Ontology-driven Topic Maps

Ontology-driven topic maps

Bernard **Vatant** <bernard.vatant@mondeca.com>

Abstract

Both topic maps specifications and literature have implicitly or explicitly presented the standard as 'ontology-agnostic', meaning they are able to support, represent and manage any kind of knowledge in any kind of ontological context, and even independently of the constraints imposed by any ontology. Topic maps are able to express 'anything about anything whatsoever'. This very claim has attracted some criticisms from experts in formal ontologies, and among the Semantic Web community, sometimes interpreting it like a puff of smoke hiding poor formal foundation.

Beyond theoretical claims and academic debate, most if not all real-world topic map implementations and software use explicitly or implicitly schemas and/or ontologies, defining classes of topics, role types and association templates, occurrence type constraints and data structure... Such sets of constraints should eventually be expressed as topic map schemas, but the relevant specification, TMCL, based on Topic Map Data Model, has still a long way to go.

Meanwhile, rapid development and adoption of ontologies in a wide range of domains and industries, and singularly based on the new W3C Web Ontology Language OWL, have put the issue in a slightly new perspective. If OWL ontologies are to become effectively mainstream core technology for the Semantic Web, topic maps applications cannot afford any more sticking to the agnostic line, pretending to play in a somehow orthogonal field.

In fact topic maps would indeed gain effectiveness and interoperability either through explicit formalization of ontologies specifically built and dedicated for topic map control, or through declaration of commitment to pre-defined ontologies, not specifically designed for that use. In either case, using OWL ontologies should be considered as the most interesting choice.

Based on examples chosen from actual Mondeca ITM implementations of ontology-driven topic maps, the presentation will show the benefits of this approach, and their current limits.

Table of Contents

1. Introducing the partners : Topic Maps and Ontologies

1.1. Old story, new perspectives

While topic maps have been around for a while, ontologies have now come of age, and are bound to become the backbone of emerging Semantic Web technologies in the years to come. How can ontologies and topic maps live together? Since the new W3C recommendation, the Web Ontology Language OWL [OWL], is built on top of RDF, one could think that achieving semantic and syntactic interoperability between topic maps and RDF will somehow entail automatically interoperability between topic maps and OWL-RDF ontologies. Actually, many debates and publications in the recent years have raised in Semantic Web and topic map communities about comparison and interoperability of the two technologies, and it has been a recurrent subject in XML conferences.

A good reference is Lars Marius Garshol's 2003 paper [GA03] which sums up in great detail similarities and differences between RDF and topic maps families of standards at the data model level. As the title of Garshol's paper says very well, we have to live now with both technologies around, they are neither equivalent nor really competiting, and they should be widely interoperable. If no standard interoperability framework is yet available, it's more out of process issues than technical ones, the main one being the fact that they have been developped by different standard organizations, namely W3C and ISO. There have been recently interesting steps in the good direction. The new W3C Semantic Web Best Practices and Deployment Working Group, set up in March 2004, has considered among many issues to tackle, integration of topic maps legacy in the Semantic Web big picture. At that point, this issue is not on the top priority of the WG and no formal task force has been dedicated to it so far. But it is on the backburner [SWBP].

Without pre-empting whatever conclusion of this process, this paper addresses a very specific aspect of a possible interoperability framework, the capacity to use OWL to express the ontology layer in a knowledge organization system using topic maps. The purpose is not again mapping RDF and Topic Map data models, but to show :

- Why real-world topic maps need, and generally are driven by, implicitly ontologies.
- How those ontologies can be explicited using OWL.
- How XTM files can reference directly elements of such ontologies.
- The trade-off between expressivity and formal constraint which is at stake in such a choice.

We'll discover in the process some interesting insights on what data types are or can be, and how they fit in this framework.

1.2. Theoretical view : topic maps are ontology-agnostic

Topic maps have been designed to be deliberately 'ontology-agnostic', in the sense that they are intended to be able to represent and manage any kind of subjects and relationships between them, in any kind of ontological context. Better, topic maps would be able to represent any kind of ontological framework.

