paradigm by contrasting query through the Knowlegator interface with that of a the same query made directly to a relational database with equivalent content. For example, querying for documents which contain sentences describing “lipids that interact with proteins” can be more easily formulated from the ontology by visual query than in the relational database scenario (Figures 3 and 4). Figure 3 also highlights the inclusion of Broad Lipid Names in the query such that synonyms of the lipids, in different classification schemes can be readily queried at the same time. In the database scenario, to make this query each concept should be modeled into a separate table and the relations are modeled into additional connection tables (Figure 4) to reduce redundancies. Every time there is a new relation, there must be a new relationship table. The SQL query (Figure 4) for the mentioned statement would require multiple table-joins and is not particularly intuitive to a user with no prior knowledge of the database. Using Knowlegator, the statement can be easily retrieved through a series of right mouse-clicks and selecting the required options.

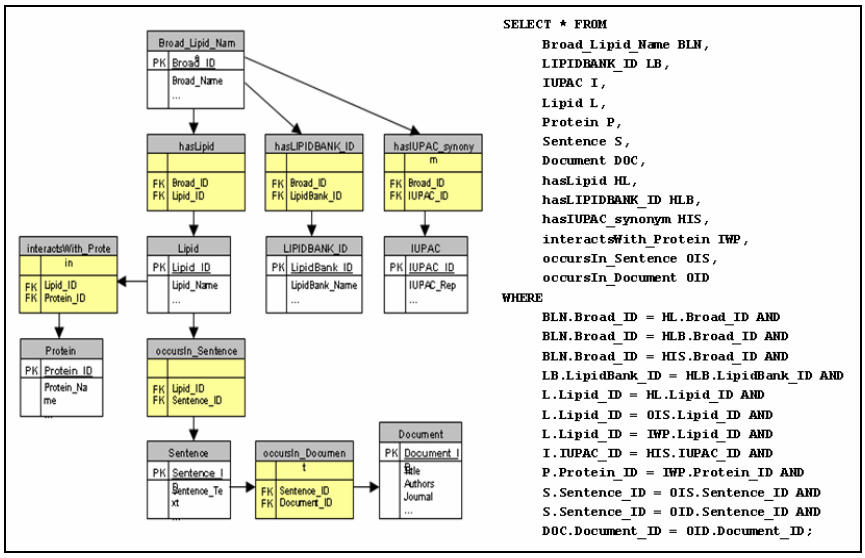


Fig. 4. A relational database query for documents that contain sentences describing the interaction of proteins with lipids and their corresponding lipid synonyms.

6 Conclusion

The challenge in our Lipidomics scenario is the navigation of large volumes of complex biological knowledge typically accessible only in legacy unstructured full-text format. This was achieved through the coordination of distributed literature sources, natural language processing, ontology development, automated ontology instantiation, visual query guided reasoning over OWL-DL A-boxes. The major innovations were to: translate the results of natural language processing to instances of a ontology domain model designed by end users; exploit the utility of A-box reasoning to facilitate knowledge discovery through the navigation of instantiated ontologies and thereby enable scientists to identify the importance of newly identified lipids through their known associations, synonyms and interactions with classes of protein and diseases.

References:

[1] Baker, C.J.O. and Cheung, K.H. (Eds.) (2006) Semantic Web: Revolutionizing Knowledge Discovery in the Life Sciences, Springer.
[2] B. Aleman-Meza, M. Nagarajan, C. Ramakrishnan, L. Ding, P. Kolari, A.P. Sheth, I.B. Arpinar, A. Joshi, T. Finin, Semantic Analytics on Social Networks: Experiences in Addressing the Problem of Conflict of Interest Detection, 15th International World Wide Web Conference (WWW2006), Edinburgh, Scotland, UK, May 2006
[3] Witte, R., Kappler, T. and Baker, C.J.O. (2006a) 'Ontology Design for Biomedical Text Mining', In Baker, C.J.O. and Cheung, K.H. (Eds.) (2006) Semantic Web: Revolutionizing Knowledge Discovery in the Life Sciences, Springer. Chap. 13, pp. 281–313.
[4] Haarslev, V., Moeller, R., Wessel, M.: Querying the semantic web with racer + nrql. In Bechhofer, S., Haarslev, V., Lutz, C., Moeller, R., eds.: CEUR Workshop Proceedings of KI-2004 Workshop on Applications of Description Logics (ADL 04), Ulm, Germany (2004)
[5] Wenk MR. The emerging field of Lipidomics. Nature Review Drug Discovery, July 2005, Vol. 4, No. 4, pp.594-610.
[6] IUPAC-IUB Commission on Biochemical Nomenclature (CBN). The nomenclature of lipids (recommendations 1976). 1977. Eur. J. Biochem. 79: 11–21; 1977. Hoppe-Seylers Z. Physiol. Chem. 358: 617–631; 1977. Lipids. 12: 455–468; 1977. Mol. Cell. Biochem. 17: 157–171; 1978. Chem. Phys. Lipids. 21: 159–173; 1978. J. Lipid Res. 19: 114–128; 1978. Biochem. J. 171: 21–35 (http://www.chem.qmul.ac.uk/iupac/lipid/).
[7] Fahy E, Subramaniam S, Brown HA, Glass CK, Merrill AH Jr, Murphy RC, Raetz CR, Russell DW, Seyama Y, Shaw W, Shimizu T, Spener F, van Meer G, VanNieuwenhze MS, White SH, Witztum JL, Dennis EA. A comprehensive classification system for lipids. Journal of Lipid Research, May 2005, Vol. 46, pp.839-862.
[8] Koh J and Wenk MR Lipid Data Warehouse (Unpublished)
[9] Boeckmann B, Bairoch A, Apweiler R, Blatter M-C, Estreicher A, Gasteiger E, Martin MJ, Michoud K, O'Donovan C, Phan I, Pilbout S, Schneider M. The SWISS-PROT protein knowledgebase and its supplement TrEMBL in 2003. Nucleic Acids Research, January 2003, Vol 31, No.1, pp.365-370.

[10] D.L. Wheeler, C. Chappey, A.E. Lash, D.D. Leipe, T.L. Madden, G.D. Schuler, T.A. Tatusova, B.A. Rapp, Database resources of the national center for biotechnology information, Nucl. Acids Res. 28 (1) (2000) 10–14,
[11] I. Schomburg, A. Chang, C. Ebeling, M. Gremse, C. Heldt, G. Huhn, D. Schomburg,BRENDA, the enzyme database: updates and major newdevelopments, Nucl. Acids Res. 32 (Database issue) (2004) D431–D433.
[12] Kanehisa M, Goto S, Kawashima S, Okuno Y, Hattori M. The KEGG resource for deciphering the genome. Nucleic Acid Research, January 2004, Vol. 32(Database issue), pp.D277-280.
[13] BioText Suite: Tools for Mining Biomedical Literature. http://research.i2r.a-star.edu.sg/kanagasa/BioText/. 2006.
[14] Doreen Tan, SL Goh, K. Rajaraman, S. Swarup, VB Bajic, Tiow Suan Sim. A user-friendly text-mining tool for streptomyces biology. Combined Scientific Meeting, Singapore, 2005.
[15] Kanagasabai Rajaraman, Zuo Li, V.B. Bajic. Extracting Transcription Factor Relations from Biomedical Texts. 5th Hugo Pacific Meeting & 6th Asia-Pacific Conference on Human Genetics, Singapore, Nov 2004.
[16] Kanagasabai Rajaraman and Ah-Hwee Tan. Mining Semantic Networks for Knowledge Discovery. IEEE Conference on Data Mining (ICDM'03), Florida, USA, pp 363-366, 2003.
[17] Sud M, Fahy E, Cotter D, Brown A, Dennis EA, Glass CK, Merrill AH Jr, Murphy RC, Raetz CR, Russell DW, Subramaniam S. LMSD: LIPID MAPS structure database. Nucleic Acid Research, January 2007, Vol. 35(Database issue), pp.D527-D532.
[18] Smith, B., Ceusters,W., Klagges, B., Kohler, J., Kumar, A., Lomax, J., Mungall, C., Neuhaus, F., Rector, A., Rosse, C.: Relations in biomedical ontologies. Genome Biology 6 (2005)
[19] A Fadhil and V. Haarslev, GLOO: A Graphical Query Language for OWL ontologies . OWL: Experience and Directions 2006, Athens, 2006.
[20] Kosseim, L., Siblini, R., Baker, C.J.O. and Bergler, S. (2006) ‘Using Selectional Restrictions to Query an OWL Ontology’, In International Conference on Formal Ontology in Information Systems (FOIS 2006), Baltimore, Maryland, USA.
[21] Baker, C.J.O., Shaban-Nejad, A., Su, X., Haarslev, V. and Butler, G. (2006a) ‘Semantic Web Infrastructure for Fungal Enzyme Biotechnologists’, Journal of Web Semantics, Vol. 4, No. 3. Special issue on Semantic Web for the Life Sciences.

Enabling the Semantic Web with Ready-to-Use Web Widgets

Eetu Mäkelä, Kim Viljanen, Olli Alm, Jouni Tuominen, Onni Valkeapää, Tomi Kauppinen, Jussi Kurki, Reetta Sinkkilä, Teppo Känsälä, Robin Lindroos, Osma Suominen, Tuukka Ruotsalo, and Eero Hyvönen

Helsinki University of Technology (TKK) and University of Helsinki
first.last@tkk.fi, http://www.seco.tkk.fi

Abstract. A lot of functionality is needed when an application, such as a museum cataloguing system, is extended with semantic capabilities, for example ontological indexing functionality or multi-facet search. To avoid duplicate work and to enable easy and cost-efficient integration of information systems with the Semantic Web, we propose a web widget approach. Here, data sources are combined with functionality into ready-to-use software components that allow adding semantic functionality to systems with just a few lines of code. As a proof of the concept, we present a collection of general semantic web widgets and case applications that use them, such as the ontology server ONKI, the annotation editor SAHA and the culture portal CultureSampo.

1 Introduction

To implement new semantic applications or to extend existing information systems (e.g. a museum cataloguing system) with semantic capabilities requires a lot of functionality dealing specifically with ontologies and metadata. Currently, needed functionalities are typically created for each application individually, requiring a lot of work, time and specific skills. Being able to lower these implementation costs would be hugely beneficial to the adoption of the Semantic Web [1] as a whole. On a general level, there are three tasks in any semantic information environment that need to be handled, either by humans or machines:

Semantic Content Consumption. Searching, browsing, visualizing and otherwise consuming semantic content. In this group belong for example library users in a semantic library environment and visitors of a semantic museum portal [2]. RSS aggregation services are a programmatic example.

