

Comparing Topic Maps Constraint Specification Languages

Giovani Rubert Librelotto¹, Renato Preigschadt de Azevedo¹,
José Carlos Ramalho² and Pedro Rangel Henriques²

¹ UNIFRA, Centro Universitário Franciscano, Santa Maria, RS, 97010-032, Brasil
{librelotto,rpa.renato}@gmail.com

² Universidade do Minho, Departamento de Informática
4710-057, Braga, Portugal
{jcr, prh}@di.uminho.pt

Abstract. Topic Map Constraint Language (TMCL) provides a means to express constraints on topic maps conforming to ISO/IEC 13250. In this article, we will use a test suite and show, step-by-step, the way we handled several kinds of Topic Maps constraints in many different instances in order to answer questions like: Do they do the same job? Are there some kinds of Topic Maps constraints that are easier to specify with one of them? Do you need different background to use the tools? Is it possible to use them in similar situations (the same topic maps instances)? May we use them to produce an equal result? How do AsTMa!, OSL, Toma, and XTche relate to Topic Maps Constraint Language (TMCL)? What kind of constraints each one of these three can not specify? We will conclude this paper with a summary of the comparisons accomplished between those Topic Maps constraint languages over the use case proposed.

1 Introduction

Topic maps are an ISO standard for the representation and interchange of knowledge, with an emphasis on the findability of information. A topic map can represent information using topics (representing any concept), associations (which represent the relationships between them), and occurrences (relationships between topics and information resources relevant to them). They are thus similar to semantic networks and both concept and mind maps in many respects.

According to Topic Map Data Model (TMDM) [GM05], Topic Maps are abstract structures that can encode knowledge and connect this encoded knowledge to relevant information resources. On one hand, this makes Topic Maps a convenient model for knowledge representation; but on the other hand, this can also put in risk the topic map consistency. A set of semantic constraints must be imposed to the topic map in order to grant its consistency.

Currently, we can find three approaches to constrain Topic Maps – OSL [Gar04], AsTMa! [Bar03], Toma [Pin07], and XTche [LRH] – that allow us to specify constraints and to validate the instances of a family of topic maps against that set

of rules. With these resemblances it is easy to conclude that they are quite similar. However they differ in some fundamental concepts. These three Topic Maps constraint specification languages were thoroughly tested and benchmarked with a huge test suite. The most significant results will be discussed in this paper. The paper is organized as follows. In the next section, the Topic Maps Constraint Languages are introduced. The used case study – an e-Commerce corporation – is introduced in section three. Section four presents the comparison among the main constraint languages. Finally, conclusions are given in section five.

2 Topic Maps Constraint Language (TMCL)

Given a specification, a constraint is a logical expression that restricts the possible values that a variable in that specification can take.

A domain specific language enables the description of constraints required by each problem in a direct, clear and simple way; moreover it allows the derivation of a program to automatize the validation task. The derived semantic validator will verify every document, keeping silent when the constraints are satisfied, and reporting errors properly whenever the contextual conditions are broken.

The language to define topic map constraints is called as Topic Map Constraint Language. This language is currently on its way for standardization (ISO 19756 [NMB04]). The objective of TMCL is to allow formal specification of rules for topic map documents. TMCL has a similar purpose as schema languages for relational databases or XML applications. The constraint language is required to formalize application of specific rules. Currently there are different proposed constraint languages that will be presented in the next subsections.

2.1 XTche language

XTche is a process for specifying constraints on topic maps with a constraint language. This language allows to express contextual conditions on classes of Topic Maps. With XTche, a topic map designer defines a set of restrictions that enables to verify if a particular topic map is semantically valid.

XTche is an XML Schema oriented language [DGM⁺01]. This fact brings two benefits: on one hand it allows for the syntactic specification of Topic Maps (not only the constraints); and on the other hand it enables the use of an XML Schema editor (for instance, XMLSpy³) to provide a graphical interface and the basic syntactic checker.