Although it reflects the unbounded expressivity that is a major quality acknowledged by topic maps users and designers, this very claim has attracted criticism from experts in semantics, formal logic and ontologies, and among the Semantic Web community, sometimes interpreting it as a puff of smoke hiding poor formal foundation ... although this capacity to 'say anything about anything' is also claimed by RDF. But it's true that the formal foundation and conceptual model of topic maps remain a subject of debate - not to say a bone of contention - inside the topic map community itself.

1.3. Pragmatic view : Real-world topic maps are ontology-driven

But when one looks closely at any topic map used in a real-world enterprise application, one will find an organization showing all the symptoms of the presence of some underlying, implicit if not explicit, ontology. Classes of topics, association types, role types, occurrence types, seem carefully chosen, and one will find out some sensible and recurrent patterns linking association types to specific role types, occurrence types to topic types, role types in association types to topic classes, implicit rules of cardinality, etc.

So one naturally wonders if those implicit ontologies should not be better off explicited and formalized, and for such a formalization to be used in a Semantic Web context, using OWL should of course be considered.

1.4. But is this not what TMCL is all about?

Indeed, requirements for future TMCL [TMCL] include capacity to express 'ontological constraints' such as those quoted above, and so do existing TM schema languages such as OSL [OSL] and AsTMa! [ASTMA]. Why not use those, then? An obvious point is that none of those languages has achieved so far the status of standard recommendation as OWL has, the standard TMCL being yet in early stages of development, and with a long requirements list. Another point is that TMCL will be dedicated for exclusive use against the Topic Map Data Model, and will be able to control a lot of specific topic map features that are not addressed in this paper. All the point of using OWL is to leverage the considerable work made to develop this language, and the strength of the formal model underlying it. It does not preclude any further fine-grained constraining that TMCL could bring about. Moreover, as we'll see now, we address here the use of topic maps in a specific context of layered knowledge organization systems, where topic maps are integrated at the level where they fit best and have their best potential : the representation of knowledge bases.

2. Layered KOS, Ontology and Knowledge Base

2.1. Introducing layered Knowledge Organization Systems

A real-world topic map of the kind mentioned above, on which we will focus now, will be considered hereafter as part of a Knowledge Organisation System. We will use this acronym KOS to refer to systems using an explicit representation of 'individual entities' (real-world objects and/or abstract concepts), a formal or implicit organization of those entities through classes and relationships, used to index or classify data, documents, information pieces, attached to the entities in a meaningful way.

Former analysis of use of topic maps in KOS can be found in various chapters of 'XML Topic Maps' book. [PA03]Whether one uses topic maps or not, it is a common and sensible (not to say 'best') practice to structure a KOS in three clearly distinct layers : the ontology layer, the knowledge base layer, and the data layer. The proposed architecture is making the following choice of languages, using each one where it seems to fit the best.

- OWL to represent the ontology layer, constraining the two other layers.
- XTM to represent the knowledge base layer, driven by classes, types and constraints defined in the ontology.
- XML schemas (in the broadest sense) to structure complex fine-grained data.

We'll now go in more details, based on an example of a KOS representing a community of practice : people, organizations they work in, domains of expertise and interest, projects they are involved in, technologies used by those projects, indexing relevant on-line information resources documenting all the above. We'll address in this section only the ontology and knowledge base layers.

2.2. The Ontology Layer

The ontology layer contains and defines types, or classes, used to organize the two other layers. In a topic map KOS, the ontology layer will contain topic types, association types, role types, and occurrence types. In our example, topic types would be e.g. 'Person', 'Technology', 'Organization', 'Project', association types would be 'Employment', 'Project Team', 'Expertise', role types 'employer', 'employee', 'manager', 'participant', 'expert', and occurrence types 'cv', 'e-mail', 'homePage', 'beginDate' ...

