Achieving Semantic Interoperability in Heterogeneous Information Systems with Topic Maps

Giovani Rubert Librelotto[*] , José Carlos Ramalho[1] , Pedro Rangel Henriques[1]

[1]Centro de Ciências e Tecnologias da Computação (CCTC)
Departamento de Informática, Universidade do Minho
Braga, Portugal, 4710-057

{grl,jcr,prh}@di.uminho.pt

Abstract. *Nowadays, data handled by an institution or company is spread out by more than one database and lots of documents of different types. To extract the information implicit in that data, it is necessary to pick parts from those various archives. To obtain a general overview, those information slices should be gather. Different approaches can be followed to achieve that integration, ranging from the merge of resources till the fusion of the extracted parts. In this paper, we introduce Metamorphosis — a Topic Maps oriented environment to generate conceptual navigators for heterogenous information systems — and we argue that Metamorphosis can be used to achieve, via Topic Maps, the referred interoperability.*

1. Introduction

Daily, a lot of data is produced by every institution or company. To satisfy the storage requirements, these organizations use most of the times relational databases, which are quite efficient to save and to manipulate structured data. Unstructured data (appearing inside documents) is stored in plain or annotated text files.

There is a problem when these organizations require an integrated view of their heterogeneous information systems. It is necessary to query/exploit every data source, but the access to each information system is different. In this situation, there is a need for an approach that extracts the information from those resources and fuses it.

Topic Maps are a good solution to organize concepts, and the relationships between those concepts, because they follow a standard notation – ISO/IEC 13250 [Biezunsky et al., 1999] – for interchangeable knowledge representation. Topic Maps are composed of topics and associations giving rise to structured semantic network that gathers information concerned with certain domain. This hierarchical topic network can represent an ontology. This is the reason why we are using successfully, for some years, this technology for classification and integration of documents in the area of digital archiving.

However, the process of ontology development based on topic maps is complex, time consuming, and it requires a lot of human and financial resources, because they can have a lot of topics and associations, and the number of resources can be very large.

To overcome this problem, *Metamorphosis* was proposed. *Metamorphosis* make possible the Topic Maps extraction, validation, storage, and browsing. It is composed of three main modules: (1) *Oveia* extracts data, from heterogeneous information systems, according to an ontology specification, and stores it in a topic map; (2) *XTCHE* validates the generated

[*]Bolsista CNPq - Brasil.

topic map, according to a constraint specification; (3) *Ulisses* browses the topic map, giving a conceptual view over the resources.

This way, *Metamorphosis* let us achieve the semantic interoperability in heterogeneous information systems because the relevant data, according to the desired information defined in an ontology specification, is extracted and stored in a topic map. The environment validates this generated topic map against a set of rules defined in a constraint language. That topic map provides information fragments (the data itself) linked by specific relation to concepts at different levels of abstraction. Moreover, the navigation over the topic map is led by a semantic network and provides an homogeneous view over the resources – this justifies our decision of call it *semantic interoperability*.

Enabling to create a virtual map of information, the information systems are kept in their original form, they are not changed. Then, the same resource can be used by *Metamorphosis* in different ways, for different topic maps. As it is possible and easy to change the map itself, data reuse is achieved.

In section 2 there is an overview about Topic Maps. The proposed architecture and its main characteristics are presented in section 3, and the following sections describe the main modules of *Metamorphosis*: *Oveia* (section 4), *XTCHE* (section 5), and *Ulisses* (section 6). Section 7 presentes a synthesis of this paper.

2. Topic Maps

Topic Maps are being increasingly accepted as an excellent option when it is necessary to organize information according to different points of view simultaneously. Topic Maps are designed to manage the infoglut, building valuable information networks from an overabundance of information; the approach enables the structuring of unstructured resources of any kind.

Topic Maps can be seen as a description of what there is about a certain domain (knowledge), by formally declaring topics, and by linking the relevant parts of the information set to the appropriate topics [Park and Hunting, 2003].

A topic, in its most generic sense, can be anything. A person, an entity, a concept, really anything regardless of whether it exists or has any other specific characteristics. It is the basis for the topic maps creation.

A topic can have one or more occurrences. An occurrence represents the information that is specified as relevant to a given topic. Occurrences and topics exist on two different layers or domains (physical and conceptual), but they are *connected*.

These topics also have associations that represent and define relationships between them. As described in ISO/IEC 13250:2000 [Biezunsky et al., 1999], a topic association specifies a relationship among specific topics (e.g. that Professor *is supervisor of* Student or Student *studies at* University). A topic association is a link between topics, each of which plays a role as a member of that association (e.g. professor supervises the student and student prepares his thesis).