The constraining process is composed of a language and a processor [LRH]. The language is based on XML Schema syntax. The processor is developed in XSLT language. The XTche processor takes a XTche specification and it generates a particular XSLT stylesheet. This stylesheet can validate a specific topic map (or a set of them) according to the constraints in the XTche specification.

XTche language meets all the TMCL requirements [NMB04]; for that purpose, XTche has a set of constructors to describe constraints in Topic Maps. But the

³ <http://www.altova.com>

novelty of the proposal is that the language also permits the definition of the topic map structure in an XML Schema style. An XTche specification merges the schema (defining the structure and the basic semantics) with constraints (describing the contextual semantics) for all the topic maps in that family.

2.2 AsTMa! language

AsTMa! [Bar03] is another language for constraint topic maps aiming to validate topic maps against a given set of rules. AsTMa! is a member of AsTMa* family (which includes AsTMa= for authoring TM, and AsTMa? for querying TM) and exposes some features of TMCL, because it has written earlier than the final version of the TMCL.

Resembling XTche the constraining process is composed of a language and a processor. The language is based on the Perl language, and the processor is written in Perl. At this time the AsTMa! Language is no longer maintained, because the author is working on a completely new distribution. So for this article we assume the AsTMa! Language definition for expression evaluation.

2.3 OSL language

According to the Ontopia Schema Language specification [Gar04], OSL has been designed to have a minimal number of features available on TMCL and a minimum expressive power.

Basically the OSL language constraints only the structure of a Topic Map. An OSL schema consists of a set of topic and association class definitions. These class definitions constrain the structure of the instances of the classes, and so control the form information may take in a topic map that uses the schema [Gar04].

As the other two languages OSL is composed by a language and a processor. The language is based on XTM [PH03] and the processor is written in Java, available for running standalone or as a plug-in for Ontopia Omnigator [Ont02].

2.4 Toma language

Toma [Pin07] is a TMCL language with a syntax very similar to SQL. It is based on path expressions that allow to access all elements in the topic map. This language has constructors like SELECT, UPDATE, INSERT, and DELETE; this constructors can be used to query and manipulate the topic map.

This language offers the MERGE statement to enable topic maps fusion. Also it offers the EXPORT statement to export the topic map to XTM. Toma provides functions that allow to modify, convert and aggregate data coming from the topic map. So, this language can be used as a Topic Maps Query Language (TMQL) and Topic Maps Manipulation Language (TMML).

3 Case study – E-Sell Corporation

A list of requirements for the new language was established by the ISO Working Group – the ISO JTC1 SC34 Project for a TMCL [NMB04]. Part of this document, consists in a section that presents a case study for a language to constraint any topic map. This case study is about an e-Commerce application.

For this use case, we created a topic map that stores information about customers, products and orders made by customers.

From that we will formalize the design decisions we have taken, and specify its vocabulary and type system (taxonomy). After that we will add application specific rules in AsTMa!, XTche, Toma, and OSL, the constraining languages.

The E-Sell’s Ontology: The objective of the ontology is to define sets of vocabularies along with its meaning that will be used within the framework. Rules or constraints also need to be defined to ensure the rigidity of the Topic Map framework that is used. This ensures that the information contained within the document is valid.

From the product class we derive subclasses which are the categories of products like: beverage, technology, and clothing. Some topics of type ”product” like: wine, radio, television, DVD, and phone are created. Another topic that needs to be covered as a class is the customer. From this class we derive subclasses which are the different customer categories like person and company.

Figure 1 shows a graph that represents a small part of E-Sell’s ontology on Vizigator [Gen]. This figure presents main topic types (order, product, and customer), the other topic types (person, company, technology, and beverage), and the topic instances (order 01, radio, Ronnie Alves, ...).

The links in that figure show the relationship between topic type and topic instances (beverage and wine, for instance) or association between two (or more) topics (for instance, order 05 is composed of DVD, radio, television, and wine).