Actually, the topic map standard does not require us to declare topics as types before using them as such, and actually gives no provision for such a declaration. A topic is considered a type (or class) as soon as some other topic, association, role or occurrence, is declared to be an instance of it, through 'instanceOf' declaration. Therefore, by parsing an XTM file and gathering all the 'instanceOf' tags, or by any other way based on implementations conformant to the TM Data Model [TMDM], it's quite easy to gather which topics are used as topic, association, role and occurrence types, and constitute the topic map 'implicit ontology'. This kind of description is for example what Ontopia's Omnigator does, e.g. for the famous opera topic map [OPTM]. But it is difficult this way to capture more subtle ontological features, like patterns of implicit constraints, such as association templates linking association type, role types and classes of role players, that are certainly present implicitly in the intention of the topic map author.

So, it is certainly abusive to claim that the ontology layer of a topic map (or RDF) file could be defined 'a posteriori'. Certainly, resources used as classes can be found and represented in an useful structured way, like the above example shows. But calling such a representation an 'ontology' could be confusing for the casual user, since classes and types have not been explicitly declared or represented anywhere before creation of the file, and neither control nor constrain this creation. To appreciate the difference between 'a posteriori' description and 'a priori' declaration, think about DTD or schemas created on the fly by some XML tools out of any random XML file with no schema declaration, against XML files created and validated against pre-defined, and hopefully well-thought, schemas.

If one now compares the situation in RDF, it's quite the same. Any RDF resource can be considered an implicit class as soon as some other resource uses it as such through an 'rdf:type' property. But RDFS and OWL provide the means for declaring that a resource is a class before populating it with instances, and OWL-DL demands that this declaration be explicit before declaring instances. The same kind of approach can be applied to topic maps, as we will see later on : what OWL brings to RDF, it can also bring to topic maps.

2.3. The Knowledge Base Layer

The knowledge base layer contains instances of the types defined in the ontology layer. In a topic maps KOS, they are topics, roles, associations and occurrences, linked to the ontology layer topics by 'instanceOf' relationships. For example 'Bernard Vatant', 'Mondeca', 'ITM' are instances of topics, and 'Employment (employer: Mondeca, employee: Bernard Vatant)' an instance of association. The notation used might be understood in RDF as well as in XTM, with the same semantics, the XTM association being equivalent to two RDF triples 'Employment, employer, Mondeca' and 'Employment, employee, Bernard Vatant'.

A layered KOS requires that the ontology layer and knowledge base layer are totally disjoint, and in particular that classes and instances to be different types of objects (like in OWL-DL). Topic maps, like RDF (or OWL-Full), have no such requirement : the same TM topic, or RDF resource, can be used both as class in the ontology and instance in the knowledge base (or, in TM, as association type and class, or class and role type ...)

In ontologies for topic maps, a sensible additional requirement would be to have distinct topics defining topic types, association types, role types and occurrence types. Those could seem too strong requirements leading to a loose of expressive power of topic maps. It might be, but the same kind of trade-off between expressivity and formal power has led to the distinction between OWL-DL and OWL-Full (or RDF). What we propose hereafter is along the same lines, and in fact will naturally lead to propose to use OWL-DL to declare ontology layers in layered topic maps. The parallel with OWL and RDF should make it clear that we don't propose here that *any* topic map should be structured that way. Having RDFS and OWL to constrain RDF does not prevent anyone to use RDF "freely" without OWL.

2.4. An OWL ontology for a TM knowledge base

This section describes on an example how to declare in OWL topic types (classes), association types and role types, and constraints linking them. We'll assume the existence of a namespace tm: at http://www.ex.org/, defining the needed generic topic maps classes and properties.

