DRAFT ONLY 16/05/05

An OWL DL construction for the ISO Topic Map Data Model

Anne Cregan
annec@cse.unsw.edu.au

Knowledge Representation and Reasoning Program,
National Information and Communication Technology
Australia (NICTA) Centre of Excellence

Artificial Intelligence Group
School of Computer Science and Engineering,
University of New South Wales AUSTRALIA

Abstract

Both Topic Maps and the W3C Semantic Web technologies are meta-level semantic maps describing relationships between information resources. Previous attempts at interoperability between XTM Topic Maps and RDF have proved problematic. The ISO's drafting of an explicit Topic Map Data Model **[TMDM 05]** combined with the advent of the W3C's XML and RDF-based Description Logic-equivalent Web Ontology Language **[OWLDL 04]** now provides the means for the construction of an unambiguous semantic model to represent Topic Maps, in a form that is equivalent to a Description Logic representation.

This paper describes the construction of the proposed TMDM ISO Topic Map Standard in OWL DL (Description Logic equivalent) form. The construction is claimed to exactly match the features of the proposed TMDM. The intention is that the topic map constructs described herein, once officially published on the world-wide web, may be used by Topic Map authors to construct their Topic Maps in OWL DL.

The advantage of OWL DL Topic Map construction over XTM, the existing XML-based DTD standard, is that OWL DL allows many constraints to be explicitly stated. OWL DL's suite of tools, although currently still somewhat immature, will provide the means for both querying and enforcing constraints. This goes a long way towards fulfilling the requirements for a Topic Map Query Language (TMQL) and Constraint Language (TMCL), which the Topic Map Community may choose to expend effort on extending. Additionally, OWL DL has a clearly defined formal semantics (Description Logic ref)

1. Introduction

1.1. The Topic Map Tradition

Topic Maps are an ISO/IEC standard **[TM 02]** for mapping web and "real-world" information resources, by reifying real-world resources as "subjects", and creating "topic" constructs to capture their characteristics and relationships with other topics and subjects. Dubbed the "GPS of the information universe", they are akin to an electronic "back-of-book" index, supporting information navigation and retrieval in on-line environments. XML Topic Maps, or XTM **[XTM 01]** is the foremost accepted representation of the ISO standard, developed as XML DTDs by the TopicMaps.Org consortium. Recent ISO/IEC work has concentrated on defining Topic Maps' intended semantics, resulting in the drafting of a Topic Map Data Model has been drafted. **[TMDM 05]**.

1.2. The Semantic Web

The Semantic Web is a W3C-led initiative with the goal of providing technologies and standards for the semantic markup of information resources, thus enabling improved web navigation and supporting intelligent web services. Like XTM, it also is an XML-based technology, with Resource Description Framework (RDF) and Web Ontology Language (OWL) layers superimposed on XML to provide more expressive representations of the characteristics of, and relationships between, logical entities. OWL DL, a formal W3C recommendation finalised in 2004, is a subset of OWL Full which provides a description logic equivalent semantics for OWL ontologies.

1.3. The Interoperability Goal

Although the Topic Map standards have been developed independently of the W3C's Semantic Web initiatives, it has long been felt that as both Topic Maps and the Semantic Web have the same goal to be meta-level maps of information entities, there must be synergies to be exploited. This paper describes how OWL DL may be used to create a Topic Map ontology whose constructs match those in the draft Topic Map Data Model. The OWL DL Topic Map ontology described herein:

- Provides the Topic Map Constructs to allow the user to create their own OWL DL Topic Maps, using the constructs in the proposed TMDM.
- Enables the user to draw on OWL's querying and constraint tools & capability to support validity and consistency checking which both enforces the intended semantics of the TMDM, and allows the user to implement additional constraints for their own user-defined ontologies, by using the constructs provided in OWL DL.
- Is Description Logic equivalent, and enables the construction of user-defined Topic Maps which have a formal Description Logic semantics.

The conversion of existing XTM topic maps to OWL DL is a separate issue, addressed in the author's previous work but not covered herein.

2. Topic Map Data Model

2.1. Motivation

In 2001, two ISO 13250 standard syntaxes for Topic Maps were established, one in HyTM and another in XML **[TM 02]**. However, the standard did not explain how the two syntaxes related to each other, and did not make the common underlying data model explicit. There are non-trivial differences between the syntaxes, and both fail to specify what implementations are to do in a number of situations. Additionally, neither standard supports constraints of the form "A person must be born in a place", "A person must have a least one name" etc.

A draft of requirements for both a Topic Map Constraint Language and Query Language have since been created, and both require a clear description of how Topic Map constructs are to be evaluated. Accordingly, the ISO Topic map team have addressed this with the Topic Map Data Model work that commenced in May 2001, and so far has culminated in the current official draft ISO 13250-2 Topic Maps Data Model **[TMDM 05]** published in January 2005.

2.2. The Topic Map Data Model (TMDM)

For ease of reference the TMDM specification has been compiled by the author and all aspects of the model are shown graphically at **Figure 1**. (Note that "Topic Map Object" can be *any* of the other Topic Map Objects shown (Association, Occurrence, TopicName, etc)[1]).

In the author's opinion, the proposed TMDM makes significant improvements to the previous Topic Map Data Model implied, but never explicitly stated, by the XTM DTD. It clearly states the directions and cardinalities of relationships, and introduces two constraints. It gives data attributes for its logical entities, which include locators such as URI references. It also states type and parent relationships separately and explicitly.

2.3. Fit with OWL DL

The fit between the TMDM and OWL DL is a good one. OWL DL has been designed for the representation of classes containing individuals, which may be related via Object Properties, and may also have attributes represented by Data Properties. OWL enables the building on ontologies with class-subclass relations, additionally allowing both Object and Data Properties to be constrained on Domain, Range, Cardinality and other properties such as transitivity. Although OWL properties are defined in one direction only, they may be linked to the corresponding property in the opposite direction by the use of the "inverse" relation construct.

[1] Although technically, if a Topic reifies another Topic, they will be forced to merge into one Topic in the merge processing

3. Construction of OWL DL model

3.1. Design Choices

In order to construct the TMDM in OWL DL, some design choices need to be made.