Content Indexing. The production of semantic metadata by indexing and publishing content with references to shared vocabularies (e.g. museum curators indexing exhibits). Sometimes, the end-users themselves fill the role of content indexers, as in the social bookmarking site del.icio.us[1] and photo sharing site Flickr[2].

[1] http://del.icio.us/
[2] http://flickr.com/

Ontology Maintenance and Publishing. Creating and maintaining the ontologies used as references for both semantic indexing and retrieval. In organized fields, this is often done by dedicated information workers, but again in the case of Web 2.0 [3] sites, it may be the users themselves that develop their vocabulary in an ad-hoc manner alongside indexing.

We argue that in many cases, tasks in the contexts above contain many common general subtasks. Specifically, we have found at least the following common functionalities: 1) *concept and instance selection*, 2) *semantic linking*, 3) *concept and instance viewing*, and 4) *shared concept and instance storage and maintenance*. To avoid duplicate work and to enable more individuals and organizations to join the Semantic Web, we propose a *web widget*[3] approach, where these functionalities are created and published as ready-to-use software components which can easily and cost-effectively be added to applications.

A web widget is a reusable, compact software component that can be embedded into a web page or application to provide functionality. Most useful web widgets also combine on-line data resources with site data to create mash-ups, such as in usual use cases of the Google Maps and Google AdSense web widgets. Web widgets can also be combined together and published as new components, e.g. with the Yahoo! Pipes service[4]. What makes web widgets very interesting, is that they allow developers to easily add otherwise very complicated or costly features to virtually any application. For example, Google Maps provides a map and satellite image database of the planet Earth combined with search and browsing capabilities for all to use.

Semantic web widgets then, as we envision them, are software components that: 1) are hooked up with either Semantic Web data sources such as ontologies or instance databases, or the processed outputs of other components, 2) offer a single, compact functionality, yet do that as completely as possible (often including user interface elements), 3) are amenable to be combined with other components and data to solve complex problems, and 4) can be easily and cost-efficiently used for adding semantic functionalities to an application.

During our research, we found that while the semantic functionalities we want to implement themselves are general, the best implementation for them varies with the type of the underlying data they are to be hooked up with. For example, selection from a geographical location ontology naturally benefits from map-based user interfaces, actor ontologies need handling of pen names and transliterations [4], while concept ontologies such as the Suggested Upper Merged Ontology SUMO [5] and the health ontology SNOMED CT[5] require other methods. Therefore, we present several widget solutions to each of the tasks based on the different requirements of each content domain.

The context for this work is the goal of creating a national semantic web infrastructure in Finland [6], where critical Semantic Web resources, such as

[3] http://en.wikipedia.org/wiki/Web_widget
[4] http://pipes.yahoo.com/
[5] http://www.snomed.org/snomedct/

ontologies and the web widgets for using them, are published as centralized services.

In the following, as proof of the viability of the idea of semantic web widgets, we first present some of the components we have created for solving common subtasks, and then apply them to one combined mash-up service, the ONKI ontology server, and two end-user applications: the SAHA metadata editor for content creation and the CultureSampo cultural heritage portal. Finally, related work is presented, followed by discussion and suggestions for future work.

2 Common Subtasks in Semantic Applications and Widgets for Solving Them

2.1 Concept and Instance Selection

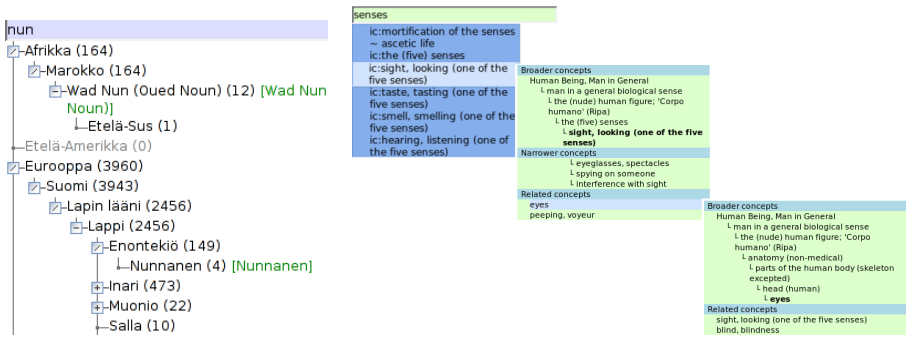
In ontological systems finding and selecting the right concepts and instances is a central task of its own in ontological user interfaces. For end-user applications, any search usually begins by first finding the right concepts with which to do the actual ontological querying [7]. For efficient semantic content indexing, accurate indexing entitities need to be found with as little effort as possible [8]. Also ontology developers need concept search when creating links between concepts, especially when developing distinct, yet heavily interlinked ontologies. In the following, web widgets to provide efficient concept selection in different situations are presented.

Semantic Autocompletion and Context Visualization When the user knows with relative certainty what they are looking for, and the labels of the entities do not overlap much, as in most non-instance vocabularies, a keyword search is a natural way of selecting concepts. For this, we have developed multiple text literal matching semantic autocompletion interfaces [9] that can be hooked into various semantic data sources to provide concept selection functionality. Particular among these are two that also expose the semantic contexts of the concepts matched.

There are two main reasons for desiring such functionality. First, this allows the user to get acquainted with the vocabularies and how they are organized. Second, particularly with large complex interlinked vocabularies, it is never guaranteed that the concept that first occurs to the user is the best one for their task. Showing the ontological context or otherwise derived concept recommendations is a powerful way of gently guiding the user and giving more options.

The first of the context exposing interfaces created, depicted in figure 1(a) shows the autocompletions directly inside a tree. This is applicable when the entities form a hierarchy and this is clearly the most important context for them. A particular case for this is in view-based search, where the view tree is usually already shown for visualization and selection purposes. The second is a navigation widget for exploring the contexts of matched entities, depicted in figure 1(b). Here the ontological context and otherwise derived concept recommendations

are shown to the user as a tree menu, with new levels of context opening when the user mouses over the concepts.



(a) inside a tree (b) with context navigation

Fig. 1. Semantic autocompletion

Map-based Search and Visualization URIs concerning geographic objects might carry coordinate information, e.g. in terms of WGS84 latitude and longitude. We have created a method, *n-point search* [10, 11], for selecting entities based on this kind of coordinate data. A search query in this method is done by pointing out n points on a map. The user clicks on the map and a search polygon is formed accordingly. If an area point of a certain place is found inside the user-given polygon, or a region-defining polygon is found to overlay the search polygon, the region instance is retrieved and added to the results.

We have also taken into account two special cases, namely, where $n = 1$ or $n = 2$. If $n = 1$ a circle is drawn around the point and the places that are inside the circle are retrieved. An alternative treatment would be to simply find the nearest places. If $n = 2$, we create a bounding box where the points are the opposite, e.g. South-West and North-East corners, of the search area. We have implemented the n-point search as a mash-up that itself uses Google Maps[6]. We used SVG [12] for drawing the polygon as a transparent layer on top of the map. In addition to selecting locations, the component is also able to visualize semantic content related to geographical locations on the map.

An example of using the component is depicted in figure 2. The n polygon corners are depicted as small crosses. The system has found from a historical place ontology [13] the municipality of Viipuri in annexed Karjala as it existed in 1906-1921, because its center points are situated within the user-specified search polygon. The municipality is visualized on the map as a red circle. The figure also depicts another useful feature of our component, in that it is able to overlay multiple maps of our own on top of the ones provided by Google [11]. Here, a

[6] http://www.google.com/maps

historical map overlay of the area shows how Viipuri looked at the time specified, which can be constrasted to how the place looks now.

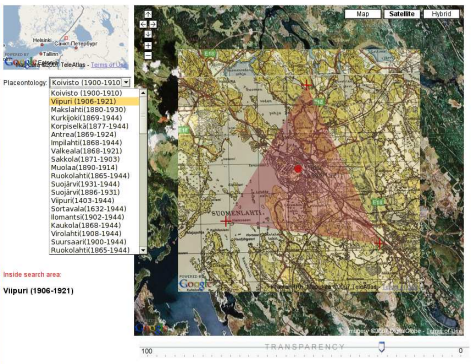


Fig. 2. Search and visualization using location data and overlaid maps.

Fig. 3. A floatlet (encircled) displays links to MuseumFinland.

Multi-Facet Search View-based, or multi-facet search is a paradigm that has recently become prominent as an easy to use interface for querying semantic content [14–16]. Here, the idea is to offer multiple views to different aspects of the content, both to visualize it and to select a subset from it by specifying constraints in the views [7].

We have implemented a general multi-facet search engine that plugged into an instance database uses the other created visualization and selection components as views. This also represents one of the more complex interaction patterns between components, as first the views provide constraint selection for the search engine, which then calculates a result set based on the instance database, and feeds this result set back to the views for visualization. As an example, the selection tree in figure 1(a) is actually part of such a configuration, with the numbers representing how many items annotated with that concept are in the current multi-facet search result set. An earlier version of the component [17] has already been used in the portals MuseumFinland[7] [2], Orava[8] [18], Veturi[9] [19] and SW-Suomi.fi[10] [20]

Concept Location, Disambiguation and Extraction from Text It is often useful to be able to locate concepts in textual resources. In annotating documents, suggestions for annotations can be found from the text [21, 22]. In web

[7] http://www.museosuomi.fi/
[8] http://demo.seco.tkk.fi/orava/
[9] http://demo.seco.tkk.fi/veturi/
[10] http://demo.seco.tkk.fi/suomifi/

browsing, on the other hand, semantic content can be linked to topics being discussed in the text [23].

We have implemented a component [8] that can locate concepts and instances in text documents based on the labels associated with them. The extraction process starts with document preprocessing, which in the case of HTML documents means extracting the textual content from the document. Then, the text is tokenized and lemmatized if needed. Next, the extraction component iterates over the tokenized document and finds strings corresponding to the concept labels. In cases where labels are ambiguous (e.g. "bank"), the component can make use of the ontological neighbourhoods of the concepts by counting nearby occurrences of neighbour concept labels of each candidate in order to guess the proper meaning. After finding the matches, the component then tags the occurences of the concepts and instances in the document and outputs this tagged copy.