2.4.1. Defining Topic types

Defining topic types, or classes is quite straightforward using owl:Class. Nevertheless, owl:Class will also be used later on to define association types, so we'll declare explicitly a generic class tm:Topic as a direct subclass of owl:Thing. A topic type is defined as a subclass of tm:Topic. For example we define the classes 'Organization' and 'AffiliatedPerson' as following

```
<owl:Class rdf:about="http://www.example.org/ontology#Organization">
   <rdfs:subClassOf rdf:about="http://www.ex.org/Topic"/>
</owl:Class>
```

```
<owl:Class rdf:about="http://www.example.org/ontology#AffiliatedPerson>
   <rdfs:subClassOf rdf:about="http://www.ex.org/Topic"/>
</owl:Class>
```

2.4.2. Defining and constraining Association and Role types

We are now showing how to define association types and attached role types. There is of course no explicit notion of 'association' and 'role' in RDF and OWL, that's why 'tm:Association' is supposed to be declared the same way as tm:Topic, and an association type is declared as a subclass of tm:Association.

```
<owl:Class rdf:about="http://www.example.org/ontology#Employment">
   <rdfs:subClassOf rdf:about="http://www.ex.org/Association"/>
</owl:Class>
```

Role types are defined as ObjectProperty on Association subclasses. The following defines two role types 'employer' and 'employee' with domain the previous association type 'Employment', and with constraints on range, specifying that the 'employer' must be an instance of 'Organization' and 'employee' an instance of 'AffiliatedPerson'

```
<owl:ObjectProperty rdf:about="http://www.example.org/ontology#employer">
   <rdfs:subPropertyOf rdf:about="http://www.ex.org/associationRole"/>
   <rdfs:domain rdf:about="http://www.example.org/ontology#Employment"/>
   <rdfs:range rdf:about="http://www.example.org/ontology#Organization"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="http://www.example.org/ontology#employee">
   <rdfs:subPropertyOf rdf:about="http://www.ex.org/associationRole"/>
   <rdfs:domain rdf:about="http://www.example.org/ontology#Employment"/>
   <rdfs:range rdf:about="http://www.example.org/ontology#AffiliatedPerson"/>
</owl:ObjectProperty>
```

Further restrictions can of course be defined, like value or cardinality restrictions. For example 'PublicEmployment' can be defined as a subclass of 'Employment', with the restrictions that the 'employer' role is played by exactly one instance of the class 'PublicOrganization'.

```
  <owl:Class rdf:about="http://www.example.org/ontology#PublicEmployment">
   <rdfs:subClassOf>
     <owl:Class rdf:about="http://www.example.org/ontology#Employment"/>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
     <owl:Restriction>
       <owl:onProperty>
         <owl:ObjectProperty rdf:about="http://www.example.org/ontology#employer"/>
       </owl:onProperty>
       <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</owl:cardinality>
     </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
     <owl:Restriction>
       <owl:allValuesFrom>
         <owl:Class rdf:about="http://www.example.org/ontology#PublicOrganization"/>
       </owl:allValuesFrom>
       <owl:onProperty>
         <owl:ObjectProperty rdf:about="http://www.example.org/ontology#employer"/>
       </owl:onProperty>
     </owl:Restriction>
   </rdfs:subClassOf>
  </owl:Class>
```

2.4.3. Knowledge Base Individuals

Individual instances of the previous classes can be defined in RDF. The following example is an instance of 'Employment', the namespace ex: being http://www.example.org/ontology#.

```
  <ex:Employment>
    <ex:employee>
      <ex:AffiliatedPerson rdf:about="http://www.example.org/kb#dupont"/>
    </ex:employee>
    <ex:employer>
      <ex:Organization rdf:about="http://www.example.org/kb#my-company"/>
    </ex:employer>
  </ex:Employment>
```