Firstly, we need to decide what the final individuals in Topic maps will be, recalling that whilst Object and Data Properties are defined with classes as their domains and ranges, they actually exist between the individuals within those classes. Accordingly, the candidates for individuals are taken to be those individual Topics, Occurrences, Associations and so forth that users would ultimately want to define within their Topic Maps.

The constructs Topic, Occurrence and Association and so forth should therefore be defined as classes which will contain those individuals. We may then define the relations between these and other Topic map constructs as Object Relations, with domains and ranges being appropriately defined classes. The user will ultimately instantiate these relations in their own ontologies, but our OWL DL ontology will provide the general framework which defines which entities may be related to which other entities, in what ways they relate (cardinality etc), and the attributes they may have.

We also explain here why we have opted to define Topics as individuals, even though they are used as types. Normally in an OWL ontology, a type would be represented as a class of individuals. However, as the TMDM requires that types are also themselves Topics which may be used in all the ways that regular Topics are, we must opt to reflect type relations as Object Relations between individual Topic Map Objects and the individual Topics which are their types.

As the "Type" and "Scope" of a Topic Map Object are filled by Topics, we have defined these as specific subclasses of "Topic". We have preserved the ability to separate Association Types, Occurrence Types, Topic Name Types and Association Role Types, as these groups are likely to have large non-overlapping components. The ability to separate these constructs will prove useful for implementing some of the desired TMCL requirements, as well as for supporting the user in managing user-defined types within their Topic Maps. We have not, however, opted to preserve any separation of Scope subclasses according to the kind of TM Object relating to it, as these as expected to differ little between TM Objects, as typically Scope relates to the division of subject domains more generally.

3.2. The OWL DL Model

The proposed OWL DL model is presented pictorially at **Figure 2**. As the author is not currently aware of any standard graphical notation for representing OWL or other ontologies beyond RDF graphs, she has devised her own notation as a natural extension of RDF notation.

Classes are represented as ovals, and the nesting of an oval within a larger oval indicates a subclass-superclass relation. Arrows represent object properties, labelled with the Object Property's name, and cardinalities, and indicating direction by pointing from the Domain to the Range. To avoid cluttering the diagram, data properties are omitted in **Figure 2**, as are constraints.

Each entity (box) in **Figure 1**, has been translated to an oval (class) within **Figure 2**. For instance, the oval "Topic" denotes an OWL class which will contain individuals Topics as instances. As explained previously, subclasses are created for the various Types and for Scope. Although shown not overlapping, we have opted not to define them as disjoint, so the individuals therein may in fact be common to multiple subclasses if the user wishes ie the same Topic could be used for both an Association Type and an Occurrence Type, as well as a Scope. This could also be prevented by additional OWL code to define these subclasses as disjoint.

Relationships between the entities in **Figure 1** are captured as Object Properties in **Figure 2**. Where there is a bi-directional relation in **Figure 1**, two relations have been given in **Figure 2**, and related as inverses. For additional ease of use, many additional inverse relations have also been defined. Logically, these are already implied by the TMDM, and will ultimately allow the user more convenience in writing code for their own ontologies.

The attributes of each entity in Figure 2 (source locator, etc) will translate to OWL Data Properties of the individuals within the OWL classes (not shown at **Figure 2**). Constraints will be implemented as restrictions on OWL classes. Full details are given in the following section.

4. Construction of OWL DL ontology

This section gives the full OWL construction corresponding to **Figure 2**, plus Data Properties and Constraints, thus fully capturing every aspect of the proposed TMDM as detailed in **Figure 1**.

The intention is that the code here defined, once finalized and approved by the relevant Standards Bodies, will be the definitive representation of the Topic Map Data Model in OWL DL syntax. It will be published as an online document, and included by import in all User-Defined Topic Maps written in OWL which wish to use the Topic Map Data Model constructs. Note that no XTM references are necessary, either in the TMDM ontology or in user-defined Topic Map ontologies.

Section 5 works through some examples showing how a user-defined Topic Map would reference and use the OWL DL constructs defined in Section 4.

4.1. OWL document Header

After the standard XML, RDF and OWL inclusions are made, this ontology is named "Topic Map Data Model Ontology":

```
<owl:Ontology rdf:about="">
  <rdfs:label>Topic Map Data Model Ontology</rdfs:label>
</owl:Ontology>
```

(Note: indicative name only, subject to ISO approval).

4.2. Classes

Firstly, OWL classes are created for each TMDM Entity:

```
<owl:Class rdf:id="Topic">
<owl:Class rdf:id="Occurrence">
<owl:Class rdf:id="TopicName">
<owl:Class rdf:id="Variant">
<owl:Class rdf:id="Association">
<owl:Class rdf:id="Association Role">
```

Scope and types are defined as subclasses of Topic. Type also has further subclasses within it:

```
<owl:Class rdf:id="Scope">
  <rdfs:subclassOf rdf:resource="#Topic"/>
</owl: Class>

<owl:Class rdf:id="Type">
  <rdfs:subclassOf rdf:resource="#Topic"/>
</owl: Class>

<owl:Class rdf:id="AssociationType">
  <rdfs:subclassOf rdf:resource="#Type"/>
</owl: Class>

<owl:Class rdf:id="AssociationRoleType">
  <rdfs:subclassOf rdf:resource="#Type"/>
</owl: Class>

<owl:Class rdf:id="OccurrenceType">
  <rdfs:subclassOf rdf:resource="#Type"/>
</owl: Class>

<owl:Class rdf:id="TopicNameType">
  <rdfs:subclassOf rdf:resource="#Type"/>
</owl: Class>
```

Individuals to be declared in user-defined OWL TM documents will belong to at least one of the classes defined. For instance a user-defined topic called "mytopic" would be declared thus:

```
<Topic rdf:id="mytopic">
</Topic>
```

Within this section, further restrictions are added onto these classes via constraints on the OWL properties related to them.

