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.

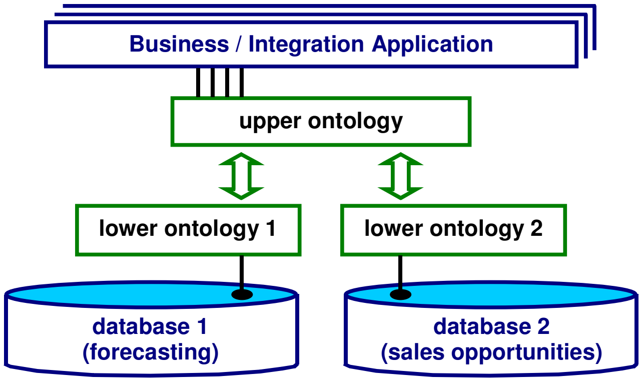

Figure 1 - Integration System Architecture

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.

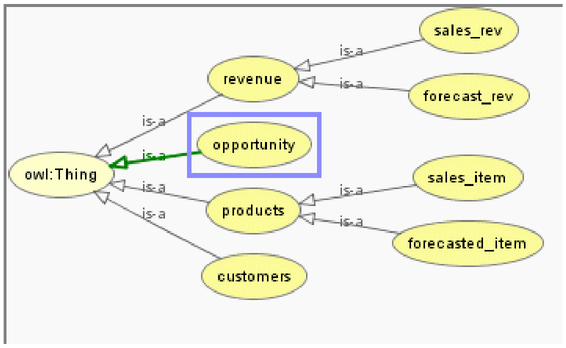

Figure 2 – Class View, Upper Ontology

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.