4 Comparing the Topic Maps Constraint Languages

In this section we will compare the three languages briefly described previously in this paper. Then we point out advantages and disadvantages of eachone.

4.1 Do you need different background to use the languages?

Yes. To use the XTche language, the topic map designer needs to have solid understanding about XML, XML Schema, XSLT, and XPath [CD99]. All XTche specifications are in XML Schema format, so the designer can use a visual tool to write the constraints. The constraint can be written in any text editor, but it has the complexity of a common XML Schema.

To specify AsTMa! constraints, the designer is required to know the AsTMa! particular syntactic. To run the AsTMa! processor, it is necessary to have Perl and Prolog compilers installed.

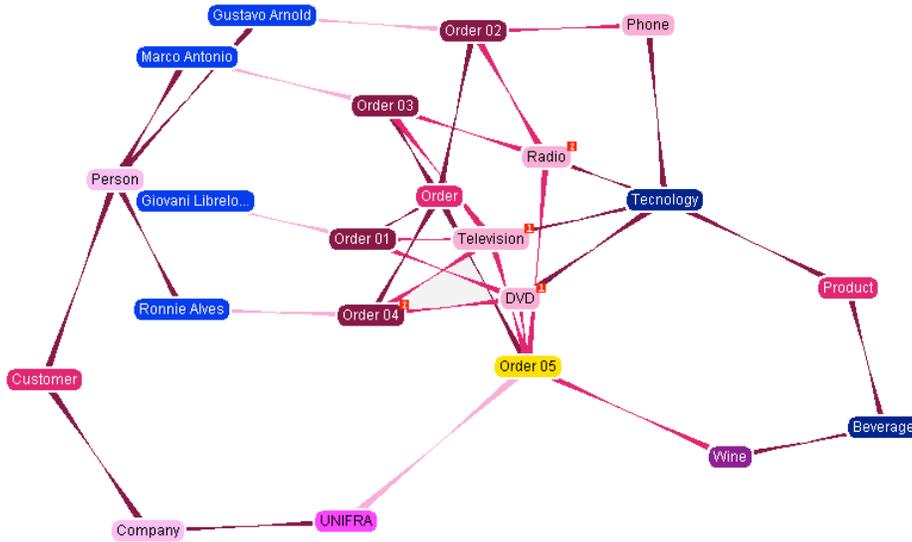


Fig. 1. The E-Sell Corporation's ontology

Toma language is SQL-based. It also takes some ideas from Object Orientation notation, Tolog language, and AsTMa* syntax.

OSL language is XTM-based, so the designer needs to specify this kind of constraints in agreement with XTM elements. The OSL tool only requires support for the Java language. Another way to execute OSL verifications is running it on Omnigator [Ont02].

4.2 Do they do the same job?

Not really. To illustrate this subject, we will present a few comparisons among these languages.

Validating generic topic map structure: In the first example, XTche, OSL, Toma, and AsTMa! languages virtually do the same job. These three languages allow to verify if a topic map (or a family of topic maps) has some inconsistency in agreement with a set of rules about its structure and content.

For instance, the association *is-making-order* represents each product line. This creates a relationship between a particular *product* and an *order*, along with the *quantity* of the product ordered. It means that an association of type *is-making-order* must have three association roles: *product*, *order*, and *quantity*. The code below shows the AsTMa! specification:

```
forall $a [ (is-making-order) ] => exists $a [ (is-making-order)
  product: *
  quantity: *
  order: * [
```

Figure 2 shows a graphical representation of this constraint specified in XTche.