2.2 Semantic Linking

Semantic metadata combined with logical rules makes it possible to automatically link related content together to support semantic browsing (semantic recommendations) [24, 2]. Automatic linking is an especially important feature in Semantic Web based systems where the content is typically aggregated from different sources. Here, manual linking is difficult because the content providers typically consider only the local view on their content excluding the global view on the aggregated content [25].

For automatic link generation, we have created functionalities based on three different techniques. First, used in the already mentioned portals MuseumFinland, Orava and SW-Suomi.fi is the rule-based linking server Ontodella [24], which is accessed by a simple HTTP request containing the URI of the current document. The system then responds, based on the item metadata and ontologies linked to it, with a set of related documents. Each recommendation also contains a human-readable explanation of the relation between the current and the recommended document. For example, when looking at a nautical flag in MuseumFinland, the object is linked to sailor's clothes because, based on the metadata and the ontologies, they are used in the same situation, seafaring.

Our second link generation component is based on calculating a similarity measure between items in the linked instance database using an event-based schema, which allows one to compare items annotated using dissimilar annotation schemas [26]. This is the component used in CultureSampo, the case application described later in this paper.

Finally, particularly for inter-portal semantic linking, we have exposed the multi-facet functionalities of our portals to mash-up use. For utilizing them, we propose the concept of *floatlets*, semantic linking widgets that can be easily plugged into any web page. Based on metadata and shared ontologies, the floatlet is able to make semantically relevant queries to the portals and show them in the context of the page. For example, figure 3 shows how the Finnish

Broadcasting Company's video archive[11] has been semantically linked based on metadata with relevant content in MuseumFinland. In the example, the current video is about the history of speed skating, which has also been described in its metadata. Based on this information, the floatlet is able to query for old skates from MuseumFinland. By clicking on the floatlet links, the skates can be examined in more detail in their original portal. From the semantic portal publisher's point of view, floatlets provide a new way for publishing and promoting their content on the web. By clicking on the floatlet links, new visitors move over to the floatlet's host portal.

The idea of floatlets is similar to Google AdSense[12] which is used for adding advertisements to web pages. However, in floatlets the returned links are based on explicit ontological annotations. This allows the web developer to specify in detail what information is to be linked, either manually or based on the metadata of the current web page.

2.3 Concept and Instance Views

To be able to comprehend the meanings of concepts and their relations, content visualization techniques are needed. The simplest way to approach this is to show the properties of a concept or an instance to the user, e.g. simply as a list of properties and their respective values. If a value is a resource, it can also act as a link to allow browsing the content.

Ontologies are typically organized into hierarchical structures based on subsumption, partition or other properties. This structural context of the concept gives important information about its meaning and relations to other concepts. These functions can be fulfilled by such context visualizations as already described with reference to semantic autocompletion and depicted in figure 1.

2.4 Shared Concept and Instance Storage and Maintenance

In many semantic applications, there is a need for storing ontological metadata, be they the annotations of an indexer or links forged between ontologies by an ontology maintainer. However, existing systems may lack the means by which to store arbitrary semantic constructs or even just URIs. For example, a museum indexing system may contain databases for museum items and actors, but provide only a text field for storing locations. On the Semantic Web however, locations need to be stored as instances with properties of their own.

To address this need, we have developed a browser-based metadata editor SAHA, which can be used either as is or as depicted in figure 4, as a web widget in existing indexing systems lacking semantic capabilities [27]. Connecting SAHA to indexing systems can be done simply by linking to SAHA with a GET parameter specifying the identifier (URI) of the document being currently edited. By clicking on the link, a SAHA window opens, containing indexing

[11] http://www.yle.fi/elavaarkisto/
[12] http://www.google.com/adsense/

fields relevant for the current document. Afterwards, the annotations located in the indexing system and SAHA can be combined by their common document identifiers.

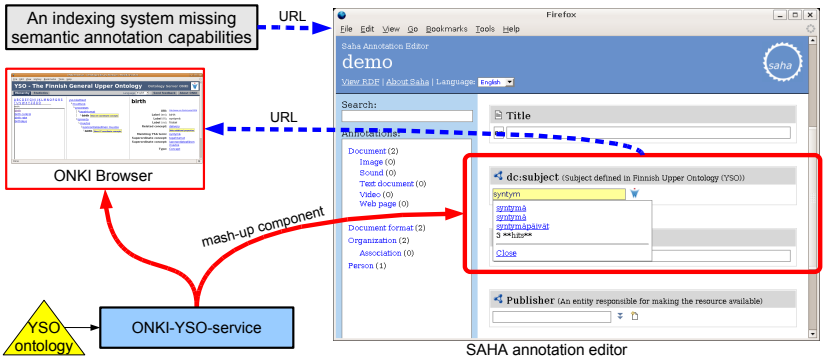


Fig. 4. The Finnish General Ontology connected to SAHA.

3 Case Applications

In the following, we present how we have combined and applied the previously described semantic web widgets in actual systems, both in content creation as well as end-user consumption.

3.1 The ONKI Server: Publishing Ontologies as Web Widgets

Currently, ontologies are typically shared by downloading them, and each application must separately implement the functionatily to support them. To avoid duplicated work and costs, and to ensure that the ontologies are always up-to-date, we argue that the ontologies should also be published as shared services. To demonstrate this aproach, we have implemented the ontology library service ONKI, which is used for maintaining, publishing and using ontologies [6].

As part of the ONKI concept, a user-interface web widget combining a semantic autocompletion search sub-component with concept fetching functionality has been implemented, which can be added to any application requiring access to a certain ontology. The widget looks like an ordinary text field, but when the user types in characters, matching concepts found from the ONKI server are listed. By selecting a concept from the result list, the concept's URI, label or other information is fetched to the client application for further processing and storage. In the context of a browser-based application, this fetching functionality has been implemented with JavaScript window referencing [28].

Another ONKI feature is the concept browser, which can be integrated to an application as an "ONKI button". When the button is pressed, a separate ONKI Browser window (see figure 5) opens, in which annotation concepts can be searched for and browsed, making use of semantic autocompletion, tree context visualization and concept property view components. For each concept entry, the browser shows a *Fetch concept* button which, when pressed, transfers the current concept information to the client application. For geographical data, a separate browser application, ONKI-Paikka[13] [29] has been created. This browser, shown in figure 6, has been implemented by combining ontological information with our geographical search and visualization widget.

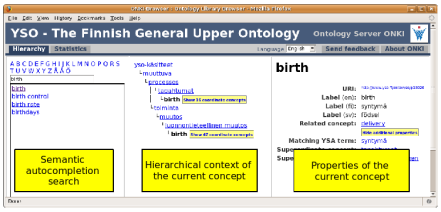


Fig. 5. The ONKI Ontology Browser's concept view.

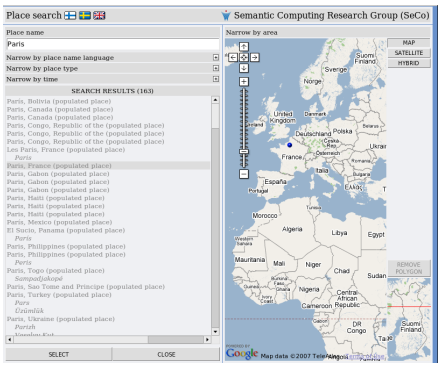


Fig. 6. The ONKI-Paikka browser.

Integrating these ONKI services to client applications only requires a minimal modification to the user interface implementation. For example, in the case of HTML pages and AJAX, only a short snippet of JavaScript code must be added to the web page for using the ONKI functionalities.

To test the ONKI solution, we have used the widgets in the browser-based annotation editor SAHA[14] [8, 27]. For example (figure 4), the Finnish General Ontology YSO [6] has been published as an ONKI service[15], and has been added as a web widget to SAHA for selecting annotation concepts. SAHA can also make use of our automatic text extraction component in extracting potential annotations from web resources.

In the case of the ONKI browsers, all concept and instance URIs are intended to be designed so that they function also as URLs. When the URI of a concept is accessed with a web browser, the relevant view is opened in the ONKI browser. This means that the URI itself acts as a functional link when added to a HTML

[13] http://www.seco.tkk.fi/services/onkipaikka/
[14] http://www.seco.tkk.fi/services/saha/
[15] http://www.yso.fi/onto/yso/

page. In accordance to W3C[16], if the URI is accessed with an RDF aware system, the machine readable RDF presentation of the content is returned instead of the ONKI browser's HTML presentation. This makes it easier to use ONKI services also in programmatic mash-up applications.

Compared to general ontology server interfaces such as the SKOS API[17], our approach is to publish highly specified functionalities as semantic web widgets that solve a specific user task, such as the need for concept search and fetch. In this, our approach complements the general APIs. The general APIs make it possible to create completely new functionality but require more programming work, while semantic web widgets make handling the most common tasks as cost-effective as possible.

3.2 CultureSampo: A Semantic Portal for Cultural Content

CultureSampo [30] is a semantic portal that gathers together a comprehensive collection of Finnish culture, including photographs, paintings, poems and biographies. Much of the functionality of the portal has been accomplished by combining the various components described above, as depicted in figure 7

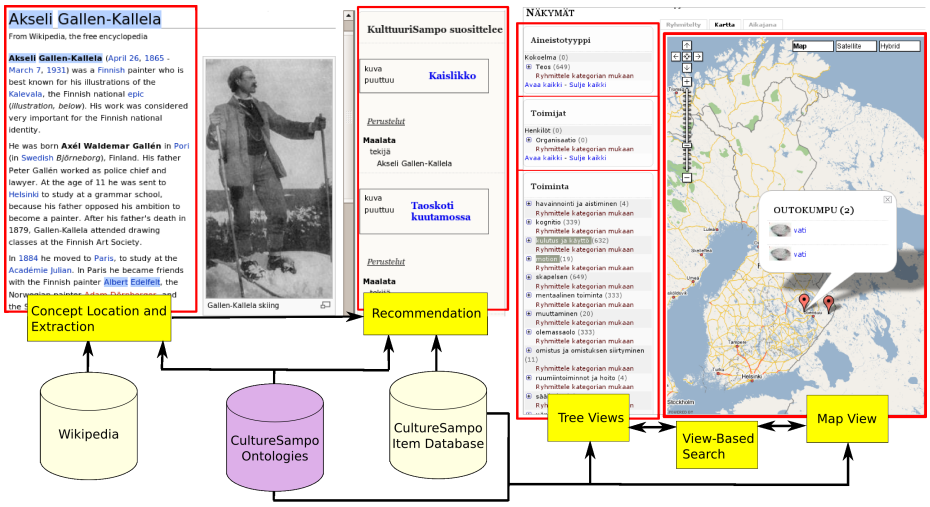


Fig. 7. The mash-up architecture of the CultureSampo portal

First, we have harnessed our automatic concept extraction component to enchance external web pages with CultureSampo content when viewed through the portal. For example, on the left in figure 7, a web page from Wikipedia[18] is

[16] http://www.w3.org/TR/swbp-vocab-pub/
[17] http://www.w3.org/2001/sw/Europe/reports/thes/skosapi.html
[18] http://www.wikipedia.org/

integrated into the portal. The person names highlighted in blue on the page are detected individuals, and their names are links to their biography in the portal. On the right of the page, other recommended items based on the content of the document are shown. These are calculated by feeding the concepts extracted from the page to our recommendation component. CultureSampo also provides multi-facet search functionality that utilizes our engine component combined with both tree and map-based search and visualization views. This functionality, along with a screenshot, is depicted on the right side in figure 7.

