Metamorphosis – a Topic Maps based Environment to Handle Heterogeneous Information Resources

Giovani Rubert Librelotto
José Carlos Ramalho Pedro Rangel Henriques

Departamento de Informática – Universidade do Minho
Campus de Gualtar
4710-057 – Braga – Portugal
{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. Usually this is achieved either by extracting data and loading it into a central repository that does the integration before analysis, or by merging the information extracted separately from each resource into a central knowledge base.

Topic Maps are a good solution to organize concepts, and the relationships between those concepts, because they follow a standard notation – ISO/IEC 13250 – for interchangeable knowledge representation. 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, we developed Metamorphosis. Metamorphosis makes 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 topic map, according to a constraint specification; (3) Ulisses browses the topic map, enabling a conceptual navigation and query over the resources.
This way, Metamorphosis let us achieve the semantic interoperability among heterogeneous information systems because the relevant data, according to the desired information specified through an ontology, 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 relations to concepts at different levels of abstraction. Note that not all data items need to be extracted from the sources to the Topic Map. We only extract the necessary metadata to build the intended ontology. This ontology will have links to enable a browser to access all data items.
Thus 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.
Creating a virtual map of the information enables us to keep the information systems in their original form, without changes. It is also possible to create as many virtual maps as the user wants generating multiple semantic views for the same sources.
The remainder of the paper is structured in the following sections: next section (sec.2) will introduce Metamorphosis, then a description of each module is presented with some detail (Oveia in sec.3, XTche in sec.4 and Ulisses in sec.5). Before concluding remarks (sec.7) we present a real world case study to consolidate our proposal — *"Emigration Museum"* (sec.6).

2 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 creation and validation.
One of the first Metamorphosis' applications was the production of website maps for conceptual navigation; 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 tyhose, it will only use part of their 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 instances (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.

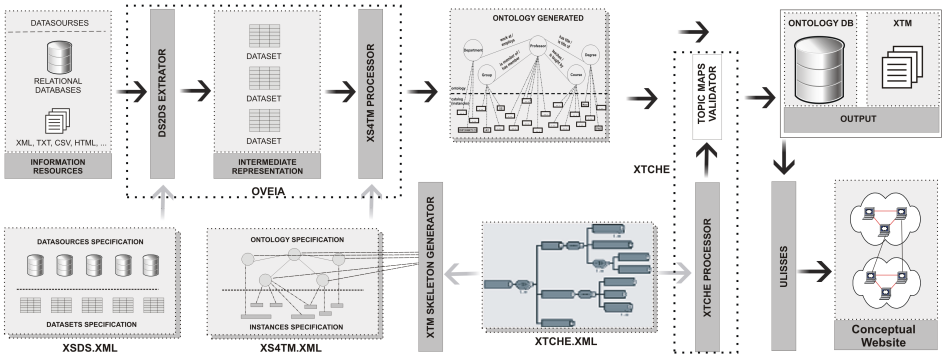

Fig. 1. Metamorphosis Architecture

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, XTche, and Ulisses, in order to demonstrate how the overall system can accomplish the task we have stated at the beginning.

