SEMANTIC INTEGRATION OF DISPARATE DATA SOURCES IN THE COG PROJECT

Jos de Bruijn
Digital Enterprise Research Institute
University of Innsbruck, Technikerstrasse 13, A-6020, Innsbruck, Austria

Keywords: Semantic Information Integration, Semantic Information Management, COG Project, Enterprise Application Integration.

Abstract: We present a novel approach to the integration of structured information sources in enterprises, based on Semantic Web technology. The semantic information integration approach presented in this paper was applied in the COG project. We describe Unicorn's Semantic Information Management along with the Unicorn Workbench tool, which is a component part of the Unicorn System, and how they were applied in the project to solve the information integration problem. We used the Semantic Information Management Methodology and the Unicorn Workbench tool to create an Information Model (an ontology) based on data schemas taken from the automotive industry. We map these data schemas to the Information Model in order to make the meaning of the concepts in the data schemas explicit and relate them to each other, thereby creating an information architecture that provides a unified view of the data sources in the organization.

1 INTRODUCTION

Information integration across application boundaries or even across company boundaries is a topic of interest to many enterprises and according to some studies, up to 30%[1] of future IT spending will go into Enterprise Application Integration (EAI). There are several reasons for the increase in the need of integration existing applications (Fensel, 2003): company mergers require integration of software infrastructure; new software systems need to be integrated with legacy systems; there are no integrated optimal solutions for all the needs of a company and new protocols; standards continue to emerge and evolve and companies need to be compliant with these new standards in order to enable cooperation with other companies.

In the Corporate Ontology Grid (COG) project we aim to overcome the problems in semantic heterogeneity between data sources in by semantic integration of the sources using a central Information Model (i.e. ontology). We built the Information Model using existing applications, data sources (assets) and input from domain experts. We then created a mapping between each data asset and the central model, thereby assigning a well-understood meaning to the concepts in each asset. The mappings now enabled us to, using the Information Model, discover the location of information throughout the data sources in the enterprise. Furthermore, because the mappings are created in a formal way, they enable us to automatically generate transformations between different sources.

In section 2 we present the Unicorn Workbench tool, which we used to solve the information integration problem in the COG project. Then, we present the information integration problem and the solution for the COG project in section 3. We then mention some related work in section 4 and finally provide some conclusions.

2 INTRODUCING THE UNICORN WORKBENCH

The Unicorn Workbench, a java-based tool created by Unicorn, was built to support the Unicorn Semantic Information Management (Schreiber, 2003) (SIM) and to enable SIM implementations in enterprises. All phases (except the first) in the SIM Methodology are to some extent supported by the Unicorn tool.

The basic concept in Unicorn is the Unicorn

[1] David Sink in InformationWeek, May 2002, InfoWorld January 2002 survey of 500 IT leaders.

de Bruijn J. (2004).
SEMANTIC INTEGRATION OF DISPARATE DATA SOURCES IN THE COG PROJECT.
In *Proceedings of the Sixth International Conference on Enterprise Information Systems*, pages 9-14
DOI: 10.5220/0002623000090014

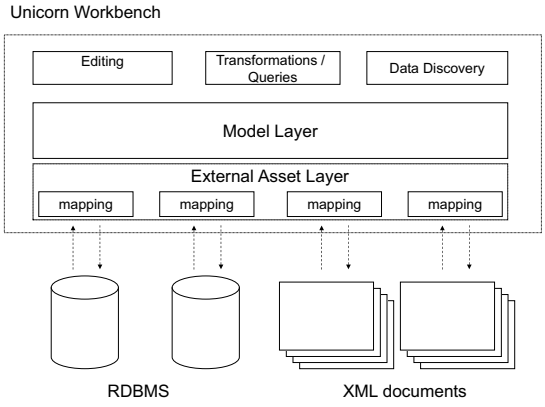

Figure 1: Semantic Information Management

Project, which consists of the Information Model, the schemas belonging to the external assets (the data sources), the transformations, the queries, and the descriptors (meta-data for human readers) for all these concepts.

The architecture of Unicorn consists of two main layers (see figure 1), namely:

- The *External Assets layer* contains the mappings to all the (disparate) data sources. All kinds of data schemas can be imported into a Unicorn project, as long as there is a parser for it. Current supported formats are relational database schemas, XML schemas, as well as COBOL Copybooks. New parsers can be written using the Asset API.
- The *Model Layer* contains the ontology (also called *Information Model*). The ontology contains all the packages, classes, properties, and business rules for describing the meaning of the data residing in the external assets.