4 Discussion

4.1 Contributions

This paper presented the idea of using the mash-up approach for implementing semantic functionalities as web widgets which can easily be included in applications, such as adding concept search functionality to an indexing application.

A major benefit of the approach is that potentially highly complicated and expensive technical features and semantic data resources can be created once and published for others to use in a compact package, which can easily be integrated to an application. By making the adoption of semantic technologies as easy as possible, one may hope to further the adoption of the Semantic Web as a whole.

One of the benefits of publishing the widgets as centralized services is that updates in content and functionalities are instantenously available for the users. This is an especially important feature when the data evolves constantly, e.g. when user generated content is involved.

4.2 Related Work

Our own prior work on a semantic portal creation tool [31] already included a general semantic view-based search tool [17] and the semantic linking service Ontodella [24], as well as a framework for combining them into a complete portal. However, the user interface components were not yet modular, and neither were the search or recommendation functions used outside that environment.

On the other hand, many semantic web browsing and editing environments do provide general configurable visualization and selection widgets inside them, such as DBin [32], Piggy Bank [33], OntoWiki [34] and Haystack [35, 36]. These, however, are intended for use only inside the specific program environment, while our components are for general use.

Complementing our pursuits, there have recently been many announcements about mash-ups that make currently existing data available in RDF. DBpedia.org [37] has published Wikipedia material, the D2R server [38] has been used for publishing the DBLP article database[19], while the RDF Book Mashup[20] provides book information from Amazon and Google Base. From our viewpoint,

[19] http://www4.wiwiss.fu-berlin.de/dblp/
[20] http://sites.wiwiss.fu-berlin.de/suhl/bizer/bookmashup/

these mash-ups provide possible data sets to which our components could be hooked. They provide the data, we provide the functionality.

4.3 Future Work

Future directions for this research include looking for new general semantic web tasks that could be implemented using the web widget approach, especially in ontology development and maintenance. For example, supporting cross-link maintenance between ontologies, ontology change history maintenance and using ontology history knowledge in searches seem to be potential further research directions. The proposed semantic web widgets could also be developed further. For example, their functionalities could be provided via additional technologies alongside the current JavaScript and SOAP Web Service APIs.

Acknowledgements

This work is part of the National Semantic Web Ontology (FinnONTO) project[21], funded mainly by the National Funding Agency for Technology Innovation (Tekes). We wish to thank Ville Komulainen for his work on the ONKI Browser.

References

1. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American **284**(5) (May 2001) 34–43
2. Hyvönen, E., Mäkelä, E., Salminen, M., Valo, A., Viljanen, K., Saarela, S., Junnila, M., Kettula, S.: Museumfinland – finnish museums on the semantic web. Journal of Web Semantics **3**(2) (2005) 25
3. O'Reilly, T.: What is web 2.0 — design patterns and business models for the next generation of software. WWW article (September 30 2005) http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html.
4. Zhang, S.L.: Planning an authority control project at a medium-sized university library. College and Research Libraries **62**(5) (2001)
5. Niles, I., Pease, A.: Towards a standard upper ontology. In Welty, C., Smith, B., eds.: Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001). (October 17-19 2001)
6. Hyvönen, E., Viljanen, K., Mäkelä, E., Kauppinen, T., Ruotsalo, T., Valkeapää, O., Seppälä, K., Suominen, O., Alm, O., Lindroos, R., Känsälä, T., Henriksson, R., Frosterus, M., Tuominen, J., Sinkkilä, R., Kurki, J.: Elements of a national semantic web infrastructure - case study finland on the semantic web. (May 18 2007) Submitted for review.
7. Mäkelä, E.: View-based search interfaces for the semantic web. Master's thesis, University of Helsinki (June 2006)

[21] http://www.seco.tkk.fi/projects/finnonto/

8. Valkeapää, O., Alm, O., Hyvönen, E.: Efficient content creation on the semantic web using metadata schemas with domain ontology services (system description). In: Proceedings of the European Semantic Web Conference ESWC 2007, Innsbruck, Austria, Springer (June 4-5 2007)
9. Hyvönen, E., Mäkelä, E.: Semantic autocompletion. In: Proceedings of the first Asia Semantic Web Conference (ASWC 2006), Beijing, Springer-Verlag, New York (August 4-9 2006)
10. Kauppinen, T., Henriksson, R., Väätäinen, J., Deichstetter, C., Hyvönen, E.: Ontology-based modeling and visualization of cultural spatio-temporal knowledge. In: Developments in Artificial Intelligence and the Semantic Web - Proceedings of the 12th Finnish AI Conference STeP 2006. (October 26-27 2006)
11. Kauppinen, T., Deichstetter, C., Hyvönen, E.: Temp-o-map: Ontology-based search and visualization of spatio-temporal maps. Demo track at the European Semantic Web Conference ESWC 2007, Innsbruck, Austria (June 4-5 2007)
12. Ferraiolo, J., Jackson, D., Fujisawa, J.: Scalable Vector Graphics (SVG) 1.1 specification W3C recommendation. Technical report, World Wide Web Consortium W3C (January 14 2003)
13. Kauppinen, T., Hyvönen, E.: Modeling and reasoning about changes in ontology time series. In: Ontologies: A Handbook of Principles, Concepts and Applications in Information Systems (January 15 2007)
14. Hildebrand, M., van Ossenbruggen, J., Hardman, L.: /facet: A browser for heterogeneous semantic web repositories. In: The Semantic Web - Proceedings of the 5th International Semantic Web Conference2006. (November 5-9 2006) 272–285
15. Oren, E., Delbru, R., Decker, S.: Extending faceted navigation for rdf data. In: International Semantic Web Conference. (November 5-9 2006) 559–572
16. Mäkelä, E., Hyvönen, E., Sidoroff, T.: View-based user interfaces for information retrieval on the semantic web. In: Proceedings of the ISWC-2005 Workshop End User Semantic Web Interaction. (Nov 2005)
17. Mäkelä, E., Hyvönen, E., Saarela, S.: Ontogator — a semantic view-based search engine service for web applications. In: Proceedings of the 5th International Semantic Web Conference (ISWC 2006). (Nov 2006)
18. Känsälä, T., Hyvönen, E.: A semantic view-based portal utilizing Learning Object Metadata (August 2006) 1st Asian Semantic Web Conference (ASWC2006), Semantic Web Applications and Tools Workshop.
19. Mäkelä, E., Viljanen, K., Lindgren, P., Laukkanen, M., Hyvönen, E.: Semantic yellow page service discovery: The veturi portal. In: Poster paper, 4th International Semantic Web Conference. (Nov 2005)
20. Sidoroff, T., Hyvönen, E.: Semantic e-goverment portals - a case study. In: Proceedings of the ISWC-2005 Workshop Semantic Web Case Studies and Best Practices for eBusiness SWCASE05. (Nov 2005)
21. Dill, S., Eiron, N., Gibson, D., Gruhl, D., Guha, R.: Semtag and seeker: Bootstrapping the semantic web via automated semantic annotation. In: In Proceedings of the 12th International World Wide Web Conference, ACM Press (2003) 178–186
22. Kiryakov, A., Popov, B., Terziev, I., Manov, D., Ognyanoff, D.: Semantic annotation, indexing, and retrieval. Journal of Web Semantics **2**(1) (2004) 49–79
23. Dzbor, M., Domingue, J., Motta, E.: Magpie: towards a Semantic Web browser. Proceedings of the 2nd International Semantic Web Conference (2003)
24. Viljanen, K., Känsälä, T., Hyvönen, E., Mäkelä, E.: Ontodella - a projection and linking service for semantic web applications. In: Proceedings of the 17th International Conference on Database and Expert Systems Applications (DEXA 2006), Krakow, Poland, IEEE (September 4-8 2006) 370–376

25. Calí, A., Giacomo, G.D., Lenzerini, M.: Models for information integration: Turning local-as-view into global-as-view. In: Proc. of Int. Workshop on Foundations of Models for Information Integration (10th Workshop in the series Foundations of Models and Languages for Data and Objects). (2001)
26. Ruotsalo, T., Hyvönen, E.: A method for determining ontology-based semantic relevance. In: Proceedings of the 18th International Conference on Database and Expert Systems Applications. (September 3-7 2007)
27. Valkeapää, O., Hyvönen, E.: A browser-based tool for collaborative distributed annotation for the semantic web. (September 26 2006) 5th International Semantic Web Conference, Semantic Authoring and Annotation Workshop, November, 2006.
28. Komulainen, V.: Public services for ontology library systems. Master's thesis, University of Helsinki, Department of Computer Science (January 2007)
29. Lindroos, R., Kauppinen, T., Henriksson, R., Hyvönen, E.: Onki-paikka: An ontology service for geographical data (2007) Submitted for review.
30. Hyvönen, E., Ruotsalo, T., Häggström, T., Salminen, M., Junnila, M., Virkkilä, M., Haaramo, M., Mäkelä, E., Kauppinen, T., , Viljanen, K.: Culturesampo–finnish culture on the semantic web: The vision and first results. In: Developments in Artificial Intelligence and the Semantic Web - Proceedings of the 12th Finnish AI Conference STeP 2006. (October 26-27 2006)
31. Mäkelä, E., Hyvönen, E., Saarela, S., Viljanen, K.: Ontoviews – a tool for creating semantic web portals. In: Proceedings of the 3rd International Semantic Web Conference (ISWC 2004). (May 2004)
32. Tummarello, G., Morbidoni, C., Nucci, M., Panzarino, O.: Brainlets: "instant" semantic web applications. In Bizer, C., Auer, S., Miller, L., eds.: Proc. of 2nd Workshop on Scripting for the Semantic Web at ESWC, Budva, Montenegro. Volume 183 of CEUR Workshop Proceedings ISSN 1613-0073. (June 2006)
33. Huynh, D., Mazzocchi, S., Karger, D.: Piggy bank: Experience the semantic web inside your web browser. J. Web Sem. **5**(1) (2007) 16–27
34. Auer, S., Dietzold, S., Riechert, T.: OntoWiki - a tool for social, semantic collaboration. In Cruz, I.F., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L., eds.: International Semantic Web Conference. Volume 4273 of Lecture Notes in Computer Science., Springer (2006) 736–749
35. Karger, D.R., Bakshi, K., Huynh, D., Quan, D., Sinha, V.: Haystack: A general-purpose information management tool for end users based on semistructured data. In: Proceedings of the CIDR Conference. (2005) 13–26
36. Quan, D., Huynh, D., Karger, D.R.: Haystack: A platform for authoring end user semantic web applications. In: Proceedings of the Second International Semantic Web Conference. (2003) 738–753
37. Auer, S., Lehmann, J.: What have innsbruck and leipzig in common? extracting semantics from wiki content. In: Proceedings of the European Semantic Web Conference ESWC 2007, Innsbruck, Austria, Springer (June 4-5 2007)
38. Bizer, C., Cyganiak, R.: D2R server — publishing relational databases on the semantic web. Poster paper at the 5th International Semantic Web Conference (November 2006)