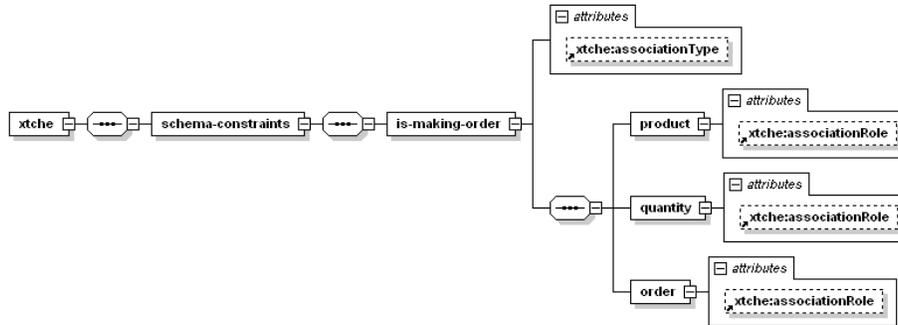


Fig. 2. XTche specification for an association structure

According to the Toma language, the same constraint is specified as shown below:

```
define constraint is_making_order_constraint
  each association $a(is-making-order)
  satisfies exists $a(is-making-order)->product = $$
    and $a(is-making-order)->quantity = $$
    and $a(is-making-order)->order = $$;
```

In OSL language, the same specification is:

```
<association>
  <instanceOf> <internalTopicRef href="#is-making-order"/> </instanceOf>
  <role min="1" max="1">
    <instanceOf> <internalTopicRef href="#product"/> </instanceOf>
    <player> <internalTopicRef href="#product"/> </player>
  </role>
  <role min="1" max="1">
    <instanceOf> <internalTopicRef href="#order"/> </instanceOf>
    <player> <internalTopicRef href="#order"/> </player>
  </role>
  <role min="1" max="1">
    <instanceOf> <internalTopicRef href="#quantity"/> </instanceOf>
    <player> <any/> </player>
  </role>
</association>
```

Validating a specific topic map structure: In the second example, the constraint is also about the association *is-making-order* where we need to ensure that a topic instance of *product* plays the role *product* and a topic instance of *order* plays the role *order*.

The code below introduces the AsTMA! function called *exists*:

```
forall $a [ (is-making-order) ]
=> exists $a [ (is-making-order)
  product: $p
  quantity: *
  order: $o [
  and
```

```

exists [ $p (product) ]
and
exists [ $o (order) ]

```

In XTche, the same specification would be:

```

<xs:element name="schema-constraints">
  <xs:complexType> <xs:sequence>
    <xs:element name="is-making-order">
      <xs:complexType> <xs:sequence>
        <xs:element name="product">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="product">
                <xs:complexType>
                  <xs:attribute ref="xtche:associationPlayer"/>
                  <xs:attribute ref="xtche:topicType"/>
                </xs:complexType>
              </xs:element>
            </xs:sequence> <xs:attribute ref="xtche:associationRole"/>
          </xs:complexType>
        </xs:element>
        <xs:element name="quantity">
          <xs:complexType>
            <xs:attribute ref="xtche:associationRole"/>
          </xs:complexType>
        </xs:element>
        <xs:element name="order">
          <xs:complexType> <xs:sequence>
            <xs:element name="order">
              <xs:complexType>
                <xs:attribute ref="xtche:associationPlayer"/>
                <xs:attribute ref="xtche:topicType"/>
              </xs:complexType>
            </xs:element>
          </xs:sequence> <xs:attribute ref="xtche:associationRole"/>
        </xs:complexType>
      </xs:element>
    </xs:sequence> <xs:attribute ref="xtche:associationType"/>
  </xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>

```

The diagrammatic view of this schema (and the next ones too) can be generated by any XML Schema editor, so we will not show them in the paper.

Toma language defines this constraint like the following:

```

define constraint is_making_order_constraint_testing_product
  each association $a(is-making-order)
  satisfies exists $a(is-making-order)->product = $p
    and $a(is-making-order)->quantity = $$
    and $a(is-making-order)->order = $o
    and $p->type = 'product'
    and $o->type = 'order';

```

Unfortunately, OSL language does not allow to specify this kind of constraint.