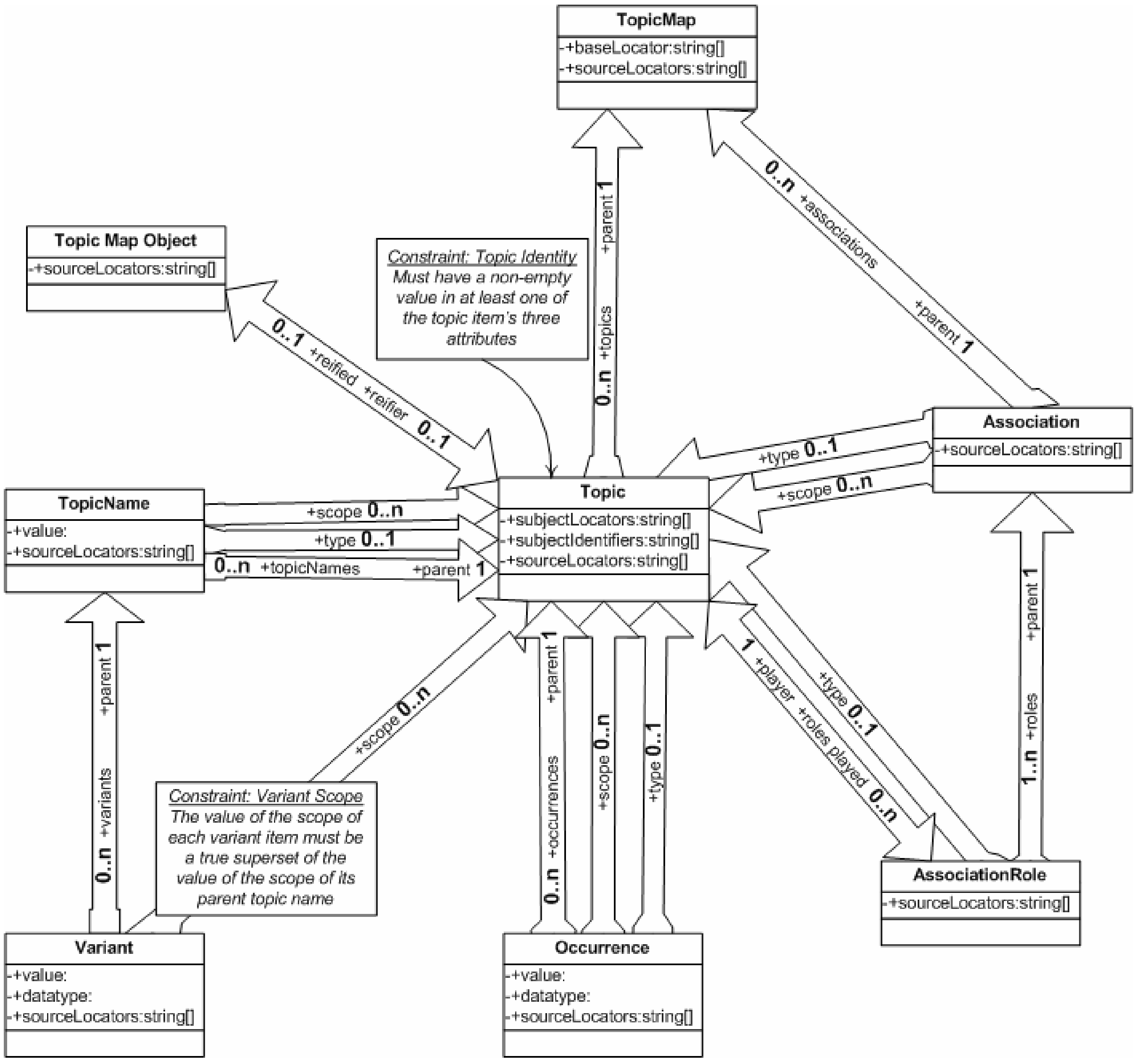
TopicMap
-+baseLocator:string[]
-+sourceLocators:string[]
Topic Map Object
-+sourceLocators:string[]
Constraint: Topic Identity
Must have a non-empty value in at least one of the topic item's three attributes
Association
-+sourceLocators:string[]
TopicName
-+value:
-+sourceLocators:string[]
Topic
-+subjectLocators:string[]
-+subjectIdentifiers:string[]
-+sourceLocators:string[]
AssociationRole
-+sourceLocators:string[]
Constraint: Variant Scope
The value of the scope of each variant item must be a true superset of the value of the scope of its parent topic name
Variant
-+value:
-+datatype:
-+sourceLocators:string[]
Occurrence
-+value:
-+datatype:
-+sourceLocators:string[]
0..n +topics +parent 1
0..n +associations +parent 1
0..1 +reified +reifier 0..1
+type 0..1
+scope 0..n
+scope 0..n
+type 0..1
0..n +topicNames +parent 1
0..n +variants +parent 1
+scope 0..n
0..n +occurrences +parent 1
+scope 0..n
+type 0..1
1 +player +roles played 0..n
+type 0..1
1..n +roles +parent 1

Figure1: Draft Topic Map Data Model (ISO 13250-2) – author's representation

4.3. Object Properties

This section defines the relationships between individuals within the Topic Map classes. As OWL properties are directional, inverses are also defined to give the complete construction. By making these logically equivalent forms available, users are given more flexibility to write their own Topic Map ontologies in the most convenient way.

4.3.1 scope

Occurrence, Topic Name, Variant and Association may all be related to any number of Scopes.[2] Thus we define a Property "scope" which has all these classes as its domain and Scope as its range. There is no need to add any cardinality restriction.

[2] Note the OWL convention that Classes commence with a capital letter, whereas properties do not, so "Scope" is a class, whereas "scope" is a property.