The Model layer describes the meaning of the data and the External Asset layer describes the location of the data. In order to make the ontology active, the Unicorn Workbench provides three functions for the user. An editing function, used to create and maintain the ontology and the mappings to the different data sources. The second function is the data discovery function, which can be employed by the user to discover the location of data, residing in the disparate data sources, using the ontological model in Unicorn. Finally, there is a transformation and querying function with which the user can create transformations of instances between different data sources and issue queries against the ontology. The queries are syntactically translated to the query language of the data source and semantically translated (i.e. the query is automatically rephrased using terms from the external asset) to be used with the data schema of the source.

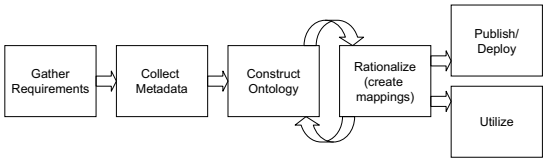

Figure 2: The Semantic Information Management Methodology

Besides these external assets, other Unicorn ontologies and ERWin[2] models can be directly imported into the Model Layer of the project (the ontology). In this way, existing logical data models and component ontologies can be leveraged in the project. Using component ontologies decreases the development effort of the project and allows to leverage proven models.

2.1 Ontology Engineering In The Unicorn Workbench

In the COG project, we followed the Semantic Information Management (SIM) Methodology (Schreiber, 2003) for the creation of the ontology and the mapping of the disparate data sources. The SIM Methodology (see Figure 2) consists of six steps:

1. *Gather requirements.* Requirements for the information architecture are collected and the scope of the project is established.
2. *Collect Metadata.* All data assets relevant to the project are catalogued and the metadata (i.e. data schemas and existing conceptual models) are imported into the Unicorn Workbench.
3. *Construct Information Model.* Using the imported metadata, the ontology is created through a process of reverse engineering and/or manual identification of classes, properties, and business rules in the source schemas.
4. *Rationalize.* In the rationalization phase, the mappings between the data schemas and the ontology are created. If the model needs to be refined, there will be an iteration step back to phase three. In general, when creating mappings from the composites in the external assets to the ontology, missing classes, properties, and business rules are discovered, which necessitates many iterations between the phases three and four to complete the model and the mappings.
5. *Publish/Deploy.* The Information Model, along with the mappings, is published to relevant stakeholders and the information model along with the

[2] An Entity-Relationship Diagram editor, see http://www3.ca.com/Solutions/Product.asp?ID=260

transformations between the data sources is deployed to the runtime architecture.

6. *Utilize.* Processes need to be created to ensure maintenance of the architecture. Changes in the data sources need to be reflected in the ontology, mappings need to be updated according to these changes, etc ...

The implementation of phases two up to and including six of the methodology are facilitated by the Unicorn Workbench tool.

This paper focusses on the third and fourth phase of the methodology, the support by the Unicorn tool, and our experiences with the methodology and the tool in the COG project. We first describe the Unicorn Workbench tool and then our experiences in the COG project.

2.2 Data Schema Integration In The Unicorn Workbench

During the metadata collection phase, a number of data schemas have been imported in the Unicorn project. These schemas have been used in the construction phase to aid in constructing the ontology. These source schemas now have to be mapped to the ontology in order to give meaning to the data residing in these sources and to enable locating data and to enable data transformation between the disparate sources and issuing queries to the sources. Just like in the construction phase, it is very important to involve the domain experts in the mapping (rationalization) phase.

During the rationalization phase the user is aided by the Unicorn Workbench in creating the mappings between the data assets and the ontology. It is possible to either view the mappings from the viewpoint of the data assets and in this way determine for each type and property to decide to which class/property it should be mapped and to create the mapping. Another possibility is to use the Data Discovery feature to drill down the class hierarchy to find out which mappings currently exist for the classes in the ontology. If a class is identified that requires further mapping, the designer can switch back to the view of the desired data asset and create the mappings.

If during the mapping phase, the designer discovers missing classes, properties, or business rules, the designer iterates back to the construction phase (phase three) to add the necessary concepts to the model. It is our experience from the COG project that especially missing properties and business rules are discovered in the rationalization phase and not so much missing classes.