3 Oveia

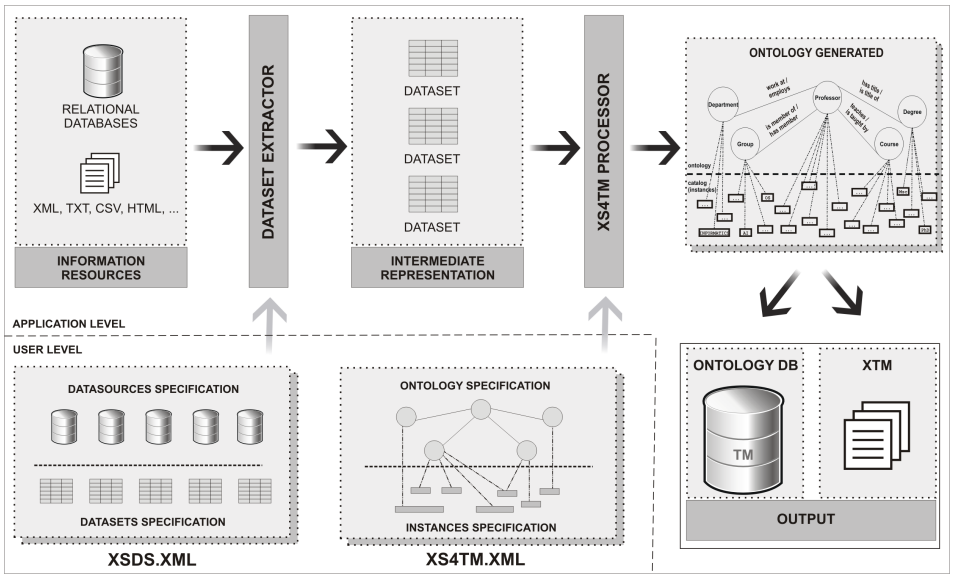

Fig. 2. Oveia Architecture

The ontology extractor Oveia (more details in [LSRH04]) is based on ISO/IEC 13250 Topic Maps [BBN99]. 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.

3.1 XSDS — XML Specification for Data Sources

Oveia supports the concept of extraction drivers. A driver extracts data from a data source and stores it in an intermediate 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 indicates which extraction driver will be used and provides values for the corresponding parameters.
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 specify the extraction of several *datasets*.

3.2 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 the 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 members 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.

3.3 Dataset Extractor

The *Dataset Extractor* is a processor that scans the input data sources 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 a specific type of source. The driver uses the appropriate technology to make the connection (e.g. JDBC Java DataBase Connectivity for databases, and an XML parser for annotated documents), and then the extraction of data is expressed 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 extracted data is stored in the *datasets*.

3.4 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 intermediate representation.
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 types, 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 intermediate representation.

The XS4TM Context Free Grammar is based in XTM 1.0 [PM01]. The *ontology* and *instances* elements have the same syntax as 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 an ontology in XS4TM.

3.5 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 processors 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.

3.6 Oveia Output

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 an 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 [WBD+00]. 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.

4 XTche – A Topic Map Semantics Validator

This section introduces the semantic validation issue in the area of Topic Maps. It presents the ideas that led to the development of a topic map constraint language (XTche [LRH04,RLH05]) and respective processor that guarantees semantic validity according to a specification.

4.1 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) [NM03]. 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. At the end the textual output of that edition (XML Schema code) should be processed to obtain a TMValidator.

4.2 XTche Processor and TM-Validator

A XTche specification, listing all the conditions (involving topics and associations) that must be checked, specifies the Topic Map 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 in order to verify its correctness.

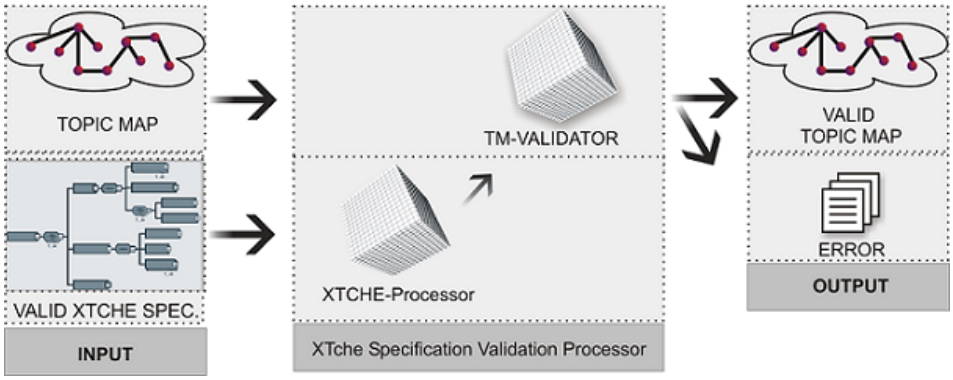

Fig. 3. XTche Validation Process

Both XSL stylesheets (the generator and the validator) are processed by a standard XSL processor like Saxon[1], what in our opinion is one of the benefits of the proposal.

5 Ulisses

Ulisses can be seen as a website generator from a XTM document (the "source" topic map) — this explains why we decided to integrate it as the last layer of Metamorphosis. It was conceived to be a autonomous (it can be used outside of Metamorphosis context) and simple way of creating full sites, with design, content and topical links; however, the layout of the site generated can be customized (page design, colors, ...) to satisfy the specific user needs. Allowing the navigation on a conceptual network (an ontology described by the source topic map), Ulisses can be seen as a useful tool to develop the so called *semantic web*.