Data types: According to the TMDM [GM05], Topic Maps do have a concept of data and data types, but there is no commitment to any set of primitives such as XML Schema (XSD) [DGM⁺01]. That may be a good move, since XSD

is – like any other set – quite arbitrary. Useful, but arbitrary. So, if a topic map designer wants to validate an age occurrence as a number, he needs to use a constraint language.

The only way to constrain text in AsTMa! is to use regular expressions. For instance, to allow the invocation of "boolean test functions", such as:

```
in (age): ?is_age()
```

The AsTMa! validator would call this function (implemented externally). According to its creator, this issue would have to be addressed if AsTMa! evolves into a new version.

XTche specification below tests if a person type topic has an age occurrence of integer type (any XSD data type can be used in a XTche specification).

```
<xs:element name="schema-constraints">
  <xs:complexType> <xs:sequence>
    <xs:element name="person">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="age">
            <xs:complexType>
              <xs:simpleContent>
                <xs:extension base="xs:integer">
                  <xs:attribute ref="xtche:occurrenceType"/>
                </xs:extension>
              </xs:simpleContent>
            </xs:complexType>
          </xs:element>
        </xs:sequence> <xs:attribute ref="xtche:topicType"/>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
</xs:element>
```

Toma has functions to convert parameters. The function *to_num* converts text to a number. The function *to_unit* converts between units defined by *Units Conversion Library* by *Maio Foundation*⁴. However, this language does not allow the user to use all XSD data types as XTche does.

In terms of data type, OSL does not provide any kind of data type.

4.3 Where each language shows its strength

The topic maps constraints about topics and association structures are easier to specify in these four languages. For instance, the constraint "customer must have a contact number which is either a phone or a fax number" is specified in AsTMa! like this:

```
forall $c [ * (customer) ]
=> exists $c [ in (phone): * ]
  or
  exists $c [ in (fax): * ]
```

According to the XTche language, the respective specification is below.

⁴ http://sourceforge.net/docman/display_doc.php?docid=2810&group_id=19449#SECTd0e169

```

<xs:element name="schema-constraints">
  <xs:complexType> <xs:sequence>
    <xs:element name="customer">
      <xs:complexType>
        <xs:choice>
          <xs:element name="phone">
            <xs:complexType>
              <xs:attribute ref="xtche:occurrenceType"/>
            </xs:complexType>
          </xs:element>
          <xs:element name="fax">
            <xs:complexType>
              <xs:attribute ref="xtche:occurrenceType"/>
            </xs:complexType>
          </xs:element>
        </xs:choice> <xs:attribute ref="xtche:topicType"/>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
</xs:element>

```

The code below shows the Toma specification for this kind of constraint:

```

define constraint customer_must_have_a_contact_number_constraint
  each topic $t
    where $t.type = 'customer'
    satisfies exists $t.oc.id = 'phone'
      or $t.oc.id = 'fax';

```

In other hand, OSL correspondent code is presented below. However, this language have a limitation: it does not work with boolean operations. So the constraint "either a phone or a fax number" is not supported.

```

<topic>
  <instanceOf>
    <internalTopicRef href="#customer"/>
  </instanceOf>
  <occurrence min="0" max="1">
    <instanceOf>
      <internalTopicRef href="#phone"/>
    </instanceOf>
  </occurrence>
  <occurrence min="0" max="1">
    <instanceOf>
      <internalTopicRef href="#fax"/>
    </instanceOf>
  </occurrence>
</topic>

```

The code above defines a topic instance of customer that has zero or one phone occurrence and zero or one fax occurrence. But, according to this OSL specification, there is no way to verify if a topic instance of customer has both occurrences.

4.4 Is it possible to use them in similar situations (the same topic maps instances)?

It is possible to use them in several similar situations but it is important to care about the topic map format. XTche language only processes topic maps in

XTM format. There is a small project in the XTche context to create a processor that converts other topic maps formats – LTM (Linear Topic Map) [Gar02] and HyTM (HyTime Topic Maps) [NBB03] – to XTM.