Mappings are created in two stages. First the *Coarse Mapping* (see Figure 3) is created, linking composites in the data sources (e.g. tables, complex

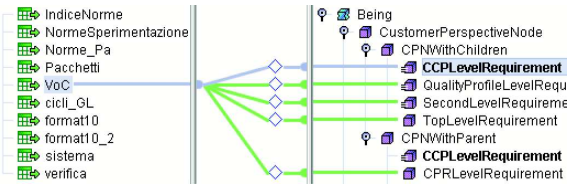

Figure 3: Coarse mapping in the Unicorn Workbench

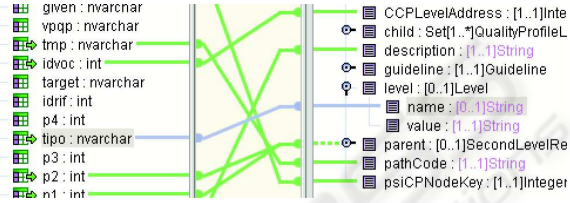

Figure 4: Detailed mapping in the Unicorn Workbench

types) to classes in the ontology. Then the *Detailed Mapping* (see Figure 4) is created, linking atoms from the source schemas to the properties in the ontology.

Conditions can be specified on the values of the atoms of the composite in the source schema. This way certain groups of instances can be mapped to different classes in the ontology. These conditional instance mapping groups are, however, not required to be mutually exclusive. Conditional mapping can not only be applied to classes (global conditional mapping), but also to individual properties (local conditional mapping), as mappings are eventually created on the property level, during the detailed mapping. The other way around is also possible: different composites can be mapped to the same class.

The mappings in the detailed mapping stage are usually made between atoms and *direct* properties. A direct property is a regular property of the concerning class. It is however also possible to create mappings to *indirect properties*. An indirect property is not a property of the concerning class, but a direct or an indirect property of a class, which is the type a direct property of the class (i.e. an indirect property is a property of a related class). It is therefore possible to map to an indirect property at an arbitrary depth.

When instances of different composite types in the source schema need to be mapped to a single class in the ontology, it is possible to create a mapping view. Such a mapping view can contain joins over different composites within one external asset. A join can be defined over the composites using an existing or an (in Unicorn created) "implicit" foreign key.

Foreign keys in the database or "implicit" foreign keys are used either to indicate a simple relationship with another class or to indicate inheritance. In the

former case, the foreign key can be mapped to a property in the class that references the target class that represents the target composite type of the foreign key. In the latter case, the foreign key is mapped directly to the inheritance relationship. This latter case applies when the extension of the composite (i.e. the set of instances described by the composite) is a subset of the extension of another composite and this relationship is made explicit in the database using a foreign key.

When mapping external assets to the central ontology, it is possible to use so-called subtype mapping. An atom in a composite in the external asset can be mapped to a subclass of the type of the property in the ontology. This is of course a valid mapping, because a subclass of the original type is also a valid type for the property.

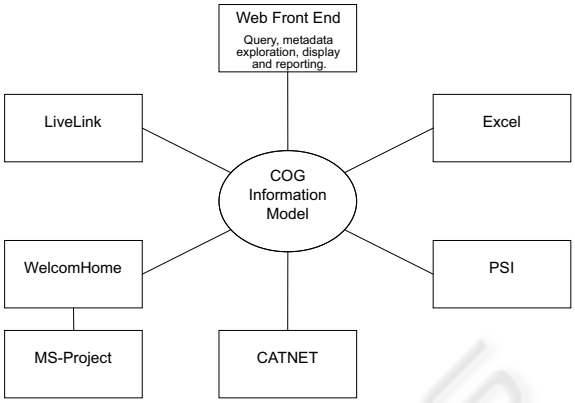

Figure 5: COG logical architecture. Shows the integration of the different data sources using the Unicorn Workbench tool.

3 INFORMATION INTEGRATION IN THE COG PROJECT

In the COG project we used external assets provided by CRF (Centro Ricerche Fiat). The source schemas are taken from real-life data sources currently in use by CRF and sources to be implemented at CRF in the future. The goal of the project is to implement a single integrated (semantic) information architecture for the various information sources provided by CRF in order to show the applicability of using ontologies for information integration in industry. The sources include relational databases, XML data sources and PDF and spreadsheet documents. These PDF documents are accessed using the LiveLink document management system, which in turn has an XML interface. For the integration of the Excel spreadsheet documents, a special parser was written using the Asset API.

3.1 The Information Integration Problem In The COG Project

There were five main data sources (see also Figure 5) provided by CRF to be integrated in the COG project. These sources consist of three relational databases, namely CATnet, PSI, and WelcomHome, one XML data sources, namely LiveLink, and a collection of Excel spreadsheet documents.

WelcomHome is an application used for project management (it supports the Microsoft Project application). LiveLink is a knowledge and document management system. CATnet and PSI are applications developed in-house at CRF to support the vehicle development and testing process.

CATnet has been developed to support the entire testing process, which is a major part of the FIAT vehicle (prototype) development process and essential for ensuring the quality of the products. Test requests that are linked to test plans are submitted to the CATnet system. These test plans support the planning of the execution of the tests, which is related to the vehicle development phase, the test laboratory, etc...