Semantic Enterprise Technologies

Massimo Ruffolo[1,2], Inderbir Sidhu[2], Luigi Guadagno[2]

[1] ICAR-CNR - Institute of High Performance Computing and Networking
of the Italian National Research Council
[2] fourthcodex inc.
University of Calabria, 87036 Arcavacata di Rende (CS), Italy
e-mail: ruffolo@icar.cnr.it
e-mail: {lguadagno,isidhu}@fourthcodex.com
WWW home page: http://www.fourthcodex.com

Abstract. Nowadays enterprises request information technologies that leverage structured and unstructured information for providing a single integrated view of business problems in order to foster better business process management and decision making. The growing interest in semantic technologies is due to the limitation of existing enterprise information technologies to answer these new challenging needs. Semantic Web Technologies (SWT), the current open standard approaches to semantic technologies based on semantic web languages, provide some interesting answers to novel enterprise needs by allowing to use domain knowledge within applications. However, SWT aren't well suited for enterprise domain because of some drawbacks and a lack of compatibility with enterprise-class applications. This paper presents the new Semantic Enterprise Technologies (SET) paradigm founded on the idea of *Semantic Models* that are executable, flexible and agile representation of domain knowledge. Semantic Models are expressed by means of the *Codex Language* obtained combining Disjunctive Logic Programming (Datalog plus disjunction) and Attribute Grammars both extended by object-oriented and two-dimensional capabilities. Semantic Models enable to exploit domain knowledge for managing both structured and unstructured information. Since the Codex Language derives from the database field, it allows SET to provide advanced semantic capabilities well suited for enterprises. Differences and interoperability issue between SET and SWT are briefly discussed in the paper that shows, also the SET Reference Architecture (SETA), an application example and the business value of SET.

1 Introduction

Nowadays enterprise knowledge workers need information technologies that leverage structured and unstructured information for providing a single integrated view of business problems in order to foster better business process management and decision making. They want answers to specific business requirements, not documents and reports, search document base by concepts not simply by keywords, query databases and unstructured information repositories in a uniform way taking into account the meaning of data and information, obtain application integration and interoperability exploiting semantic-aware services.

The growing area of Semantic Technologies [2] could answer these novel needs by providing a new enterprise-class of semantically-enabled business applications. In this scenario semantics means the use of domain knowledge to affect computing by allowing the meaning of associations among information to be known and processed at execution time. Semantic Technologies must be able to:

- Allow computable semantic representations of domain knowledge, and provide reasoning capabilities on it, in order to enable software to do useful tricks such as finding hidden relationships in a complicated web of objects.
- Handle very large-scale knowledge bases containing both structured and unstructured information.
- Automate the capturing of events of events and entities to connect people, places, and events using information in different formats coming from many different sources.
- Assist human monitoring and analysis of situations, workflows, collaboration and communication.
- Facilitate interoperability by exploiting enterprise concepts to link applications, data sources, and services in easily to use composite views, providing real-time interaction, analysis, and decision-support.
- Deliver their capabilities as value added ingredient components that are easy to embed with existing enterprise applications and integration architectures.

These challenges exceed the capabilities and performance capacity of current open standards approaches to semantic technologies based on semantic web languages (i.e. OWL and RDF) [16]. Such approaches have the benefit to create interoperability over the web but they suffer of the following important drawbacks when applied to enterprise domains: (i) they have a lack of compatibility with relational databases, the most widely adopted enterprise information technologies for representing, storing and managing enterprise structured data; (ii) they allow for "connecting the ontological dots" by using predefined ontological-data model that assists to discover and infer new knowledge (when proper reasoners are available). But 'real world' enterprise business practices are not neatly predefined and in fact are more likely to be dynamic, emergent, or even chaotic. In other words, fully articulated knowledge models (ontologies) are not necessary for recognizing relevant facts in the ever-growing knowledge bases, or inferring new useful related information, or ensuring that enough knowledge is available just in time to improve decision-making; (iii) they do not propose mechanisms for handling directly the already available huge amount of unstructured information.

This paper describes the novel *Semantic Enterprise Technologies* (SET) paradigm founded on a the idea of *Semantic Models* (SM) that are executable, flexible and agile representation of domain knowledge (e.g. simple taxonomies equipped with few and simple descriptors, very rich ontologies equipped with complex business rules and descriptors). SM are represented by means of a new language, called *Codex Language* obtained by combining Disjunctive Logic Programming (Datalog plus disjunction) [4,6,9] and attributes grammars both extended by means of object-oriented [3,12] and two-dimensional capabilities [13,14,15]. SM enable to exploit domain knowledge for managing both structured (e.g. relational databases) and unstructured information (e.g. document repositories).

SET overcome the above mentioned SWT limitations by allowing the following fundamental set of features: (i) they basically came from the database world. So they propose query language and reasoning approaches well suited for the relation model. The Codex language in fact, are based on the Closed World Assumption (CWA) and the Unique Name Assumption (UNA) whereas semantic web languages are based on Open World Assumption (OWA) and do not consider UNA. In order to make SET interoperable with SWT a translation approach that takes into account the different semantics of the Codex Language and OWL has been defined. This way SET can be considered complementary to SWT, (ii) the Codex Language allows to represent *Semantic Models* that can be composed of "just enough" taxonomies and rules or of, if required, complex ontologies containing also relationships, constraints, axioms, so the knowledge representation process fits the very dynamic enterprise environments; (iii) they provide powerful unstructured information management mechanisms that allow concepts annotation and extraction from unstructured sources (semantic enterprise metadata acquisition) via a pattern-based approach. So precise semantic information extraction, classification and search, enabling semantic indexing and querying of the enterprise knowledge, are also possible.

In order to make effective technological and business advantages of SET, as already happens for existing enterprise-class information technologies, a reference architecture that constitutes the framework for applying SET features to enterprise domains is required. This paper proposes SETA (the SET reference Architecture) that describes the technologies and architecture enabling the use of Semantics in an enterprise. SETA aims at transforming multiple sources (structured and unstructured) and bits of dynamic information with domain and concept coverage from disparate enterprise systems into useful knowledge that fosters better enterprise performances.

SET have already been applied to contact-center software, CRM applications, asset and content management repositories, news and media delivery services, health care organizations and more. Their distinctive features help to shape the future of new knowledge-powered computing solutions in many different traditional areas like Competitive Intelligence, Document and Content Management, CRM, Text Analytics, Information Extraction. The application of SETs to real cases shows that they can improve value creation capabilities of enterprises allowing the definition and execution of more efficient and effective business processes and decision making.

The remainder of this paper is organized has follows. Section 2 presents the structure of Semantic Models and describes the Codex Language, Section 3 provides a comparison between Codex Language and OWL and drafts the interoperability approach, Section 4 describes the SET reference Architecture, Section 5 sketches an application of SET to health care risk management, Section 6 contains a brief description of the business value of SET and Section 7 concludes the paper.

2 Semantic Models

SET are based on the concept of Semantic Model. A SM is a flexible and agile representation of domain knowledge. Semantic Models can be constituted by either just small pieces of a domain knowledge (e.g. small taxonomies equipped with few

rules) or rich and complex ontologies (obtained, for example, by translating existing ontologies and by adding rules and descriptors) that gives respectively weak or rich and detailed representation of a domain. More formally a SM is a seven-tuple of the form:

$$SM = \langle \mathcal{H}_C, \mathcal{H}_R, \mathcal{O}, \mathcal{T}, \mathcal{A}, \mathcal{M}, \mathcal{D} \rangle$$

where:

- $\mathcal{H}_C$ and $\mathcal{H}_R$ are sets of classes and relations *schemas*. Each schema is constituted by a set of attributes, the type of each attribute is a class. In both $\mathcal{H}_C$ and $\mathcal{H}_R$ are defined partial orders allowing the representation of concepts and relation taxonomies (with multiple inheritance).
- O and T are sets of class and relation *instances* also called *objects* and *tuples*.
- $\mathcal{A}$ is a set of axioms represented by special rules expressing constraints (rules always true) about the represented knowledge.
- $\mathcal{M}$ is a set of *reasoning modules* that are logic programs constituted by a set of (disjunctive) rules that allows to reason about the represented and stored knowledge, so new knowledge not explicitly declared can be inferred.
- $\mathcal{D}$ is a set of *descriptors* (i.e. production rules in an two-dimensional object-oriented attribute grammar) enabling the recognition (within unstructured documents) of class (concept) instances contained in C, so their annotation, extraction and storing is possible.

SM are represented by means of the novel powerful and very expressive *Codex Language* described in the following.