In the same perspective, AsTMa! language only processes topic maps that are in AsTMa= format.

Toma can not generate TM or XML content. Toma assumes that users that want to create applications using Toma will use in addition other technologies (Java, Perl, Python, etc.). Each of those technologies provide sets of techniques and methodologies to create XML content as well as any other content.

Talking about OSL, this language is part of Omnigator tool [Ont02]. Many Topic Maps formats can be validated according to a set of OSL rules. Ontopia enables the navigation over the following topic map formats: XTM, LTM, and HyTM; ontologies in RDF (Resource Description Framework) [LS99] format can be navigated by Omnigator too.

4.5 May we use them to produce an equal result?

Maybe. The answer is *Yes* if the topic map designer wants to validate the topic map schema because these three languages confirm the validity of a topic map instance across a set of rules. The answer is *No* if the topic maps designer wants to validate the topic map with particular constraints, like existence, boolean, and conditional constraints. In this case, XTche, Toma, and AsTMa! have constructors to specify that; OSL has not.

For example, the constraint "for all topic that has the topic type customer, it must have a basename (for customer name), an occurrence (for address), a subject identifier (for customer id), and optionally additional occurrence (for email address)" [NMB04] can be constrained in XTche, AsTMa!, Toma, and OSL. The result for all these languages is a list of the topics that are not conformed with this rule. If all the topics conforms this rule, the result is the topic map validation confirmed. So, for this case: *Yes*, we may use them to produce an equal result.

However another constraint example: "for all association of is-making-order type, it must have the association roles customer and order played by the topic that is of type customer and order respectively" [NMB04] can be validated by Toma, XTche, and AsTMa! languages, and can not be validated by OSL language. Thus, for this case: *No*, we may not use them to produce an equal result.

4.6 How do AsTMa!, OSL, Toma, and XTche relate to Topic Maps Constraint Language (TMCL)?

Toma, XTche, and AsTMa! languages are based on a draft version of the TMCL, so they are able to specify almost any kind of constraint suggested by TMCL requirement. Toma and AsTMa! do not have constructors to constrain data types.

Toma, AsTMa! and XTche have constructors to make complex conditional, boolean, and existential constraints. On the other hand, OSL does not have

relationship with TMCL, and it was defined to make just simple validations in a topic map. So the language does not have boolean, existential, and conditional operators, becoming a real alternative only in simple and small projects.

OSL was not designed on the basis of TMCL requirements; it is intended only for validating the topic maps structures. For instance, OSL can not specify the following constraint: "topic radio can not be used to scope association".

5 Conclusion

This paper showed a comparison among the three TMCL-based languages – AsTMa!, OSL, Toma, and XTche – over several kinds of Topic Maps constraints in many different instances. We started with our strong motivation to check a topic map for syntactic and semantic correctness - as a notation to describe an ontology that supports a sophisticated computer system (like the applications in the area of Semantic Web or archiving) its validation is crucial!

In order to compare these languages, we succeeded in applying a case study – E-Commerce Application (subsection 6.1 of TMCL Requirements [NMB04]) – virtually representative of all possible cases. This means that: on one hand, we were able to describe the constraints required by each problem in a direct, clear and simple way; on the other hand, the Topic Maps semantic validator could process every document successfully, that is, keeping silent when the constraints are satisfied, and detecting/reporting errors whenever the conditions are broken.

Doing a comparison among these languages, some advantages of XTche emerge: (1) XTche has a XML Schema-based language, a well-known format; (2) XTche allows the use of an XML Schema graphic editor, like XMLSpy. In a diagrammatic view, it is easy to check visually the correctness of the specification; (3) XTche gathers in one specification both the structure and the semantic descriptions, and it realizes a fully declarative approach requiring no procedural knowledge for users.