XML Topic Maps (XTM) 1.0 specification [Pepper and Moore, 2001] is a XML dialect for writing Topic Maps, and it was developed to apply the Topic Maps paradigm to the World Wide Web.

3. Metamorphosis

The main idea behind *Metamorphosis* is to integrate the specification of conceptual networks or ontologies, with their storage and navigation, as well as, their automatic extraction and validation.

One of the first *Metamorphosis*' applications was the production of site maps; another of our former concerns was the contents publishing in the context of e-learning. *Metamorphosis* can be also used to test some functionalities of a dynamic web system because it creates, in a fast way, a web interface that interacts directly with data sources.

Metamorphosis takes as input:

Information resources: composed of one or more data sources: XML documents, web pages, databases, ... *Metamorphosis* does not interfere with any of it, it will only use part of the information to build the semantic network;

XML Specifications: the description of data sources (written in XSDS – *XML Specification for DataSources*); the description of the ontology (written in XS4TM – *XML Specification for Topic Maps*); and the description of the constraints to be complied by topic map instance (written in XTCHE – *Topic Maps Schema and Constraint Language*).

and generates as output:

Conceptual Website: The final generated website through which it is possible to navigate through the information system driven by concepts organized in a semantic network.

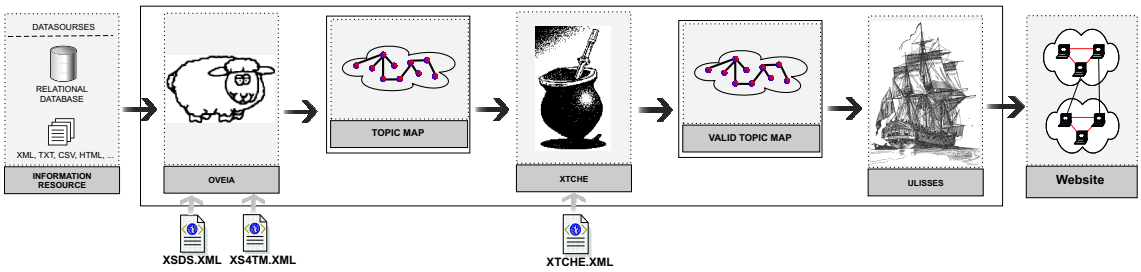

Figure 1: Metamorphosis Architecture

Figure 1 shows *Metamorphosis*' architecture that came up from the principles underlying our proposal. This architecture is composed of:

(1) *Oveia*: The processor that builds topic maps. Its core is a processor that extracts the topics instances from the information resources and builds a topic map. It reads and processes the XSDS and XS4TM specifications.

(2) Generated topic map: The topic map automatically generated by *Oveia* stored as an XTM file or alternatively a relational database.

(3) *XTCHE*: The processor that consumes the previous XTM file and verifies the topic map according to a set of constraints defined in XTCHE language.

(4) Valid topic map: The previous topic map automatically validated by *XTCHE*.

(5) *Ulisses*: The processor that takes a topic map and produces a whole semantic website, a set of web pages where it is possible to navigate through structural or syntactic links as well as through a network of concepts.

In the next sections we are going to discuss the main pieces of this architecture: *Oveia* (section 4), *XTCHE* (section 5), and *Ulisses* (section 6), in order to demonstrate how the overall system can accomplish the task we have stated at the beginning.

4. Oveia – A topic map builder

The ontology extractor – *Oveia* (more details in [Librelotto et al., 2004b]) – is based on ISO/IEC 13250 Topic Maps [Biezunsky et al., 1999]. *Oveia* extracts information fragments from heterogeneous information systems according to an XSDS specification and builds the topic map according to an ontology specified in *XS4TM* language.

The *Oveia* architecture is shown in figure 2 and it is composed mainly of five components. The *dataset extractor* receives an *XSDS specification* – providing metadata about the *physical data sources* that will be used to query each source in order to get the data needed for the ontology construction – and generates the intermediate representation (called *datasets*) – containing the data (in a unified representation) extracted from resources. The *XS4TM processor* takes as input these datasets and an *XS4TM specification* generating a *topic map*, in an internal format. An *output generator* stores the topic map in an *OntologyDB* or in an *XTM file*. The following subsections describe this architecture in detail.