The XTM rendition of the same individual instance of 'Employment' association is much more verbose, but carries exactly the same semantics.

```
<topicMap>
 <topic id="dupont">
   <instanceOf>
     <subjectIndicatorRef xlink:href="http://www.example.org/ontology#AffiliatedPerson"/>
   </instanceOf>
   <subjectIdentity>
     <subjectIndicatorRef xlink:href="http://www.example.org/kb#dupont"/>
   </subjectIdentity>
 </topic>
 <topic id="my-company">
   <instanceOf>
     <subjectIndicator xlink:href="http://www.example.org/ontology#Organization"/>
   </instanceOf>
   <subjectIdentity>
     <subjectIndicatorRef xlink:href="http://www.example.org/kb#my-company"/>
   </subjectIdentity>
 </topic>
 <association>
   <instanceOf>
      <subjectIndicatorRef xlink:href="http://www.example.org/ontology#Employment"/>
   </instanceOf>
   <member>
      <roleSpec>
         <subjectIndicatorRef xlink:href="http://www.example.org/ontology#employer"/>
      </roleSpec>
      <topicRef xlink:href="#my-company"/>
   </member>
   <member>
      <roleSpec>
         <subjectIndicatorRef xlink:href="http://www.example.org/ontology#employee"/>
      </roleSpec>
      <topicRef xlink:href="#dupont"/>
   </member>
 </association>
</topicMap>
```

The benefit of reference to the ontology classes and properties for topic types, association types and role types is two-fold. The less constraining aspect is that elements of the ontology are used as non-ambiguous subject indicators, providing basic ontological commitment. At a more constrained level, reference to the ontology provides support for semantic validation of files to be imported in an user workspace, and/or control of a knowledge base editing interface. Both features are used now in Mondeca ITM, where user workspaces, navigation and edition interfaces are controlled by OWL ontologies.

To make the level of reference completely non-ambiguous, XTM files should be able to declare at the level of the <topicMap> element a formal commitment to the reference ontology (or ontologies) used, meaning that topic, association and role types used in the file should be valid against the types and constraints declared in the ontology. Syntax for such a declaration is easy to be defined, e.g through an optional <ontologyRef> child of <topicMap>.

3. The Data Layer

The data level is where interoperability seems the less obvious, because of different views of what 'data' and 'data types' are, that we'll try to clarify a bit by re-introducing notions that have been sadly overlooked or blurred by both RDF-OWL and topic maps: the notion of complex data types, and the distinction between logical and physical data type. What we propose here is the definition of occurrence types by both logical and physical types.

3.1. 'Logical' vs 'Physical' data types

Let's take an example again. We've mentioned above 'CV' as an usual piece of information attached to a person. In topic maps, one will use an occurrence to attach to a 'Person' topic a link to an external resource, hopefully yielding actually some kind of CV document for the Person. Let's take again the topic '#dupont' defined above, and let's add such a 'cv' occurrence.

```
<topic id="dupont">
  <instanceOf>
    <subjectIndicatorRef xlink:href="http://www.example.org/ontology#AffiliatedPerson"/>
  </instanceOf>
  <subjectIdentity>
    <subjectIndicatorRef xlink:href="http://www.example.org/kb#dupont"/>
  </subjectIdentity>
  <occurrence>
   <instanceOf>
     <topicRef xlink:href="http://www.example.org/ontology#cv"/>
   </instanceOf>
   <resourceRef xlink:href="http://www.example.org/dupont/cv.xml"/>
  </occurrence>
</topic>
```

3.1.1. Topic maps specify only logical types

The above is perfectly legal XTM whatever the 'meaning' of the resource at "http://www.example.org/ontology#cv", and whatever the resource available at "http://www.example.org/dupont/cv.xml". XTM does not make any provision for the actual type of data provided by both resources. In particular, there is no provision to constrain the attached resource to have a specific *physical type* : file extension, data schema, size ... The occurrence type is just a 'logical label' on a black box. This is a deliberate choice of the XTM specification, which did not want to mess up with datatypes. The drawback of this underspecification is that it makes it difficult to integrate a topic map engine with applications that would be able to make sense of the actual data content, validate it against a specific schema, unless those applications can somehow magically match the 'logical' occurrence type to some 'physical' type, by some built-in and ad hoc mechanism.