2.1 The Codex Language

The Codex Language brings together the expressiveness of ontology languages and the power of Disjunctive logic rules. The Codex Language combines notions coming from Disjunctive Logic Programming (Datalog plus disjunction) and Attribute Grammars both extended by means of object-oriented and two-dimensional capabilities. The attribute grammars allow one to intuitively express patterns for recognizing instances of the ontology concepts in structured and unstructured data. The language has been defined as such in recognition of the fact that in order to leverage and apply Semantic Models in the enterprise, it is important to find and retrieve appropriate data from all kinds of data sources, such as, schema based structured databases, unstructured documents and semi-structured documents containing implicit structure.

In order to understand the motivation for extending the capabilities of the Codex Language beyond the modeling capabilities of most ontology languages to a language that also makes it possible to describe how to recognize instances of the ontology concepts in data, consider the following example. Imagine a financial analyst tasked with researching some corporation, say, Microsoft. An ontology could describe that a company may be owned by individuals or boards, and the knowledge base could contain the facts that Microsoft is a company, and it is owned by Bill Gates. Besides representing all of the above information, the codex language can also express the rules and patterns for identifying instances of Microsoft Corp. in documents. One possibility is for a document to mention Microsoft as “the company owned by Bill

Gates". The rules for recognizing the concept of company ownership can easily be expressed in the codex language allowing for the identification of the instances of companies owned by Bill Gates or anyone else in structured and unstructured data sets. Hence, it is now possible to determine that when a document mentions "the company owned by Bill Gates", we have really discovered a document talking about Microsoft Corp.

The Codex language supports the typical ontology constructs, such as, *class*, *object* (class instance), *object-identity*, (*multiple*) *inheritence* and *relation*s, *tuple* (relation instance). It also supports powerful reasoning by means of *reasoning modules* that are modular logic programs containing a set of (disjunctive) rules. The language augments these typical ontology modeling constructs with a mechanism that enables the description of patterns and rules over an ontology for identifying meaningful data in any data source. This part of the codex language extends classical attribute grammars by means of two-dimensional and object-oriented capabilities allowing the expression of concept *descriptors*. A descriptor represents a rule that "describes" the means for recognizing instances (objects) of a concept in unstructured documents by means of complex (two-dimensional) composition of other objects or in structured sources (e.g. databases, structured files) by means of ad-hoc queries and reasoning tasks. When a descriptor matches within an unstructured document, the document can be annotated with respect to the related concept, and moreover, an instance of the matching class can be created in the knowledge base. In order to empower unstructured information management, the Codex language can also exploit sophisticated Natural Language Processing capabilities.

In the Codex Language a *class* can be thought of as an aggregation of individuals (objects or class instances) that have the same set of properties (attributes). A class is defined by a name (which is unique) and an ordered list of attributes identifying the properties of its instances. Each attribute is identified by a name and has a type specified as a built-in or user-defined class. For instance, the classes `country` and `person` can be declared as follows:

```
class country (name:string).
class person (name:string, age:integer, nationality:country).
```

The ability to specify user-defined classes as attribute types (`nationality:country`) allows for the description of complex objects, i.e. objects made of other objects recursively (a person could be a parent that is also a person). The language also supports the definition of special classes called *collection classes* that "collect" individuals that belong together because they share some properties. Instances of these classes can either be declared explicitly as in the case of normal classes, or specified by a rule that defines the shared properties in an intensional way.

Objects (*class instances*) are declared by asserting new facts. Objects are unambiguously identified by their object-identifier (*oid*) and belong to a class. An instance for the class `manager` can be declared as follows:

```
bill:manager("Bill", 35, usa, 50000).
mario:manager("Mario", 37, italy, 40000).
```

Here, the strings "Bill" and "Mario" are the values for the attribute *name*; while `bill` and `mario` are the *object-identifier[s] (oid)* of these instances (each instance is identified by a unique oid). Instance arguments can be specified either by object

identifiers (`usa` and `italy`), or by a nested class predicate (complex term) which works like a function.

Relationships among objects are represented by means of *relations*, which, like classes, are defined by a unique name and an ordered list of attributes (with name and type). Relation instances (*tuples*) are specified by asserting a set of facts. For instance, the relation `managed_by`, and a tuple asserting that project `newgen` is managed by `bill` (note that `newgen` and `bill` are OID), can be declared as follows:

```
relation managed_by(proj:project, man:manager).
    managed_by(newgen, bill).
```

The Codex language makes it possible to specify complex rules and constraints over the ontology constructs, merging, in a simple and natural way, the declarative style of logic programming with the navigational style of ontologies. Additionally, the rules and constraints are organized as *reasoning modules*, benefiting from the advantages of modular programming. Eventually, in order to check the consistency of a knowledge base the user can specify global integrity constraints called *axioms*. For example, the following axiom expresses that each project can have only one manager:

```
::- managed_by(proj:A, man:M1), managed_by(proj:A, man:M2), M1 <> M2.
```

A descriptor can be viewed as an object-oriented production rule $p \in \Pi$ in an two-dimensional attribute grammar defined on a formal context free grammar $\mathcal{G} = \langle \Sigma, V_N, A \in V_N, \Pi \rangle$ over the alphabet Σ. In the Codex Language the domain of the attributes is the set of classes declared in the Semantic Model whereas the alphabet Σ is constituted by class names and object identifiers. More formally, a *descriptor* d is a couple $\langle h, b \rangle$ such that $h \rightarrow b$, where h is the *head* of d and b is its *body*. The following example declares the `company` class, along with an instance and a descriptor for this instance:

```
class company (name:string, nationality:country, market:market_area).
    acme: company("Acme Inc.", usa, rocket_skiis).
    <acme> -> <X: hiStr(), matches( X, "[Aa](?:acme|ACME)")>.
```

The descriptor head `<acme>` represents the object (or the class of objects) that the user desires to recognize within unstructured documents. The descriptor body represents the rule "describing" the structure of the objects in the head in terms of regular expressions (or other objects).

As another example, consider the following fragment of the Codex Language representing the extraction of a table containing stock index:

```
collection class italian_stock_market_index_row(
    stockMarketIndex: stock_market_index, [value]){}
        <italian_stock_market_index_row(stockMarketIndex:S, L)> ->
            <X: stock_market_index(name_index:N, italy)> <X: value(V)>{L:=add(L,V)}+.
collection class italian_stock_market_index_table
    ([italian_stock_market_index_row]){}
        <table(L)> -> <X: italian_stock_market_index_row()> {L:=add(L,X)}+ DIRECTION = "vertical".
```

In the example, the collection class `italian_stock_market_index_row` represents table rows composed of the `stock_market_index` and a sequence of numeric `value[s]`; the collection class `italian_stock_market_index_table` will contains the vertical sequence of rows that constitute the table.

The Codex language allows the expression of very complex patterns that utilize the ability to treat any unstructured or semi-structured document as a two-dimensional plane and exploit the full expressive power of semantic models. The above example

in particular shows the ability to focus on complex and very specific information for extraction from unstructured documents, which in this case happens to be a table of stock indexes related to Italian companies only.

It is noteworthy that *descriptor*[s] can be expressed using a visual support and that the instances of concepts matching the descriptors can be extracted and stored in the Knowledge Base. These instances can also be serialized as XML, RDF, etc. to be used for analytical and/or Web (Semantic Web) based applications.

3 The Semantic Enterprise and the Semantic Web

In order to motivate why Semantic Enterprise Technologies paradigm could represent an opportunity for improving current semantic technology capabilities, it is important to compare and contrast the semantic enterprise approach against the semantic web approach. For example, in the closed world of an enterprise, reasoning using closed world and unique name assumptions offers certain advantages over the open world assumption necessary when working with the Web. Another important aspect for consideration is the need for performing semantic reasoning over enterprise data residing in relational databases. It is also important to be able to integrate rules with ontologies in order to maximize the return from the ontology building effort. The benefits possible from the integration of logic programming and OWL have already been described in [1,8,11]. In this section we provide a detailed description of the advantages of the SET paradigm and its interoperability with semantic web technologies.

3.1 Closed World and Unique Name Assumptions

Enterprise databases are founded on Unique Name Assumption (UNA) and Closed World Assumption (CWA). The semantics of these databases are intuitive and familiar to their users. The CWA is known only information explicitly stored in the Knowledge Base, so a CWA-based rules entails false for information not explicitly declared in the Knowledge Base. The UNA assumes that names of information elements stored in the Knowledge Base are unique, so two elements with different names are necessarily distinct. The codex language is based on closed world and unique name assumption. This makes it highly suitable for modeling and reasoning in the enterprise. Conversely, OWL is based on Open World Assumption which is better suited for the Semantic Web.

A Semantic Model can be seen as an extension of the enterprise database. In order to understand the effect the semantic assumptions of a database can have on the behavior of a query over exactly the same data, consider the databases shown in **Fig. 1.** When database (A) is queried for the company that has its headquarters only in the USA, a CWA-based reasoner returns `Intel`, whereas an OWA-based reasoner is unable to answer this query because there is no explicit statement stating that Intel has its headquarters **only** in the US. Database (B) represents the relationship that for each project we can have at most one leader and

that a project leader can lead many projects. The DL representation is the following TBox: `PROJECT MANAGER hasLeader: PROJECT --> MANAGER` and ABox: `PROJECT(p1) PROJECT(p2) MANAGER(John) MANAGER(Chris) hasLeader(p1,John) hasLeader(p2,Chris)`. In order to represent the database constraint that a project may have at most one manager, we impose the following DL constraint: $\top \leq 1$ `hasLeader`. In the case of a relational database, when a user attempts to insert the fact that John is also a leader of the project `p2`, there is a violation of referential integrity, and the action is not allowed by the database. However, if we assert the same fact in a DL ABox as `hasleader(p2,john)`, the system does not complain about any possible violation. When the system is queried for the leader of the project `p2`, the OWA-based reasoner infers that `Chris` and `John` are the project leaders and that `Chris=John` because of the lack of UNA. In order to obtain the same behavior as CWA in this case, the user must explicitly declare that `Chris` $\neq$ `John`. This is obviously counterintuitive for the normal enterprise user.

A)

COMPANY	HEADQUARTERS	CITY ADDRESS
Microsoft	USA	Seattle
FourthCodex	USA	Chicago
Microsoft	Ireland	Dublin
FourthCodex	Italy	Cosenza
Intel	USA	Saint Clara

B)

PROJECT		
ID	NAME	FK_MAN
1	P1	1
2	P2	2

1:1 hasLeader 0:n

MANAGER	
ID	NAME
1	Chris
2	John

Fig. 1. Two simple relational databases

Negative Queries Negative queries are queries for data where the query condition **does not** hold. For example, an airline database may contain all the pairs of airports connected by its flights. The CEO of the airline might want to query for all the airports not connected by its hub airport. An OWA based reasoner would have trouble answering such a query, since the database does not, and for all practical purposes, can not, contain information about every airport that is not serviced. In the absence of asserted or explicitly derived negative information, the OWL based reasoner cannot answer such query. On the other hand, answering such a query is trivial for CWA based systems which correctly assume that if there doesn't exist any record of a flight between two airports, then it is save to assume that there are not flights between those airports. Answering queries containing negative criteria in an intuitive way usually requires some form of closed-world reasoning.