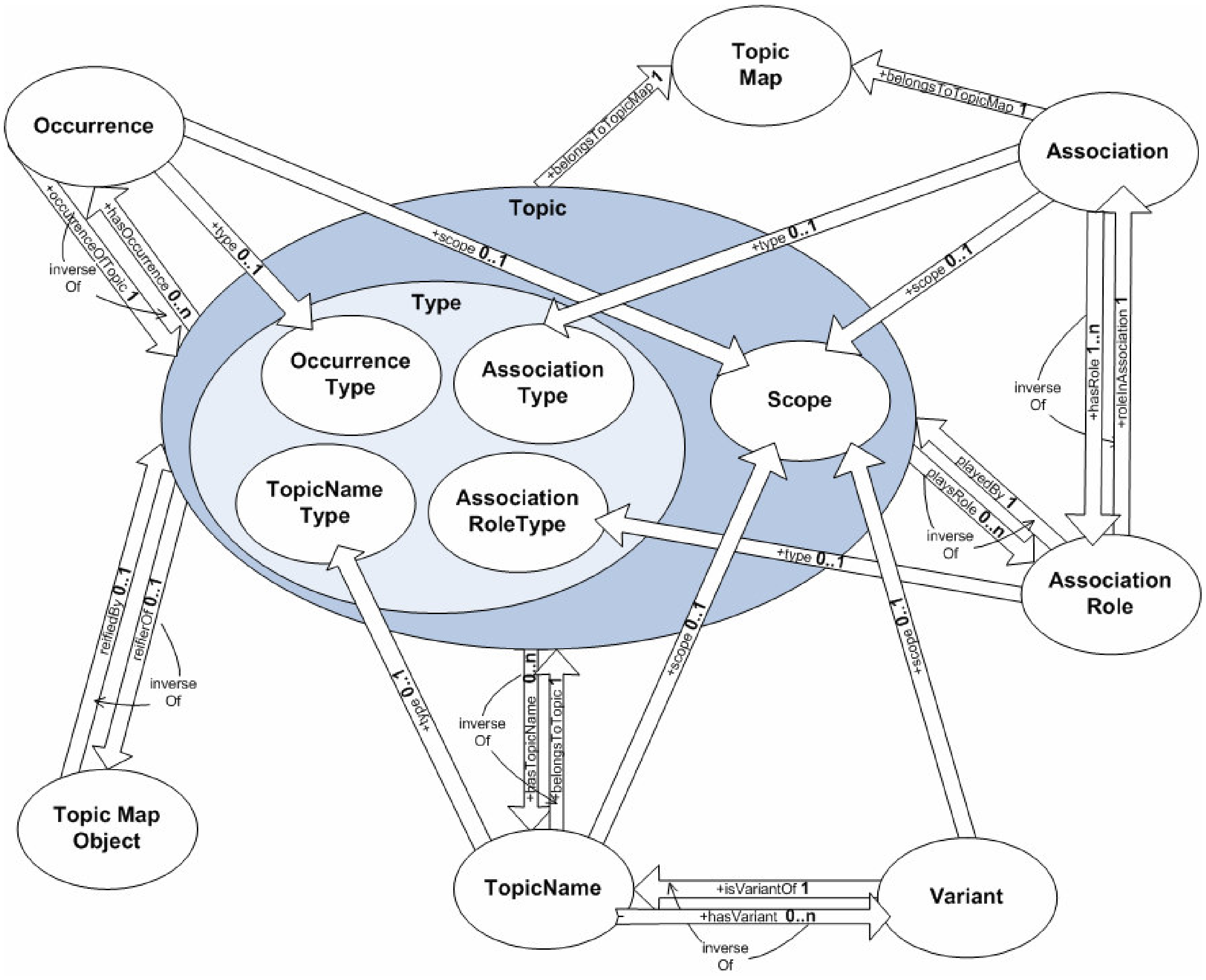

Figure 2 OWL DL Topic Map Graphical Representation

```
  <owl: ObjectProperty rdf:ID= “scope”>
   <rdfs:domain>
       <owl:Class>
            <owl:unionOf rdf:parsetype=”Collection”>
                <owl:Class rdf:about=”#TopicName” />
                <owl:Class rdf:about=”#Variant” />
                <owl:Class rdf:about=”#Occurrence” />
                <owl:Class rdf:about=”#Association”/>
            </owl:unionOf>
        <owl:Class>
   </rdfs:domain>
   <rdfs:range rdf:resource=”#Scope>
</owl:ObjectProperty>
```

Variant Scope Constraint:
NOTE: There is a constraint on Variant Scope as follows: ***The value of the scope of each individual variant item must be a true superset of the value of the scope of its parent topic name***.

*//** TO BE ADVISED: the best way to implement this is still under consideration by the author **//*

4.3.2 type

Individuals within the classes TopicName, Occurrence, Association and AssociationRole may each have 0 or 1 types. Accordingly, “type” is defined as a functional property from individuals within the union of these classes to the “Type” class:

```
<owl: ObjectProperty rdf:ID= “type”>
  <rdf:type    rdf:resource= “&owl;FunctionalProperty”/>
  <rdfs:domain>
       <owl:Class>
            <owl:unionOf rdf:parsetype=”Collection”>
                <owl:Class rdf:about=”#TopicName” />
                <owl:Class rdf:about=”#AssociationRole” />
                <owl:Class rdf:about=”#Occurrence” />
                <owl:Class rdf:about=”#Association”/>
            </owl:unionOf>
        <owl:Class>
  </rdfs:domain>
  <rdfs:range rdf:resource=”#Type>
</owl:ObjectProperty>
```

A functional property relates each item in its domain to one and only one item in its range. Thus if a Topic Map individual has a user-defined type property, it is able to be related to only one individual from the Type class. It may also have zero types in a user-defined Topic Map if no type property is declared for it.