The test plans are linked to the so-called "technical memory", where the technical procedures for test execution are specified. This technical memory consists of PDF documents accessible through the LiveLink system. After the test executions, the test results are stored in the CATnet database for later retrieval. This retrieval of test results is critical in order to assure quality of the products. Whenever a customer complaint is received or a defect in a vehicle is detected, it has to be possible to easily retrieve the results of the tests performed on the concerning automobile component.

To facilitate this retrieval process, there is a customer perspective (also called the 'Voice of Customer' or VoC) tree for each vehicle model in the PSI system. The customer perspective trees are used to locate the appropriate tests that should have been performed on a particular vehicle system. The marketing manager can drill down the tree to locate the specific tests that have been performed, after which the test results can be retrieved from the CATnet system.

3.2 Solving the integration problem in COG using the Semantic Information Management

During the development of the ontology in the COG project, some shortcomings in the Unicorn tool became apparent. In order to overcome these shortcomings, these problems were fed into the requirements

for the new version of the Unicorn Workbench. During the COG project, a new version of Unicorn (v2.5) was released that overcomes most shortcomings in the old version identified during the development for COG. Examples of features developed especially for the COG project are the mapping to subtypes, specification of inverse properties, and mapping of foreign keys to inheritance relationships.

Another important development for the new version of Unicorn that has proven very useful for the COG project was the Asset API. Using this API it was possible to develop parsers for the integration of Excel documents and Microsoft Access databases (i.e. PSI) that can be used for importing data schemas from these platforms. Without this Asset API and these parsers, these sources could not have been integrated.

Based on the data sources provided by CRF and the interviews with domain experts at CRF, four main subject areas for the Information Model were identified, namely:

- *Motor vehicle development*. This area covers the motor vehicles themselves, such as automobile parts and vehicle systems.
- *Project management*. This area covers the project management capabilities present in WelcomHome.
- *Testing Management*. This area covers the management of the execution of the tests. Things like testing guidelines, test plan, testing laboratories, and so on, are the components of this area.
- *Test requests*. This area covers the customer perspective tree, standards, test requests, test results, etc ...

For each of these subject areas a package in the ontology was created, after which the classes, properties, and business rules were created corresponding to the composites in the source schemas. Interviews with domain experts were mostly used as input for the ontology development process, as well as working with the current applications (mainly PSI and CATnet) in use. This was the third phase (the construction phase) in the SIM methodology.

The next step (phase four - rationalization) was to map the source schemas to the ontology. This mapping was done using the mapping functionality of the Unicorn tool. During this mapping phase, many iterations back to phase three were necessary. It turned out that the classes in the ontology had been identified correctly, but still many properties and business rules had to be added or changed while trying to map the external assets.

4 RELATED WORK

The MOMIS (Bergamaschi et al., 2001) approach is a semi-automatic approach to schema integration, developed at the university of Modena, Italy. The approach has not been used in any major industrial projects and is mainly an academic research activity. Any data source can be connected to the architecture, as long as an ODL_{I^3} wrapper is created. MOMIS has a single mediator, which provides a global data schema and query interface to the user.

InfoSleuth (Fowler et al., 1999) is a multi-agent system for semantic inter-operability in heterogeneous data sources. Agents are used for query and instance transformations between data schemas. An agent is aware of its own ontology and the mapping between that ontology and the data schema, it is aware of the shared ontologies and it can map its ontology to those of other agents. InfoSleuth uses several shared ontologies, made available through the ontology agents. Individual data sources have (through the resource agents) a mapping to these shared ontologies. The shared ontologies are linked together through one-to-one ontology mapping. Note that the user agents use the shared ontologies as their vocabulary and local ontologies are only maintained by the resource agents.

PROMPT (Noy and Musen, 2000) provides a a semi-automatic approach to ontology merging, not specifically data schema integration. In the ontology merging in PROMPT it is assumed that the original ontologies no longer exist after the merged ontology has been created. Therefore, there are no mappings between the original and the merged ontologies, as there are in most data schema integration solutions.

ONION (Mitra and Wiederhold, 2001) takes a centralized, hierarchical approach to ontology mapping, where the user views the (global) articulation ontologies. The source ontologies are mapped to each other via articulation ontologies that are in turn used by the user to express queries. The articulation ontologies are organized in a tree structure. An articulation ontology used for the mapping of two source ontologies can in turn be one of the sources for another articulation ontology. The creation of a hierarchy can be seen as a form of ontology clustering. But while (Visser and Tamma, 1999) take a top-down approach (first the root application ontology is specified, then child ontologies are created as is necessary), ONION takes a bottom-up approach in the creation of the articulation ontologies; furthermore, there is no defined root ontology for the cluster.