From the above discussions and examples it follows that CWA and UNA are better suited for the enterprise domain because their semantics are more intuitive for the users. Also, the reasoning based on such assumptions produces more useful new information. So an approach to knowledge representation coming directly from the database world is able to preserve the database semantics allowing the user to query and reason on databases in a more natural way.

3.2 Rules and Integrity Constraints

Rules It is important to integrate rules with ontologies in order to fully capture the knowledge of an enterprise or a domain. Enterprises need to realize a return on their

investment in building ontologies, and a such, ontologies must not be perceived as just a documentation technique for describing a domain, but also as a language and system that can capture and execute domain rules expressed over the ontology. This has also been recognized by the Semantic Web community which is actively working on adding rules to the Semantic Web language stack [7]. The Rule Interchange Format (RIF) working group of the World Wide Web Consortium (W3C) is currently working on standardizing such a language. Responding to popular demand, the Semantic Web Rule Language (SWRL) has been proposed. However, as the authors point out, SWRL has been designed as a first-order language and straightforwardly integrated with OWL as a simple extension. For these reasons SWRL is trivially undecidable and does not address nonmonotonic reasoning tasks, such as expressing integrity constraints. The codex language on the other hand has built-in support for rules. These rules can operate on the concepts, relationships and constraints defined in the ontology, as well as any other arbitrary atoms and predicates desired by the user. The logical underpinning for codex is provided by Datalog, providing closed world semantics such as default negation and unique name assumption as well as support for recursive queries. As consequence of being Datalog based, codex also supports all of the semantics of **SQL** easily extending the benefits of semantic technologies to enterprise databases.

Integrity Constraints In OWL, domain and range restrictions constrain the type of objects that can be related by a role. Also, participation restrictions specify that certain objects have relationships to other objects.

For example, we can state that each student *must* have a student number as: $Student \subseteq \exists hasStudentNumber.StudentNumber$ [10]. However, even though this restriction can be expressed, its semantics are quite different from those of an equivalent constraint in a relational database. A relational database will not allow the user to insert a student without specifying a student number. Because of its open world assumption, OWL will allow a Student without a student number, since the assumption will be that the student has a student number, but it hasn't yet been added to the ABox. Straightforward specification and enforcement of integrity constraints is extremely important for enterprises as such constraints are an essential part of their domain and business model. In the codex language it is trivial to specify and apply such constraints.

3.3 Disjunctive Reasoning

The Codex Language is developed on top of the DLV system [4,9] that allows to exploit the answer set semantics and stable model semantics [6] for disjunctive logic programs. The possibility to exploit disjunction (in the head of rules) and constraints enable to express reasoning tasks for solving complex real problems. The disjunction allows to compute a set of models (search space) that can contain the possible solution for a given problem, whereas constraints allow to choice the solution by adopting a brave or cautious reasoning approach. In the brave reasoning are considered the solutions that are true in at least one model, in the cautious reasoning a solution to be considered acceptable must be true in all the models. Disjunction allows to express very rich business rules and to model reasoning task able to solve different kinds of

problems like: planning problems (under incomplete knowledge), constraints satisfiability, abductive diagnosis. In the following, in order to better explain disjunctive capabilities of the Codex Language an example of team building is provided.

```
module(teamBuilding){
     (r) inTeam(E, P) ∨ outTeam(E, P) :- E:employee(),P:project().
     (c1) :- P:project(numEmp:N),not #count{E:inTeam(E, P)}=N.
     (c2) :- P:project(numSk:S),not #count{Sk:E:employee(skill:Sk), inTeam(E, P)}  ≥ S.
     (c3) :- P:project(budget:B),not #sum{Sa,E:E:employee(salary:Sa), inTeam(E, P)} ≤ B.
     (c4) :- P:project(maxSal:M),not #max{Sa:E:employee(salary:Sa), inTeam(E, P)} ≤ M.  }
```

The reasoning module contain a disjunctive rule `r` that guesses whether an employee is included in the team or not, generating the search space by exploiting the answer set semantics. The constraints `c1`, `c2`, `c3`, and `c4` model the project requirements, filtering out those solutions that do not satisfy the constraints. So knowledge encoded into the Semantic Model can be exploited for providing solutions to complex business problems.

3.4 Interoperability With the Semantic Web

Currently OWL is the standard language on which the Semantic Web movements is trying to really implement it [16]. A lot of dictionaries, thesaurus and ontologies, expressed by means of this language, are already available. Many companies and organizations have invested in the construction of semantic resource like enterprise models and domain ontologies that they want use for building semantic applications. So all the organizations deal with the problem to reuse these semantic resources, developed by means of OWL, and to make them interoperable with already existing databases and application.

The SET paradigm address the interoperability problem with SWT providing an import-export approach that enable to "translate" OWL ontologies in Semantic Models (without descriptors) and viceversa. When a Semantic Model is obtained from an already existing OWL ontologies descriptors can be added in order to enable SET to exploit the model for semantic applications.

To make OWL and the Codex Language interoperable is a complex problem because it requires the translation from OWL (based on description logic) to the Codex Language (based on Disjunctive Logic Programming). Problems related to the joining of DL and DLP has been addressed by many authors [5,8,10]. They presented many methods to translate OWL-DL to logic programming. All these methods require first the definition of which fragment of OWL to deal with. For example, In [10] a detailed description of how a considerable fragment of OWL-DL can be processed within logic programming systems. To this end, the author derives an enhanced characterization of Horn-SHIQ, the description logic for which this translation is possible, and explained how the generated Datalog programs can be used in a standard logic programming paradigm without sacrificing soundness or completeness of reasoning. For the translation from OWL to Codex Language some fragments of the languages for which the semantic equivalence between the original OWL ontology and the obtained Semantic Model is guaranteed have been identified. In this context semantic equivalence means the possibility to obtain the same entailment in the source and in the destination language despite the differences of semantic assumptions. A more detailed discussion

around OWL-Codex Language translation is out of the scope of this paper, for further details is possible to consult [3].

4 SETA: The SET reference Architecture

The SET reference Architecture describes the technologies and architecture enabling the use of Semantics in an enterprise. The various entities for enabling semantic applications in an enterprise are shown in Fig. 2. below.

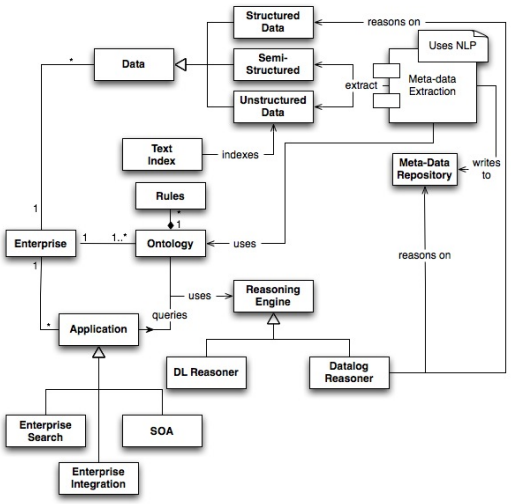


Fig. 2. Semantic Enterprise

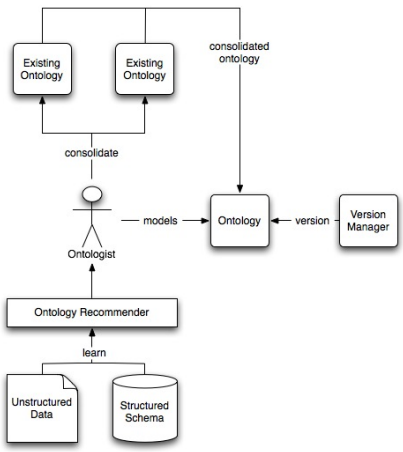


Fig. 3. Ontology Management

In this figure we identify the high level components composing the Semantic Enterprise. These include tools and technologies for:

- modeling and managing Semantic Models
- specifying rules over a Semantic Model
- extracting metadata from unstructured and semi-structured sources
- storing and indexing the extracted metadata
- reasoning over extracted metadata and existing structured data

The modeling environment usually includes a graphical interface (GUI) for creating, modifying and managing Semantic Models. This interface also allows one to specify rules over the Semantic Model. These rules may include the concept descriptors and the reasoning engine uses the constraints and relationships in conjunction with these rules for performing its reasoning tasks. It is vital to include both structured and unstructured data in order to enable the semantic enterprise. This requires two important features from the architecture: (i) use of existing structured data sources, (ii) extraction of useful information from unstructured data sources. The technologies comprising the semantic architecture should be able to use the existing enterprise databases for providing semantic capabilities since the enterprises have large amounts of data and hence it is not feasible to convert all this data into a different format for the purpose of enabling semantic enterprise. So far all of the technologies in the

enterprise have only been able to use structured data for providing decision making capabilities and enabling various applications. With the availability of ontologies and reasoning capabilities it is now possible to leverage unstructured and semi-structured data, since the semantics allow us to assign the correct interpretation. It is important to remember that a large amount of information is locked up in such unstructured sources, and it is important to include it in any decision making process and applications. Extracting useful information from unstructured data requires a multi-pronged approach. This kind of approach includes:

- Natural Language Processing (NLP) including Part of Speech (POS) tagging, sentence splitting, entity extraction, and other NLP related capabilities.
- Concept recognition and extraction.
- Supervised and unsupervised classification.

Supporting the Semantic Enterprise is a well defined process that requires the management of Semantic Models. This goes beyond ontology modeling, and includes the entire ontology life-cycle management comprising of ontology versioning, combination of ontologies into a higher level ontology, ontology comparison, and the addition of rules and descriptors to ontologies as shown in Fig. 3. above.

5 An e-Health Application

In this section an application of SET to a real case is shown. The scope of the application is to support wards to monitor errors and risks causes in lung cancer cares. The application has been developed in the context of a project aimed at provide some Italian hospitals with health care risk management capabilities.