[1] `http://saxon.sourceforge.net/`

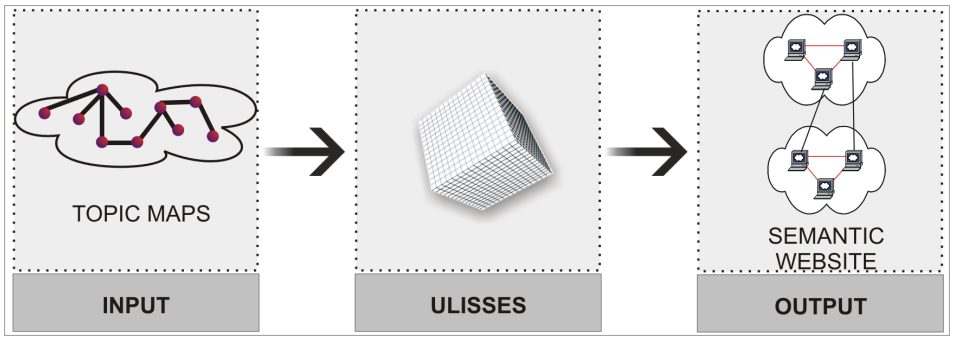

Fig. 4. Ulisses Architecture

The basic idea behind the website generation is to create one HTML page for each topic or association. Hyperlinks are then used to connect related topics or topics and associations. A navigation menu, allowing to go back to the home page or to choose another view of the topic map, is always present in every page.

A Topic page displays: the *topic name* and its *type* (for instance, `Manuel` of type `Emigrant`); the *topic occurrences*, organized by type, giving access to the respective information resources (for instance, `"Notable Men"` and `Almanac-X` are information resources of type `Book` listed as occurrences of `Manuel`, as well as `url1` is another occurrence, now of type `website`); the *related topics*, exhibiting for each case the role that the topic plays in the association responsible for the relation (for instance, `is-the-owner-of Palace-Y` describes the role that `Manuel` plays in the association that relates him to the topic `Palace-Y`); the *topic instances*, listing the topics that are instances of that topic (for example, `Manuel` is an instance of topic `Emigrant`.

An Association page displays: the *association name* and *type* (for instance, `owns` is an association of type `Emigrant-richness`); the *association members* and *roles*, organized by types, and exhibiting both the topics that are enrolled in that association (for instance, `Manuel is-the-owner-of Palace-Y` identifies two members of the association `owns` enhancing the role played by the topic of type `Emigrant`, while the pair (also shown) `Palace-Y is-owned-by Manuel` emphasizes the role played by the topic of type `House`).

As told above, each topic or association name displayed in one HTML page is a hyperlink to the respective page, thus implementing the conceptual navigation over the semantic network described by the topic map.

We developed three different versions of Ulisses: Ulisses I and II read the input from a XTM text file, while Ulisses III takes as input a *OntologyDB* (see above, sec. 3). Concerning the generation strategy, the original version (Ulisses I) is a *static generator*—it processes just once the XTM file and creates at that time all the website pages; the generation is time-consuming and the site directory huge, however the topic map navigation is very fast. The drawback of that approach

is that any change on the "source" TM implies the complete regeneration; otherwise the navigator becomes inconsistent/obsolete. To overcome that problem, the other two versions follow the opposite approach, implementing a *dynamic generation*; the first page (the homepage) is created at generation time and the others are created by need at navigation time.

6 Emigration Museum: a case study

During the last centuries a huge number of Portuguese people (women and mainly men) left the country to go away to work abroad. Until the middle of twentieth century, the most important destination for emigrants was Brazil (an old Portuguese colony and a very large and rich country). Becoming rich, many of them came back and did notable things with real social impact; they constructed manor houses and palaces, schools, hospitals, churches, factories, and they developed the industry and commerce. As there are plenty of documents and evidences about those emigrants and the outcomes of there lives, a group of Historians in Fafe (a town in the North of Portugal) decided to create a virtual museum devoted to the Brazilian Emigration; after a first prototyped (`www.museu-emigrantes.org`), we were involved in the conception of the information system.

The aim is to create a website that provides as many information as possible about each emigrant, and multiple navigation paths (offering various ways to handle the information) so that different views over the acquired knowledge are allowed. So, this museum on the Web should provide, not only data on individuals, but also knowledge about the social influence of their character and activities, in some geographical place at a certain date. To achieve that second, and main objective, it should be possible to cross data, exploiting the relations between the different information items (or units). Some interesting topics are: emigrant name; birth place and date; travel destination, departure and return dates, carrier; marital status; passport number; psychological profile; social or laboral event; industrial or commercial business; etc. Some important associations are: is; has; buy; creates; pays; offers; develops; etc.