5 CONCLUSION

In this paper we have outlined the problems in information integration we were facing in the COG project. We have described a solution to the integration problem using the Semantic Information Management (SIM) Methodology (Schreiber, 2003) and the Unicorn Workbench tool, which we have applied in the COG project.

We have described how the SIM together with the Unicorn Workbench was used in the COG project and what the role is in the overall COG Architecture.

Many problems in the construction of the Information Model and the mapping to the disparate data schemas were caused by the poor understanding of the source data schemas. The data schemas contained concepts in the Italian language, while the ontology engineering and mapping was done by non-Italian speakers. What further complicated the matter was the fact that the users that worked with the existing applications were no expert on the database schemas that were being used, which made the mapping a hard problem. It turned out that the only possibility for the ontology engineer to construct the ontology was to have a look at the applications together with the end-users, which was a tedious job.

These problems indicate the necessity of the usage of a central Information Architecture, through which the nature of the data residing throughout the organization can be understood.

Much of the mentioned related work consists of academic research prototypes. The Unicorn Workbench tool, along with the Semantic Information Management architecture, has proven itself in many projects in an industrial setting.

Many of the mentioned approaches take a semi-automatic approach to the data schema (or ontology) mapping. The mentioned approaches all use ontology mapping or ontology merging as a basis. The Unicorn Workbench does not map ontologies explicitly, but is specialized in the mapping of database (and several other types of) schemas to a central ontology and provides an integration platform for data sources throughout the enterprise.

Major limitations of the current Unicorn Workbench tool are the lack of support for semi-automatic mapping, as in PROMPT (Noy and Musen, 2000) and Chimæra (McGuinness et al., 2000), and the lack of support for the integration of ontologies. The tool support only the integration of data sources into one ontology, and does not support the mapping of several ontologies in different organization(al unit)s. The existence of only one ontology can lead to several problems, as pointed out in (Visser and Cui, 1998) and (Uschold, 2000).

ACKNOWLEDGEMENTS

The research presented in this paper was funded by the COG project, under contract number IST-2001-38491, http://www.cogproject.org/. Some materials presented in this paper are the copyright of Unicorn Solutions, Inc. and are used with permission.

We would like to acknowledge all partners in the COG project and all people in DERI who have looked at earlier versions and provided useful feedback.

REFERENCES

Bergamaschi, S., Castano, S., Beneventano, D., and Vincini, M. (2001). Semantic integration of heterogeneous information sources. *Special Issue on Intelligent Information Integration, Data & Knowledge Engineering*, 36(1):215–249.

Fensel, D. (2003). *Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce, 2nd edition*. Springer-Verlag, Berlin.

Fowler, J., Nodine, M., Perry, B., and Bargmeyer, B. (1999). Agent-based semantic interoperability in infosleuth. *SIGMOD Record*, 28(1).

McGuinness, D., Fikes, R., Rice, J., and Wilder, S. (2000). An environment for merging and testing large ontologies. In *Proc. 7th Intl. Conf. On Principles of Knowledge Representation and Reasoning (KR2000)*, Colorado, USA.

Mitra, P. and Wiederhold, G. (2001). An algebra for semantic interoperability of information sources. In *IEEE International Conference on Bioinformatics and Biomedical Egineering*, pages 174–182.

Noy, N. F. and Musen, M. A. (2000). Prompt: Algorithm and tool for automated ontology merging and alignment. In *Proc. 17th Natl. Conf. On Artificial Intelligence (AAAI2000)*, Austin, Texas, USA.

Schreiber, Z. (2003). Semantic information management: Solving the enterprise data problem. To be found on the http://www.unicorn.com/ website.

Uschold, M. (2000). Creating, integration, and maintaining local and global ontologies. In *Proceedings of the First Workshop on Ontology Learning (OL-2000) in conjunction with the 14th European Conference on Artificial Intelligence (ECAI-2000)*, Berlin, Germany.

Visser, P. and Cui, Z. (1998). On accepting heterogeneous ontologies in distributed architectures. In *Proceedings of the ECAI98 workshop on applications of ontologies and problem-solving methods*, Brighton, UK.

Visser, P. and Tamma, V. (1999). An experience with ontology clustering for information integration. In *Proceedings of the IJCAI-99 Workshop on Intelligent Information Integration in conjunction with the Sixteenth International Joint Conference on Artificial Intelligence*, Stockholm, Sweden.