The application takes as input Electronic Medical Records (EMRs) and risk reports coming from different hospital wards. An EMR is generally a flat text document (having usually 3 pages) written in Italian natural language. EMRs are weakly structured, for example, the personal data of the patient are in the top of the document, clinical events (e.g medical exams, surgical operations, diagnosis, etc.) are introduced by a date. Risk reports, filled at the end of clinical process, are provided to patients by wards to acquire information about errors with or without serious outcomes, adverse events, near misses.

The goal of the application is to extract semantic metadata about oncology therapies and errors with temporal data. The application extracts personal information (name, age, address), diagnosis data (diagnosis time, kind of tumor, body part affected by the cancer, cancer progression level), care and therapies information. Extracted information are exploited to construct, for each cared patient, an instance of lung cancer clinical process. Acquired process instances are analyzed by means of data and process mining techniques in order to discover if errors happen following patterns in phases of drugs prescription, preparation or administration.

The application has been obtained by representing a medical Semantic Model inherent to lung cancer that contains: (i) concepts and relationships referred to the disease, its diagnosis, cares in term of surgical operations and chemotherapies with the associated side effects. Concepts related to persons (patients), body parts and risk causes are also represented. All the concepts related to the cancer come from the

ICD9-CM diseases classification system, whereas the chemotherapy drugs taxonomy, is inspired at the Anatomic Therapeutic Chemical (ATC) classification system. (ii) a set of descriptors enabling the automatic acquisition of the above mentioned concepts from Electronic Medical Records (EMRs). In the following a piece of the medical Semantic Model that describes (and allows to extract) patient name, surname, age and disease is shown.

```
class anatomy ().
    class body_part (bp:string) isa {anatomy}.
        class organ isa {body_part}.
            lung: organ("Lung").
            <lung>-><X:histr(), matches(X,"[Ll]ung")>.
        ...
    ...
class disease (name:string).
    tumor: disease("Tumor").
    <tumor>-><X:histr(), matches(X,"[Tt]umor")>.
    cancer: disease("Cancer").
    <cancer>-><X:histr(), matches(X,"[Cc]ancer")>.
    ...
relation synonym (d1:disease,d2:disease)
    synonym(cancer,tumor).
    ...
class body_part_desease () isa {disease}.
    lung_cancer: body_part_disease("Lung cancer").
    <lung_cancer>-><diagnosis_section> CONTAIN <lung> & <X:desease(),synonym(cancer,X)>
    ...
collection class patient_data (){}
    collection class patient_name (name:string){}
        <patient_name(Y)> -> <X:hiStr(),matches(X,"name:")> <X:hiToken()> {Y := X;}
            SEPBY <X:space()>.
    collection class patient_surname (surname:string){}
            <patient_surname(Y)>->
                <X:hiStr(),matches(X,"sur(:?name)?:")> <X:hiToken()> {Y:=X;} SEPBY <X:space()>.
        collection class patient_age (age:integer){}
            <patient_age(Y)>-><X:hiStr(),matches(X,"age:")> <Z:hiToken()>{Y := $str2int(Z);}
                SEPBY <X:space()>.
    ...
collection class patient_data (name:string, surname:string,
                               age:integer, diagnosis:body_part_disease){}
        <patient_data(X,Y,Z,lung_cancer)> ->
            <hospitalization_section> CONTAIN <P:patient_name(X1)>{X:=X1}
            & <P:patient_surname(Y1)>{Y:=Y1} & <P:patient_age(Z1)>{Z:=Z1} & <lung_cancer>.
...
```

The classes `diagnosis_section` and `hospitalization_section` used in the above descriptors represent text paragraphs containing personal data and diagnosis data recognized by proper descriptors that aren't shown for lack of space. The extraction mechanism can be considered in a WOXM fashion: Write Once eXtract Many, in fact the same descriptors can be used to enable the extraction of metadata related to patient affected by `lung_cancer` in unstructured EMRs that have different arrangement. Moreover, descriptors are obtained by automatic writing methods (as happens, for example, for the cancer and tumor concepts) or by visual composition (as happens for `patient_data`)

Metadata extracted by using the Semantic Model are stored as collection class instances into a knowledge base. For the simple piece of Semantic Model shown above the extraction process generates the following `patient_data` class instance for an EMR: `"@1": patient_data("Mario","Rossi","70",lung_cancer)`.

The application is able to process many EMRs and risk reports in a single execution and to store extracted metadata in XML format.

6 The Business Value of Semantic Enterprise Technologies

The SET paradigm allows value creation from different perspectives. From the technological point of view, the value creation capabilities of the SET paradigm can be explained by introducing the *knowledge powered computing* vision. This vision is founded on the transformation of enterprise information into knowledge by converting knowledge into software via a portfolio of embeddable semantic components. Semantic Models transform information into knowledge and can be leveraged to directly embed knowledge into software making it actionable. This way Semantic-aware enterprise applications can be obtained. In fact, to provide value SET cannot be separate, external, or isolated, rather they have to interoperate with the complex portfolios of applications and information repositories, owned by enterprises to enhance their performances.

From the strategic point of view, turning knowledge into software enable to deliver a new generation of *knowledge powered features* that allow to better assist in driving business processes and in making decisions. A new categories of knowledge worker can leverage such enhanced features to obtain functionalities and domain knowledge interchanges that creates information intelligence where knowledge powered applications deliver more precise answers with adaptive responses. So better performances can be achieved by improving decision support and making, exception handling, emergency response, compliance, risk management, situation assessment, command and control.

From the tactical point of view, is important to note that SET features enable a new way to use data and information. SET allow to leverage new information resources like e-mail, web pages, forums, blogs, wikis, CRM transcripts, search logs, organizational documents as already happens for database. To exploit executable Semantic Models enables a better understanding of the interrelationships and shared context of existing structured and unstructured enterprise information. So more informed users can work smarter with better business process execution and monitoring, decision-making and planning.

7 Conclusion

This paper presented the Semantic Enterprise Technologies (SET) paradigm. SET are based on the concept of Semantic Model that are executable, flexible and agile representation of domain knowledge (e.g. simple taxonomies equipped with few and simple descriptors, very rich ontologies equipped with complex business rules and descriptors) and to exploit it for managing both structured (e.g. relation databases) and unstructured information (e.g. document repositories). Semantic Models are expressed by means of the *Codex Language* obtained combining Disjunctive Logic Programming (Datalog plus disjunction) and Attribute Grammars both extended by object-oriented and two-dimensional capabilities. SET overcome the limitation of Semantic Web Technologies (SWT) in enterprise domains. SET provide mechanisms to address several important modeling problems that frequently happens in enterprises and that are hard, if not impossible to solve using OWL alone, but can easily be addressed using the Codex Language. SET are interoperable with SWT thank to a translation

mechanism. Translation allows to import portions of already existing OWL ontologies to use in semantic enterprise applications and to export portions of Semantic Models toward semantic Web applications. Leveraging enhanced semantic features of SET, enterprises can transform information into knowledge in order to achieve better business performances.

References

1. Baader F., Calvanese D., McGuinness D.L., Nardi D., Patel-Schneider P.F., eds. The Description Logic Handbook: Theory, Implementation, and Applications. CUP - 2003
2. Davies J., Studer R., Warren P. Semantic Web Technologies: Trends and Research in Ontology-based Systems. Wiley, July 11, 2006, ISBN-13: 978-0470025963.
3. Dell'Armi T., Gallucci L., Leone N., Ricca F., Schindlauer R. "OntoDLV: an ASP-based System for Enterprise Ontologies". *Proceedings of the 4th International Workshop on Answer Set Programming*, Porto, Portugal, September 8–13, 2007.
4. Eiter T., Gottlob G., Mannila H. Disjunctive Datalog. ACM TODS 22(3) (1997) 364418.
5. Eiter T., Lukasiewicz T., Schindlauer R., Tompits H. Combining Answer Set Programming with Description Logics for the Semantic Web. In: Principles of Knowledge Representation and Reasoning: Proceedings of the Ninth International Conference (KR2004), Whistler, Canada. (2004) 141–151.
6. Gelfond M., Lifschitz V. Classical Negation in Logic Programs and Disjunctive Databases. NGC, vol. 9, pg. 365-385, 1991.
7. Horrocks I., Patel-Schneider P.F., Boley H., Tabet S., Grosof B., Dean M. Swrl: A semantic web rule language combining owl and ruleml W3C Member (2004) Submission. http://www.w3.org/Submission/SWRL/.
8. Krotzsch M., Hitzler P., Vrandecic D., Sintek M. How to reason with OWL in a logic programming system. In Proceedings of the Second International Conference on Rules and Rule Markup Languages for the Semantic Web, RuleML2006, pp. 17–26. IEEE Computer Society, Athens, Georgia, November 2006.
9. Leone N., Pfeifer G., Faber W., Eiter T., Gottlob G., Perri S., Scarcello F. The DLV System for Knowledge Representation and Reasoning. ACM TOCL 7(3) (2006) 499562
10. Motik B. Reasoning in Description Logics using Resolution and Deductive Databases. Phd Thesis. 2006 Karlsruhe.
11. Motik B., Horrocks I., Rosati R., Sattler U. Can OWL and Logic Programming Live Together Happily Ever After? 5th International Semantic Web Conference, Athens, GA, USA, November 5-9, 2006, LNCS 4273.
12. Ricca F., Leone N. "Disjunctive Logic Programming with types and objects: The DLV+ System". *Journal of Applied Logic*, Elsevier, Volume 5, Issue 3, September 2007, Pages 545-573.
13. Ruffolo M., Leone N., Manna M., Sacc D., Zavatto A. Esploiting ASP for Semantic Information Extraction. In proceedings of the ASP'05 workshop - Answer Set Programming: Advances in Theory and Implementation, University of Bath, Bath, UK, 27th–29th July 2005.
14. Ruffolo M., Manna M. A Logic-Based Approach to Semantic Information Extraction. In proceedings of the 8th International Conference on Enterprise Information Systems (ICEIS'06), Paphos, Cyprus, May 23-27, 2006
15. Ruffolo M., Manna M., Oro E. Object Grammar. Internal Report of High Performance Computing and Networking Institute of the Italian National Research Council. 2007
16. Smith M. K., Welty C., McGuinness D. L. OWL web ontology language guide. W3C Candidate Recommendation (2003) http://www.w3.org/TR/owl-guide/.

The 6th International Semantic Web Conference and the 2nd Asian Semantic Web Conference

November 11~15 2007
BEXCO, Busan KOREA