However, as told above, the available resources, that should be exploited to extract the relevant data, are of many different kinds (official or technical records, literary documents, physical evidences, etc.), and are also available in different types of support: databases, annotated documents, and so on. For this case study we considered only three information sources: *travel diaries*, full of details written by the emigrant during the long (ship) trips; *biographical notes*, found in old almanacs, very rich in data concerning the character and social impact of the emigrant; *passport records*, obtained from the Portuguese foreign affairs bureau with factual data about travels. The first two are archived as XML documents (instances of two different document types), and the third one is a database.

In order to implement such an information system we could design a very large central repository, and impose that all the resources are consulted in order to extract the data to populate that huge database. Instead of that, we followed

a completely different approach. We decided to use Metamorphosis to keep the data sources as they are and to generate a website where the visitor can start by accessing a topic and then navigate over the knowledge following the relations included in the underlying ontology.
Oveia was fed with: (a) the XSDS structural and physical description of the XML documents (*travel diaries*, and *biographical notes*), and the database (*passport records*) to be parsed to extract topics; and (b) the XS4TM specification of the topic map to be built (notice that this TM corresponds to the ontology defined for the Emigration Museum). After 3,5 minutes, Oveia produced a 1,14MBytes (35588lines) XTM file containing a topic map with 1043 topics (instances of 25 topic types) and 1541 associations (instances of 32 association types). From a valid topic map, semantically checked by XTche, Ulisses generated a static website (a page for each topic or association, as explained in the previous section) that allows the user to browse the information accessing any item with the need to care about its origin. The main page, displaying the topic *emigrant*—that plays a role in 27 associations (of 12 different types)—is the most evident example of the knowledge integration achieved. The contextual constraints, specified in XTche, are mainly concerned with the *use of some topics* (that can only be *association roles* and nothing more), and with the cardinality of associations as well as the type of their members.

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, its validation against a set of rules, 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 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.
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.

As XTche specification language is based on XML Schema language, one of our next concerns is the implementation of the XTM-Skeleton-Extractor. The idea is to infer from the schema that specifies the constraints the basic specification of the Topic Map that we want to validate; this specification will be the skeleton that the user can complete to obtain the XS4TM specification (the second Oveia's input).

References

[BBN99] Michel Biezunsky, Martin Bryan, and Steve Newcomb. ISO/IEC 13250 - Topic Maps. ISO/IEC JTC 1/SC34, December 1999. `http://www.y12.doe.gov/sgml/sc34/document/0129.pdf`.

[LRH04] Giovani Rubert Librelotto, José Carlos Ramalho, and Pedro Rangel Henriques. XTche - A Topic Maps Schema and Constraint Language. In *XML 2004 Conference and Exposition*, Washington D.C., U.S.A, 2004. IDEAlliance.

[LSRH04] Giovani Rubert Librelotto, Weber Souza, José Carlos Ramalho, and Pedro Rangel Henriques. Using the Ontology Paradigm to Integrate Information Systems. In *International Conference on Knowledge Engineering and Decision Support*, pages 497–504, Porto, Portugal, 2004.

[NM03] Mary Nishikawa and Graham Moore. Requirements for a Topic Map Constraint Language JTC 1 NP Number. ISO/IEC 19756. ISO/IEC JTC 1/SC34 N0405, 2003. `http://www.y12.doe.gov/sgml/sc34/document/0405.htm`.

[PM01] Steve Pepper and Graham Moore. XML Topic Maps (XTM) 1.0. TopicMaps.Org Specification, August 2001. `http://www.topicmaps.org/xtm/1.0/`.

[RLH05] José Carlos Ramalho, Giovani Rubert Librelotto, and Pedro Rangel Henriques. Constraining Topic Maps: A TMCL declarative implementation. In *Extreme Markup Languages 2005*, Montreal, Canada, August 2005. IDEAlliance.

[WBD+00] Kevin Williams, Michael Brundage, Patrick Dengler, Jeff Gabriel, Andy Hoskinson, Michael Kay, Thomas Maxwell, Marcelo Ochoa, Johnny PaPa, and Mohan Vanmane. *Professional XML Databases*. Wrox Press, 2000.