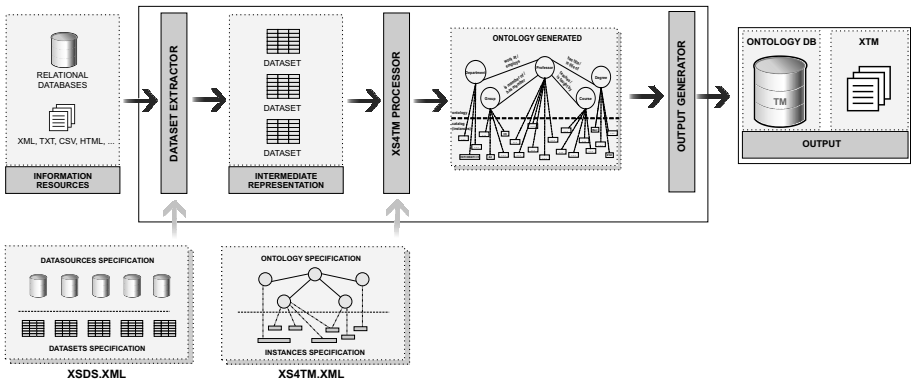

Figure 2: Oveia Architecture

4.1. XSDS – XML Specification for DataSources

Oveia supports the concept of extraction drivers. A driver extracts data from a data source and store it in an intermediary representation, called *datasets*. XSDS language defines the transformations and filters over the data sources. XSDS gives precise information about each data source that should be scanned to extract topics and associations.

An XSDS specification has two parts: *datasources* and *datasets*. The first one defines the path to the physical resources. Each resource is defined in a <*datasource*> element. This element has a set of attributes that indicate which extractor driver will be used and the necessary parameters, because each driver has its own attributes. The second part of this specification is defined in a <*datasets*> element. It declares which data (record fields or DTD elements) must be extracted from each *datasource*. Each *datasource* can be used to declare several *datasets*.

4.2. Datasets: Intermediate Representation

The *datasets* compose the intermediate representation that contains the extracted data from the resources. Each *dataset* has a relation to an entity in these resources and it is represented through a table, where each line is a record following the structure specified in XSDS. The *datasets* representation guarantees that *Oveia* sees an uniform data structure that represents all the participating resources.

The *dataset* declaration is composed by a query to extract the data from resources. Each *dataset* has an unique identifier. This identifier will be used throughout the architecture to reference a particular *dataset*.

The fundamental idea is that all objects have labels that describe their meaning. For instance, the following object represents a member's category: <1, PhD>, where the string *1* is a identifier of this category, and *PhD* is a human-readable label. The datasets are very simple, while providing the expressive power and flexibility needed for integrating information from disparate sources.

4.3. Dataset Extractor

The *Dataset Extractor* is a processor that reads the input files and parses them to get desired data into the *datasets*, in agreement with an XSDS specification.

The *Dataset Extractor* is composed of several extraction drivers (at moment, two), each one responsible for handling specific type of source. The driver uses the appropriated mechanisms to make the connection (e.g. JDBC – *Java DataBase Connectivity* – for databases, and an XML parser for annotated documents), and then the extraction data is performed in the query language adequate to the type of source in use: SQL will be used to extract information from a relational database while XPath will be used for the extraction in XML documents. Finally, the data extracted is stored in the *datasets*.

4.4. XS4TM – XML Specification for Topic Maps

XS4TM is a domain specific language conceived to specify the process of ontology extraction from information systems; in our case, from the *dataset*.

Looking at a topic map an ontology designer can think of it as having two distinct parts: an ontology and an object catalog (instances). The ontology is defined by topic types, association types, occurrence role types, etc. The catalog is composed by a set of pointers to information objects that are present in the resources and are linked to the ontology. So, a specification in XS4TM is composed of two parts:

Ontology: the definition of the ontology requires in XS4TM the same effort as in XTM; it is necessary to specify every topic type, association type, occurrence type, ...;

Instances: the instances definition describes each topic and association that will be extracted from the information resource.

The XS4TM Context Free Grammar is based in XTM 1.0 DTD[1]. The *ontology* and *instances* elements have the same syntax that the *topicMap* element in XTM model.

The XS4TM language is intended to make the specification of Topic Maps extraction more flexible. However, the use of XS4TM is not much more difficult because this language is an extension of the XTM standard; it means the XS4TM DTD includes and augments the XTM DTD. In XS4TM, the ontology is specified like in XTM: with the same elements and attributes. So, if the designer knows XTM syntax, he does not need to learn another syntax to specify ontology in XS4TM.