Additional restrictions are set to force each relevant Topic Map individual to have a "type" from the subclass of the appropriate kind within the "Type" class:

```
<owl:Class rdf:about="#Association"/>
    <owl:subClassOf>
         <owl:Restriction>
              <owl:onProperty rdf:resource="#type"
              <owl:allValuesFrom rdf:resource="#AssociationType"
         </owl:Restriction>
      <owl:subClassOf>
</owl:Class>

<owl:Class rdf:about="#AssociationRole"/>
    <owl:subClassOf>
         <owl:Restriction>
              <owl:onProperty rdf:resource="#type"
              <owl:allValuesFrom rdf:resource="#AssociationRoleType"
         </owl:Restriction>
      <owl:subClassOf>
</owl:Class>

<owl:Class rdf:about="#Occurrence"/>
    <owl:subClassOf>
         <owl:Restriction>
              <owl:onProperty rdf:resource="#type"
              <owl:allValuesFrom rdf:resource="#OccurrenceType"
         </owl:Restriction>
      <owl:subClassOf>
</owl:Class>

<owl:Class rdf:about="#TopicName"/>
    <owl:subClassOf>
         <owl:Restriction>
              <owl:onProperty rdf:resource="#type"
              <owl:allValuesFrom rdf:resource="#TopicNameType"
         </owl:Restriction>
      <owl:subClassOf>
</owl:Class>
```

4.3.3 parent

Each TopicName must have exactly one parent Topic. This is enforced via a cardinality restriction on the property relating TopicName to Topic:

```
<owl: ObjectProperty rdf:ID= "belongsToTopic">
   <rdfs:domain rdf:resource= "#TopicName" />
   <rdfs:range  rdf:resource= "#Topic" />
</owl:ObjectProperty>

<owl:Class rdf:about="#TopicName"/>
    <owl:subClassOf>
         <owl:Restriction>
              <owl:onProperty  rdf:resource="#belongsToTopic"
              <owl:Cardinality rdf:datatype=
                   "&xmls;nonNegativeInteger">1</owl:Cardinality>
           </owl:Restriction>
      <owl:subClassOf>
</owl:Class>
```

Conversely, an individual Topic may have any number of TopicNames. This is defined as an inverse property of the "belongsToTopic" property defined above, giving users the option to define Topic/TopicName relations using either Topic or TopicName as the starting point, as is most convenient.

```
<owl: ObjectProperty rdf:ID= "hasTopicName"
   <owl:inverseOf rdf:resource="#belongsToTopic"/>
</owl:ObjectProperty>
```

Each Occurrence has exactly one parent Topic, enforced via a Cardinality constraint on the "occurrenceOfTopic" Object Property:

```
<owl: ObjectProperty rdf:ID= "occurrenceOfTopic">
   <rdfs:domain rdf:resource= "#Occurrence" />
   <rdfs:range  rdf:resource= "#Topic" />
</owl:ObjectProperty>

<owl:Class rdf:about="#Occurrence"/>
    <owl:subClassOf>
         <owl:Restriction>
              <owl:onProperty rdf:resource="#occurrenceOfTopic"
              <owl:Cardinality rdf:datatype=
                   "&xmls;nonNegativeInteger">1</owl:Cardinality>
           </owl:Restriction>
      <owl:subClassOf>
</owl:Class>
```

Conversely, a Topic may have any number of Occurrences, defined as an inverse of occurrenceOfTopic :

```
<owl: ObjectProperty rdf:ID= "hasOccurrence"
   <owl:inverseOf rdf:resource="#occurrenceOfTopic"/>
</owl:ObjectProperty>
```

Each Variant has exactly one parent TopicName, enforced via a Cardinality constraint on the "hasVariant" Object Property:

```
<owl: ObjectProperty rdf:ID= "isVariantOf"
   <rdfs:domain rdf:resource= "#Variant" />
   <rdfs:range  rdf:resource= "#TopicName"/>
</owl:ObjectProperty>

<owl:Class rdf:about="#Variant"/>
    <owl:subClassOf>
         <owl:Restriction>
              <owl:onProperty rdf:resource="#isVariantOf"
              <owl:Cardinality rdf:datatype=
                   "&xmls;nonNegativeInteger">1</owl:Cardinality>
           </owl:Restriction>
      <owl:subClassOf>
</owl:Class>
```

Conversely, a TopicName may have any number of Variants, defined as an inverse property of hasVariant:

```
<owl: ObjectProperty rdf:ID= "hasVariant"
   <owl:inverseOf rdf:resource="#isVariantOf"/>
</owl:ObjectProperty>
```

An AssociationRole has exactly one parent Association, enforced via a Cardinality constraint on the "roleInAssociation" Object Property:

```
<owl: ObjectProperty rdf:ID= "roleInAssociation"
   <rdfs:domain rdf:resource= '#AssociationRole' />
   <rdfs:range  rdf:resource= '#Association"/>
</owl:ObjectProperty>

<owl:Class rdf:about="#AssociationRole"/>
    <owl:subClassOf>
         <owl:Restriction>
              <owl:onProperty rdf:resource="#roleInAssociation"
              <owl:Cardinality rdf:datatype=
                     "&xmls;nonNegativeInteger">1</owl:Cardinality>
          </owl:Restriction>
      <owl:subClassOf>
</owl:Class>
```

Conversely, each Association must have at least one AssociationRole, set up as a minimum cardinality restriction on the inverse property "hasRole":

```
<owl: ObjectProperty rdf:ID= "hasRole"
   <owl:inverseOf rdf:resource="#roleInAssociation"/>
</owl:ObjectProperty>

<owl:Class rdf:about="#Association"/>
    <owl:subClassOf>
         <owl:Restriction>
              <owl:onProperty rdf:resource="#hasRole"
              <owl:minCardinality rdf:datatype=
                     "&xmls;nonNegativeInteger">1</owl:minCardinality>
          </owl:Restriction>
      <owl:subClassOf>
</owl:Class>
```

4.3.4 Topics Playing Association Roles

Every AssociationRole must be played by exactly one Topic, enforced via a Cardinality constraint on the "playedBy" Object Property:

```
<owl: ObjectProperty rdf:ID= "playedBy"
   <rdfs:domain rdf:resource= "#AssociationRole" />
   <rdfs:range  rdf:resource= "#Topic" />
</owl:ObjectProperty>

<owl:Class rdf:about="#AssociationRole"/>
    <owl:subClassOf>
         <owl:Restriction>
              <owl:onProperty  rdf:resource="#playedBy"
              <owl:Cardinality rdf:datatype=
                     "&xmls;nonNegativeInteger">1</owl:Cardinality>
          </owl:Restriction>
      <owl:subClassOf>
</owl:Class>
```

Conversely, each Topic may play any number of AssociationRoles, captured by playsRole, defined as an inverse property of playedBy:

```
<owl: ObjectProperty rdf:ID= "playsRole"
   <owl:inverseOf rdf:resource="#playedBy"/>
</owl:ObjectProperty>
```