The main problem about XTche is the size of this code. If a topic map designer does not have a XML Schema editor, the specification is too complex in a comparison with other languages. This XTche problem is a Toma and AsTMa! advantage: the size of AsTMa! constraints are small, very similar to regular expressions and SQL, respectively.

In a related work, Eric Freese [Fre02] says that it should be possible to use the DAML+OIL language to provide a constraint and validation mechanism for topic map information. The cited paper discusses how to describe validation and consistency of the information contained in Topic Maps using DAML+OIL and RDF, showing how to extend XTM and how to define PSIs and class hierarchies, as well as to assign properties to topics.

The main conclusion is that XTche, Toma, and AsTMa! comply with almost all requirements stated for TMCL whereas OSL just includes topic maps structure validation.

References

- [Bar03] Robert Barta. AsTMa! Bond University, TR., 2003. <http://astma.it.bond.edu.au/constraining.xsp>.
- [CD99] James Clark and Steve DeRose. XML Path Language (XPath) - Version 1.0. <http://www.w3.org/TR/xpath>, November 1999.
- [DGM⁺01] Jon Duckett, Oliver Griffin, Stephen Mohr, Francis Norton, Nikola Ozu, Ian Stokes-Rees, Jeni Tennison, Kevin Williams, and Kurt Cagle. *Professional XML Schemas*. Wrox Press, 2001.
- [Fre02] Eric Freese. Using DAML+OIL as a Constraint Language for Topic Maps. In *XML Conference and Exposition 2002*. IDEAlliance, 2002. http://www.idealliance.org/papers/xml02/dx_xml02/papers/05-03-03/05-03-03.html.
- [Gar02] Lars Marius Garshol. LTM – The Linear Topic Map Notation. Ontopia, 2002. <http://www.ontopia.net/topicmaps/ltm.html>.
- [Gar04] Lars Marius Garshol. The Ontopia Schema Language – Reference Specification. <http://www.ontopia.net/omnigator/docs/schema/spec.html>, 2004.
- [Gen] Pamela Gennusa. Ontopia’s Vizigator(tm) - Now you see it! In *XML 2004 Conference and Exposition*, Washington D.C., U.S.A. IDEAlliance.
- [GM05] Lars Marius Garshol and Graham Moore. Topic Maps – Data Model. In *ISO/IEC JTC 1/SC34*. <http://www.isotopicmaps.org/sam/sam-model/>, January 2005.
- [LRH] Giovanni Rubert Librelotto, José Carlos Ramalho, and Pedro Rangel Henriques. Constraining topic maps: a TMCL declarative implementation. In *Extreme Markup Languages 2005: Proceedings*. IDEAlliance.
- [LS99] Ora Lassila and Ralph R. Swick. Resource Description Framework (RDF) Model and Syntax Specification. World Wide Web Consortium, February 1999. <http://www.w3.org/TR/REC-rdf-syntax>.
- [NBB03] Steven R. Newcomb, Michel Biezunski, and Martin Bryan. The HyTime Topic Maps (HyTM) Syntax 1.0. ISO/IEC JTC 1/SC34 N0391, 2003. <http://www.jtc1sc34.org/repository/0391.htm>.
- [NMB04] Mary Nishikawa, Graham Moore, and Dmitry Bogachev. Topic Map Constraint Language (TMCL) Requirements and Use Cases. ISO/IEC JTC 1/SC34 N0548, 2004. <http://www.jtc1sc34.org/repository/0548.htm>.
- [Ont02] Ontopia. The Ontopia Omnigator, 2002. <http://www.ontopia.net/omnigator/>.
- [PH03] Jack Park and Sam Hunting. *XML Topic Maps: Creating and Using Topic Maps for the Web*, volume ISBN 0-201-74960-2. Addison Wesley, 2003.
- [Pin07] Rani Pinchuk. TopiWriter User Manual - Toma, 2007.