4.5. XS4TM processor

This component uses the XS4TM specification and retrieves the information it needs to build the ontology from the *datasets*. It is an interpreter that takes advantage of the information organization in *datasets* (an internal universal representation for extracted data) and generates all the associations between the relevant topics according to XS4TM.

The XS4TM processor's behavior can be described in three steps: reads the the XS4TM specification and extracts from the *datasets* the topics and associations found; creates the topic map; finally, stores it into an *OntologyDB* or an XTM file.

[1] http://www.topicmaps.org/xtm/1.0/#dtd

4.6. Oveia Output – OntologyDB or XTM file

Once we chose XML as our development framework, the first version of the output generator stored the topic map to a file in XTM format. However, XTM files can grow exponentially. Huge XTM files are space and time consuming making their processing a hard task, specially from the web server side; and the performance tends to be worse as the interaction activity grows. So, in real cases it is crucial to find other ways to store very big ontologies. Therefore, it was decided to use also database technology besides XTM files.

The Topic Maps model maps quite well into the relational model. This way it was decided to create a relational model for Topic Maps, named *OntologyDB*, following the structure mapping adopted in [Williams et al., 2000]. This model is easy to understand and to implement systematically.

The current version of the output generator can export the topic map to an XTM file and to a relational database. In the second case, the topic map, automatically generated by *Oveia*, is converted into related tables and stored in the *OntologyDB*.

5. XTCHE – A Topic Maps Semantic Validator

This section introduces the semantic validation issue in the area of Topic Maps. It presents the topic map constraint language (XTCHE [Librelotto et al., 2004a]) and respective processor that guarantees semantic validity according to a specification.

5.1. XTCHE – A Topic Maps Schema and Constraint Language

When developing real topic maps, it is highly convenient to use a system to validate it; this is, to verify the correctness of the actual instance against the formal specification of that family of topic maps (according to the intention of its creator).

The syntactic validation of a topic map is assured because it is described by an XML specification (XTM format). However, it is well known that structural correctness does not mean the complete meaningfulness of the map – semantics should also be guaranteed.

So, a specification language that allows us to define the schema and constraints of a family of Topic Maps is necessary. A list of requirements for the new language was recently established by the ISO Working Group – the ISO JTC1 SC34 Project for a Topic Map Constraint Language (TMCL) [Nishikawa and Moore, 2003]. XTCHE language meets all the requirements in that list.

XTCHE is designed to allow users to constrain any aspect of the topic map; for instance: topic names and scopes; association members, roles and players allowed in an association, instances of a topic (enumeration), association in which topics must participate, occurrences cardinality, etc.

Like XTM, XTCHE specifications can be too verbose; that way it is necessary to define constraints in a graphical way with the support of a visual tool. To overcome this problem, XTCHE syntax follows the XML Schema syntax; so, any XTCHE constraint specification can be written in a diagrammatic style with a common XML Schema editor. The textual output of that edition (XML Schema code) should now be processed to obtain a *TM-Validator*.

5.2. XTCHE Processor and TM-Validator

The XTCHE language, listing all the conditions (involving topics and associations) that must be checked, specifies the Topic Maps validation process (*TM-Validator*), enabling

the systematic codification (in XSL) of this verification task. In that circumstances we understood that it was possible to generate automatically the validator developing another XSL processor to translate an XTCHE specification into the *TM-Validator* XSL code.

According to Figure 3, the XTCHE processor is the *TM-Validator* generator; it takes a topic map constraint specification (an XML-Schema, written according to the XTCHE language), and generates an XSL stylesheet (the *TM-Validator*) that will process an input topic map to generate a valid topic map or error messages.

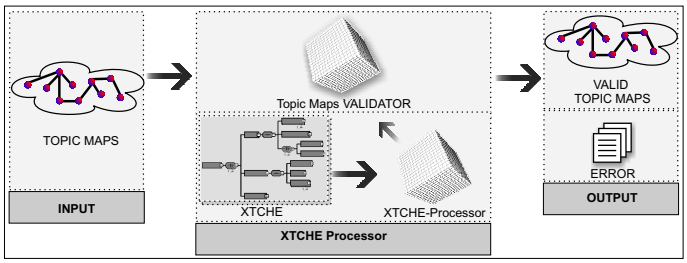

Figure 3: XTCHE validation process

Both XSL stylesheets (the generator and the validator) are processed by a standard XSL processor like Saxon[2], what in our opinion is one of the benefits of the proposal.