4.3.5 Reification

Any topic map object may be reified by one only Topic, and any Topic may be used to reify one only Topic Map Object:

```
<owl: DataProperty rdf:ID= "reifiedBy"
   <rdf:type    rdf:resource= "&owl;FunctionalProperty"/>
   <rdfs:range  rdf:type= "#Topic" />
</owl:DataProperty>

<owl: DataProperty rdf:ID= "reifierOf"
   <rdf:type    rdf:resource= "&owl;FunctionalProperty"/>
   <owl:inverseOf rdf:resource= "#reifiedBy" />
</owl:DataProperty>
```

4.3.6 Parent Topic Map

Lastly, object properties are created to capture the notion that each Topic and Association belongs to exactly one parent TopicMap :

```
<owl: ObjectProperty rdf:ID= "belongsToTopicMap">
   <rdfs:domain>
       <owl:Class>
            <owl:unionOf rdf:parsetype="Collection">
                <owl:Class rdf:about="#Topic" />
                <owl:Class rdf:about="#Association" />
            </owl:unionOf>
        <owl:Class>
   </rdfs:domain>
   <rdfs:range  rdf:resource= "#TopicMap" />
</owl:ObjectProperty>

<owl:Class rdf:about="#Topic"/>
    <owl:subClassOf>
         <owl:Restriction>
              <owl:onProperty  rdf:resource="#belongsToTopicMap"
              <owl:Cardinality rdf:datatype=
                     "&xmls;nonNegativeInteger">1</owl:Cardinality>
          </owl:Restriction>
      <owl:subClassOf>
</owl:Class>

<owl:Class rdf:about="#Association"/>
    <owl:subClassOf>
         <owl:Restriction>
              <owl:onProperty  rdf:resource="#belongsToTopicMap"
              <owl:Cardinality rdf:datatype=
                     "&xmls;nonNegativeInteger">1</owl:Cardinality>
          </owl:Restriction>
      <owl:subClassOf>
</owl:Class>
```

This concludes the definition of OWL object properties.

4.4. Data Properties

This section defines the relationships between individuals within the Topic Map classes, and data values, of specified datatypes. The standard XML Schema Part 2: Datatypes **[XMLS2 04]** are used (denoted "&xmls" here). As per the TMDM specification, the locators are given as being of type “string”, but the option of forcing them to be URI references by setting datatype = ”&xmls; any-URI” should probably be considered, as this is the standard for locating resources on the Semantic Web.

4.4.1 baseLocator

A TopicMap may have a base locator that is a string:

```
<owl:Class rdf:id=”TopicMap”/>

<owl: DataProperty rdf:ID= “baseLocator”
   <rdf:domain rdf:resource=”#TopicMap”/>
   <rdfs:range  rdf:datatype= “&xmls;string” />
</owl:DataProperty>
```

4.4.2 Locator

As there is a constraint over the three Topic attributes (see also **4.4.6**), we create a data property “Locator” that will contain the three data properties sourceLocator, subjectLocator, and subjectIdentifier as subproperties:

```
<owl: DataProperty rdf:ID= “Locator”
   <rdfs:range  rdf:datatype=”&xmls;anyURI”/>
</owl:DataProperty>
```

4.4.3 sourceLocator

Every topic map object may have any number of source locators which are URIs:

```
<owl: DataProperty rdf:ID= “sourceLocator”
   <rdfs:subPropertyOf rdf:resource=”#Locator”/>
   <rdfs:range  rdf:datatype= =”&xmls;anyURI”/>
</owl:DataProperty>
```

4.4.4 subjectLocator

A Topic may have any number of subject Locators of type anyURI (all required to point to the same “thing” but enforcing this is outside the scope of the TMDM):

```
<owl: DataProperty rdf:ID= “subjectLocator”
   <rdfs:subPropertyOf rdf:resource=”#Locator”/>
   <rdf:domain rdf:resource=”#Topic”/>
   <rdfs:range  rdf:datatype=”&xmls;anyURI”/>
</owl:DataProperty>
```

4.4.5 subjectIdentifier

A Topic may have any number of subject Identifiers:

```
<owl: DataProperty rdf:ID= “subjectIdentifier”
   <rdfs:subPropertyOf rdf:resource=”#Locator”/>
   <rdf:domain rdf:resource=”#Topic”/>
   <rdfs:range  rdf:datatype= “&xmls;anyURI” />
</owl:DataProperty>
```

4.4.6 Topic Locator Constraint

The TMDM specifies the following constraint on Topic: ***at least one of the three attributes subjectLocator, subjectIdentifier and sourceLocators must be present***. This is enforced in OWL by creating a minimum cardinality restriction on class Topic with regard to the Locator property defined earlier. As “Locator” encompasses the other three properties bound by the constraint, the restriction ensures at least one of them must be present.

```
<owl:Class rdf:about=”#Topic”/>
    <owl:subClassOf>
         <owl:Restriction>
              <owl:onProperty rdf:resource=”#Locator”
              <owl:minCardinality rdf:datatype=
                   ”&xmls;nonNegativeInteger”>1</owl:minCardinality>
           </owl:Restriction>
      <owl:subClassOf>
</owl:Class>
```

4.4.7 topicNameString

TopicName may be given one value, of type string only. (The datatype is given by the Property’s range):

```
<owl: DataProperty rdf:ID= “topicNameString”
   <rdf:type    rdf:resource= “&owl;FunctionalProperty”/>
   <rdfs:domain rdf:resource= “#TopicName”/>
   <rdfs:range  rdf:resource= “&xmls;string”/>
</owl:DataProperty>
```

4.4.8 Variant and Occurrence Text

Variant and Occurrence may have either:

- Name or text (respectively) as a string value **OR**
- a locator where the relevant name or text resides.