The ontological declaration defining http://www.example.org/ontology#cv should provide the kind of information needed. What mechanisms does OWL provide for such declaration, that's what we are going to see now.

3.1.2. OWL-RDF specify only simple physical types

Unfortunately, RDFS-OWL demands specification of data physical types at the lowest possible level of granularity, using the XML datatypes. An usual OWL practice to attach CV information to a person would typically detail fields usually captured in a CV document as so many properties with domain 'Person', either DatatypeProperty such as 'firstName', 'familyName', 'birthDate', or objectProperty such as 'placeOfBirth', 'formerEmployer', 'academicBackground', linking to other individuals in the knowledge base. What can be gained using such fine-grained information is obvious, but there are several severe drawbacks to it.

- It makes the ontology being over-detailed, and accordingly difficult to maintain and re-use. It will not be re-usable by someone with slightly different requirements for fields needed in a CV, although there will be general agreement on the relevancy of attaching a CV to a person.

- If one wants to extract all information that makes up the CV, it will have to go through very specific queries on the knowledge base, using all the relevant properties and sorting them out to recompose what will look like a CV to the end-user.

- The CV data are often already available in the information system, in structured documents built against specific data schemas, as XML schemas and DTDs, already managed by specific applications. The interest and cost of replicating such information in the ontology is something to consider.

3.2. Linking logical and physical complex data types

To cope with those difficulties, the mechanism proposed here is to defer the declaration details of CV data types (the physical data type) to existing external schema that applications integrated with the KOS can make sense of, by re-using available schemas in a modular way. Neither RDF nor topic maps make actual provision for that kind of mechanism, but workarounds can be found that make it possible to express it using the two languages. It goes through reification of complex data, that allows to explicitly make distinct the logical and physical data types, and as necessary bind the former to the latter.

In the CV example, the mechanism is to create a 'CurriculumVitae' subclass of some generic 'ComplexData' class, having the generic ObjectProperty 'schemaRef'. We assume again that those generic Class and ObjectProperty are defined in the tm: namespace, with non-ambiguous semantics. In the example, the schemaRef resource is a DTD, actually defining about 50 common fields of a CV. The ObjectProperty 'cv' defined on domain 'Person' and range 'CurriculumVitae', is declared as subproperty of tm:occurrence, and will be used as an occurrence type in XTM files referring to this ontology.

```
<owl:Class rdf:about="http://www.example.org/ontology#CurriculumVitae">
  <rdfs:subClassOf rdf:resource="http://www.ex.org/ComplexData"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:DatatypeProperty rdf:about="http://www.ex.org/schemaRef"/>
      </owl:onProperty>
      <owl:hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
      http://www.xml.org/xml/schema/f0eee0ed/CVML-1_0.dtd.xml
      </owl:hasValue>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:ObjectProperty rdf:about="http://www.example.org/ontology#cv">
  <rdfs:subPropertyOf rdf:about="http://www.ex.org/occurrence"/>
  <rdfs:range rdf:about="http://www.example.org/ontology#CurriculumVitae"/>
  <rdfs:domain rdf:about="http://www.example.org/ontology#AffiliatedPerson"/>
</owl:ObjectProperty>
```

Validation of topic 'dupont' and its 'cv' occurrence of the previous example is now a two-step process.

- Logical validation : check that 'cv' domain and range are used correctly.

- Physical validation : check that the resource at http://www.example.org/HR/CV/dupont.xml is valid against the DTD at http://www.xml.org/xml/schema/f0eee0ed/CVML-1_0.dtd

In an integrated environment, this validation process can actually be made by two different applications. The ontology-driven topic map engine could make the logical step, and any XML application the physical validation and subsequent relevant processing such as publication, transformation, knowledge extraction ...