6. Ulisses – A Topic Maps driven Conceptual Navigator

In order to browse an ontology, specified with Topic Maps, we should see it as a graph, where nodes represent concepts (topics), and edges represent links to physical information resources (occurrences) or to other concepts (associations). So, using the graph it is obvious how to create a website for navigation (with a generic web browser): one web page for each concept (not only the name, but all its characteristics) associated with outgoing edges, each one implemented as a link to another web page. So, when a link is chosen a similar page is displayed showing another concept or an information record.

As the other 2 tools (*Oveia* and *XTCHE*), *Ulisses* – a conceptual browser described in [Librelotto et al., 2003] – is a generic processor that can be used outside *Metamorphosis*. It supports conceptual navigation over XTM files and OntologyDB databases. *Ulisses* allows the navigation over topic maps generated by tools (like *Oveia*) or by hand. In addiction, these sources can be previously checked for semantic correctness (according to a validation with XTCHE) or not.

So, when the source is a XTM file, this navigational component is just implemented as a set of XML transformations; when the source is *OntologyDB*, appropriate SQL-based tools are used to navigate over it.

7. Conclusion

This paper describes the integration of heterogeneous information systems using the ontology paradigm, in order to generate an homogeneous view of these resources. The proposal is an environment, called *Metamorphosis*, for the automatic construction of Topic Maps with data extracted from the various data sources and a semantic browser to look for the required information.

Although developed for use in our main working area – XML documents processing applied to Public Archives and Virtual Museums — we are convinced that *Metamorphosis*

[2]http://saxon.sourceforge.net/

can be applied with similar success in the general area of information system for data integration, analysis, and knowledge exploitation.

For instance, suppose that an user has an heterogeneous information system and he wants to make it accessible through the web, but when he starts creating his HTML index pages he ends up with pages of about 5 Megabytes; web browsers can not hold pages above 1 or 2 Megabytes, so, he get into trouble. In situations like that *Metamorphosis* can help a lot, on account of the way Ulisses structures the website.

A case study about an academic department was presented in [Librelotto et al., 2004b] describing the generation of a topic map from MySQL databases according XSDS and XS4TM specifications. The final result, stored in *OntologyDB* managed by Microsoft SQL Server 2000, had 4420 topics and 4223 associations; the same topic map could have 172490 lines in the XTM format. *Ulisses* provides the complete navigation over the generated topic map. The desired results were obtained in an acceptable time. At this stage, just a pragmatic test was made; we checked manually the topic maps produced.

Talking about future works, a friendly user-interface to write XS4TM and XSDS specifications is under development. In the near future, *Metamorphosis* will be tested with new case studies, and we will conceive an easy and systematic way to verify the generated topic map against the actual sources and specifications. To assure the absolute correctness of this environment, each module should be formally validated.

References

Biezunsky, M., Bryan, M., and Newcomb, S. (December, 1999). ISO/IEC 13250 - Topic Maps. ISO/IEC JTC 1/SC34. http://www.y12.doe.gov/sgml/sc34/document/0129.pdf.

Librelotto, G. R., Ramalho, J. C., and Henriques, P. R. (2003). Ontology driven Websites with Topic Maps. In *The International Conference on Web Engineering*, Oviedo, Spain.

Librelotto, G. R., Ramalho, J. C., and Henriques, P. R. (2004a). XTCHE - A Topic Maps Schema and Constraint Language. In *XML 2004 Conference and Exposition*, Washington D.C., U.S.A. IDEAlliance. (to be published).

Librelotto, G. R., Souza, W., Ramalho, J. C., and Henriques, P. R. (2004b). Using the Ontology Paradigm to Integrate Information Systems. In *International Conference on Knowledge Engineering and Decision Support*, pages 497–504, Porto, Portugal.

Nishikawa, M. and Moore, G. (2003). Topic Map Constraint Language (TMCL) Requirements and Use Cases. ISO/IEC JTC 1/SC34 N0405rev. http://www.isotopicmaps.org/tmcl/requirements.html.

Park, J. and Hunting, S. (2003). *XML Topic Maps: Creating and Using Topic Maps for the Web*, volume ISBN 0-201-74960-2. Addison Wesley.

Pepper, S. and Moore, G. (August, 2001). XML Topic Maps (XTM) 1.0. TopicMaps.Org Specification. http://www.topicmaps.org/xtm/1.0/.

Williams, K., Brundage, M., Dengler, P., Gabriel, J., Hoskinson, A., Kay, M., Maxwell, T., Ochoa, M., PaPa, J., and Vanmane, M. (2000). *Professional XML Databases*. Wrox Press.