The TMDM distinguishes between the two cases based on datatype; however, as this is not implementable in OWL without extra processing, we have opted to separate the two cases into different Data Properties:

```
<owl: DataProperty rdf:ID= “variantName”
   <rdf:type    rdf:resource= “&owl;FunctionalProperty”/>
   <rdfs:domain rdf:resource= “#Variant”/>
   <rdfs:range  rdf:resource= “&xmls;string”/>
</owl:DataProperty>

<owl: DataProperty rdf:ID= “occurrenceText”
   <rdf:type    rdf:resource= “&owl;FunctionalProperty”/>
   <rdfs:domain rdf:resource= “#Occurrence”/>
   <rdfs:range  rdf:resource= “&xmls;string”/>
</owl:DataProperty>

<owl: DataProperty rdf:ID= “variantNameLocatedAt”
   <rdf:type    rdf:resource= “&owl;FunctionalProperty”/>
   <rdfs:domain rdf:resource= “VariantName”/>
   <rdfs:range  rdf:datatype= “&xmls;anyURI”/>
</owl:DataProperty>

<owl: DataProperty rdf:ID= “occurrenceTextLocatedAt”
   <rdf:type    rdf:resource= “&owl;FunctionalProperty”/>
   <rdfs:domain rdf:resource= “Occurrence”/>
   <rdfs:range  rdf:datatype= “&xmls;anyURI”/>
</owl:DataProperty>
```

5. Examples

In this section, we use the OWL TMDM ontology as defined above to create our own User-defined Topic Map. The "hamlet" example from the XTM1.0 specification was used as the inspiration and the information presented herein corresponds very closely with it.

Assume that the Topic Map Data Model Ontology laid out in Section 4 resides at "http://somelocation/tmdm"

5.1. User TM document Header

Set up "tm" namespace for tmdm, make standard XML, RDF and OWL inclusions are made, import the "Topic Map Data Model Ontology" and name the user ontology:

```
<owl:Ontology rdf:about="">
  <rdfs:comment> An example user-defined Topic Map for
Shakespeare, constructed in OWL DL using the TMDM OWL DL
constructs   </rdfs:comment>
  <rdfs:label>Shakespeare Topic Map Ontology</rdfs:label>
  <owl:imports rdf:resource ="http://someofficiallocation/tmdm"/>
</owl:Ontology>
```

The ontology can contain more than one Topic Map, so we name the individual Topic Map.

```
<tm:TopicMap rdf:id="Shakespeare_Topic_Map">
</tm:Topic Map>
```

Our user-defined Topics and Associations will belong to this Topic Map.

Note the use of the prefix "tm", which indicates the construct is from the imported TMDM ontology.

5.2. Topics, Topic Names, Types, Occurrences

Say we would like to create a topic "hamlet" which has:

- A topic name String: "Hamlet, Prince of Denmark"
- An occurrence of type "plain-text-format" which resides at the Gutenberg.org ftp site

The OWL code which achieves this is given below:

```
<tm:Topic rdf:id="hamlet">
     <tm:belongsToTopicMap
rdf:resource="#Shakespeare_Topic_Map"/>
</tm:Topic>

<tm:TopicName rdf:id="#hamletTopicName">
    <tm:topicNameString>The Tragedy of Hamlet,
               Prince of Denmark</topicNameString>
     <tm:belongsToTopic rdf:resource="#hamlet"/>
</tm:TopicName>

<tm:OccurrenceType rdf:id="plain_text_format">
      <tm:belongsToTopicMap
rdf:resource="#Shakespeare_Topic_Map"/>
</tm:OccurrenceType>

<tm:Occurrence rdf:id="#hamletOccurrence1">
       <tm:type rdf:resource="#plain_text_format"/>
       <tm:occurrenceOfTopic rdf:resource="#hamlet"/>
       <tm:occurrenceTextLocatedAt rdf:resource=
"ftp://www.gutenberg.org/pub/ gutenberg/.hamlet_full_text.txt" />
</tm:Occurrence>
```

Firstly, we create a Topic "hamlet", and state that it belongs to the "Shakespeare_Topic_Map" TopicMap. Then we create an individual TopicName, which has topicNameString of "Hamlet, Prince of Denmark", and state that this TopicName belongs to the Topic "hamlet". Note that we could have opted to define the TopicName first, and then define the Topic, connecting it to the Topic Name using the "hasTopicName" Object Property. This would give an exactly equivalent construction.

We then create the OccurrenceType "plain_text_format". Since OccurrenceType has been defined in the TMDM ontology as a subclass of topic, all OccurrenceTypes are automatically Topics. We state that it belongs to our Shakespeare TopicMap.

Then we create an individual Occurrence, which has a topicNameString of "Hamlet, Prince of Denmark", and give it a type of "plain_text_format". Since in the TMDM ontology we stated that when the domain of the "type" Object Property was "Occurrence" the range must be "OccurrenceType", OWL consistency checks will produce an error if this is not the case.

Using the occurrenceOfTopic ObjectProperty, we state that this Occurrence is of the Topic "hamlet". Lastly we use occurrenceTextIsLocated to give the URI location where the full text of hamlet is located. Note that this must be a valid URI or an error will be produced.

5.3. Locators

OWL consistency checks will still however produce an error on the above code, because we have neglected to specify at least one locator for our Topics.

A locator such as SubjectIdentifier can be set as follows (similarly for baseLocator, sourceLocator and subjectLocator):

```
<tm:Topic rdf:id="dk">
      <tm:subjectIdentifier>http://www.topicmaps.org/xtm/1.0/
          country. xtm#DK/</tm:subjectIdentifier>
</tm:Topic>
```

Here we have set the subjectIdentifier for a Topic "dk" to a web reference where country codes are kept.

5.4. Associations and Association Roles

Suppose we would now like to capture the authorship relationship between the work "hamlet", the author "William Shakespeare" and the type of work "play".