The notion of physical type can of course be understood in a very broad sense. It goes from the ideal case mentioned above, of an XML schema or DTD, to a human-readable standard form of administrative document, through a format of image in terms of file extension, dimensions in pixels, rights of diffusion.

Such a mechanism makes the whole KOS more modular and therefore improves usability from the OWL-RDF viewpoint, by making the actual data schema an external resource, that can be managed and modified without any impact on the ontology itself. The other way round, switching to another schema, or adding another allowed schema is made as simple as possible. Moreover, it would be much more easy to re-use the ontology in another enterprise, simply changing the schema(s) of CV allowed to fit the local practices.

Last but not least, this approach make it easier to leverage the legacy of existing data schemas, without the overhead of re-writing and maintaining their content inside an OWL file. In an integrated environments, dedicated applications know what to do with specific complex data already well organized. There is no point to mess up this with the ontology and knowledge base.

4. Conclusion

The proposed framework provides ways for integration of topic maps in the global Semantic Web picture, without demanding any modification of existing standards, nor complete TM-RDF correspondance rules. The basic ontology needed to specify generic TM classes used in the example is really small, and can be defined in a way backward-compatible with existing TM standards. It leaves place to constrain more specific topic maps features like scope and names, that have not been addressed here, in the future TMCL. It sets architectural principles for information systems where each standard takes its natural place without playing the 'more-meta-than-you' game, which leads to the abuse of languages in levels where they seem irrelevant, like using ontologies to model physical data, or topic maps to model ontologies. Once again, we'll insist as a final note on this distinction between ontology, knowledge base and data layers, to be considered as a sensible legacy of many existing information systems, that should not be lost in some 'RDF soup' when trying to find out the best Semantic Web recipes.

5. Acknowledgments

Special thanks to Jack Park, Lars Marius Garshol for their judicious remarks that helped me to fine-tune the final version of this paper.

Bibliography

[PA03] 'XML Topic Maps - Creating and Using Topic Maps for the Web', Jack Park, Sam Hunting, and al., Addison-Wesley 2003. ISBN 0-201-74960-2. Ch.7 'Knowledge Representation, Ontological Engineering and Topic Maps', and Ch.15 'Topic Maps in Knowledge Organization'.

[OWL] W3C Recommendation - Web Ontology Language - Reference. http://www.w3.org/TR/2004/REC-owl-ref-20040210/

[TMDM] ISO 13250-2 Topic Maps Data Model. http://www.isotopicmaps.org/sam/

[TMCL] Topic Map Constraint Language (TMCL) Requirements and Use Cases. http://www.isotopicmaps.org/tmcl/requirements.html

[OSL] The Ontopia Schema Language. http://www.ontopia.net/omnigator/docs/schema/tutorial.html

[ASTMA] Asymptotic Topic Map Languages.http://astma.it.bond.edu.au/

[GA03] Living with topic maps and RDF, Lars Marius Garshol, 2003. http://www.ontopia.net/topicmaps/materials/tmrdf.html

[SWBP] W3C Semantic Web Best Practices and Deployment WG. http://www.w3.org/2001/sw/BestPractices/

[OPTM] Opera Topic Map Overview - Ontology. http://www.ontopia.net/omnigator/models/topicmap_complete.jsp?tm=opera.xtm

Biography

Bernard **Vatant**
Mondeca
Paris
France
bernard.vatant@mondeca.com

Involved in Knowledge Engineering since the late '90s, Bernard Vatant has been working since 2000 as Senior Consultant for Mondeca, designing knowledge models, topic maps and ontologies. He has been participating in various standard bodies, as member of XTM Authoring Group in 2000-2001, founding member and chair of OASIS Published Subjects Technical Committee from 2001 to 2003, and member of W3C WebOnt group since 2003.

Since his former life as a maths teacher, from 1975 to 1997, his research track has steadily been towards efficient and usable knowledge representation, singularly in science popularization and education. This track led him to currently focus on semantic interoperability of knowledge organization systems, like topic maps, thesaurus and ontologies.