To do this we define an Association which has three association roles for the work, author and type of work. Firstly we set these up as Association Role Types, then create the Association which has roles of these types. Assuming that we have already set up the topics "hamlet", "William Shakespeare" and "play", we simply slot these in to the relevant AssociationRole using the Object Property playedBy:

```
<tm:AssociationRoleType rdf:id="work">
      <tm:belongsToTopicMap rdf:resource="#shakespeare"/>
      <tm:subjectLocator …/>
</ tm:AssociationRoleType>

<tm:AssociationRoleType rdf:id="type_of_work">
      <tm:belongsToTopicMap rdf:resource="#shakespeare"/>
      <tm:subjectLocator …/>
</ tm:AssociationRoleType>

<tm:AssociationRoleType rdf:id="author">
      <tm:belongsToTopicMap rdf:resource="#shakespeare"/>
      <tm:subjectLocator  …/>
</ tm:AssociationRoleType>

<tm:Association rdf:id="will_wrote_play_hamlet">
     <tm:belongsToTopicMap rdf:resource="#shakespeare"/>
</tm:Association>

<tm:AssociationRole rdf:id="author_Will">
      <tm:type rdf:resource= "#author"/>
      <tm:roleInAssociation rdf:resource=
            "#will_wrote_play_hamlet">
       <tm:playedBy rdf:resource="#William Shakespeare">
</tm:AssociationRole>

<tm:AssociationRole rdf:id="work_hamlet">
      <tm:type rdf:resource= "#work"/>
      <tm:roleInAssociation rdf:resource=
            "#will_wrote_play_hamlet">
      <tm:playedBy rdf:resource="#hamlet">
</tm:AssociationRole>

<tm:AssociationRole rdf:id="type_of_work_hamlet">
      <tm:type rdf:resource= "#type_of_work"/>
      <tm:roleInAssociation rdf:resource=
            "#will_wrote_play_hamlet">
      <tm:playedBy rdf:resource="#play">
</tm:AssociationRole>
```

Consistency checks enforce the following OWL restrictions:

- Association Roles can only be played by topics
- An Association must have at least one Role
- A Role must belong to exactly one Association
- Types of Association Roles must come from the "AssociationRoleType" class (subclass of Topic)

Note that in this example, we have opted not to give our Association an associationType property, but we could easily do so in the same manner as it is done for each AssociationRole, and OWL would check that the type given to the Association belonged to the AssociationType class (subclass of Topic).

5.5. Scope

Say we would like to give the Topic "hamlet" a different topicNameString depending on whether we are referring to "Hamlet" the play, or "Hamlet" the character. We do this by creating Scope individuals for "play" and for "character" and assigning the respective scopes to TopicNames which have different Topic name Strings. Scopes are automatically Topics as Scope is a subclass of Topic in the TMDM ontology.

```
 <tm:Scope rdf:id = "play">
     <tm:belongsToTopicMap rdf:resource="#shakespeare"/>
     <tm:subjectLocator rdf:resource=…/>
</tm:Scope>

<tm:Scope rdf:id = "character">
     <tm:belongsToTopicMap rdf:resource="#shakespeare"/>
     <tm:subjectLocator rdf:resource=…/>
</tm:Scope>

<tm:Topic rdf:id="hamlet">
     <tm:belongsToTopicMap rdf:resource="#shakespeare"/>
     <tm:subjectIndicator rdf:resource=…/>
</tm:Topic>

<tm:TopicName rdf:id="#hamletTopicName">
    <tm:topicNameString>The Tragedy of Hamlet,
               Prince of Denmark</topicNameString>
     <tm:scope rdf:resource="#play"/>
     <tm:belongsToTopic rdf:resource="#hamlet"/>
</tm:TopicName>

<tm:TopicName rdf:id="#hamletCharacterName">
    <tm:topicNameString>Hamlet</topicNameString>
    <tm:scope rdf:resource="#character"/>
     <tm:belongsToTopic rdf:resource="#hamlet"/>
</tm:TopicName>
```

Note that both TopicNames belong to the same Topic. If we wanted to refer to "hamlet" the geographical township, this would refer to a different subject, and should therefore be created as a different Topic. (We probably would put it in a different Topic Map too.) Note that scope can be used with Associations, AssociationRoles and Occurrences also.

5.6. Variants

Say we would like to give our "hamletTopicName" Topic Name from the previous "scope" example at 5.5 some Variant names for display under different circumstances.

Note the constraint on Variant Scope as follows:

The value of the scope of each individual variant item must be a true superset of the value of the scope of its parent topic name.

So any variants of "hamletTopicName" must have at least all the scopes that apply to "hamletTopicName", plus at least one more. Note in the example the scope "play" is added to each Variant, as this is the scope of "hamletTopicName". Additionally we specify the scopes which define the parameters within which the Variant is to be used.

```
<tm:Scope rdf:id="display"/>
<tm:Scope rdf:id="icon"/>
<tm:Scope rdf:id="large"/>
<tm:Scope rdf:id="small"/>

<tm:Variant rdf:id="display+icon+large">
    <tm:isVariantOf rdf:resource="hamletTopicName"/>
    <tm:scope rdf:resource="#play" />
    <tm:scope rdf:resource="#display" />
    <tm:scope rdf:resource="#icon"/>
    <tm:scope rdf:resource="#large"/>
    <tm:variantNameLocatedAt
            rdf:resource="img/hamlet64x64.png"/>
</tm:Variant>

<tm:Variant rdf:id="display+icon+small">
    <tm:isVariantOf rdf:resource="hamletTopicName"/>
    <tm:scope rdf:resource="#play"/>
    <tm:scope rdf:resource="#display" />
    <tm:scope rdf:resource="#icon"/>
    <tm:scope rdf:resource="#small"/>
    <tm:variantNameLocatedAt
            rdf:resource="img/hamlet16x16.png"/>
</tm:Variant>
```

6. Conclusions and Future Work

This paper has presented a proposal for an OWL DL formalisation of the ISO drafted standard Topic Map Data Model. The constructs defined herein provide the basis for building user-defined Topic Maps in OWL DL or OWL Full.

The author believes that the use of this formalism and associated tools will meet the majority of the requirements for a Topic Map Query Language (TMQL) and a Topic Map Constraint Language (TMCL). Some additional extensions to the OWL tools may be needed to meet all the requirements specified, but the use of the OWL platform and existing tools as a base should significantly reduce the workload required.

The author intends in immediate future work to analyse the extent to which OWL DL and associated tools may satisfy the TMQL and TMCL use cases specified.

DISCLAIMER:
Please note that this document is currently at DRAFT stage only, and is NOT an official proposal. The purpose of this document is to illustrate the fit between OWL and the TMDM, and the potential for the use of OWL DL as a standard for Topic Map implementation. After discussion with the ISO/IEC, it may or may not be revised and put forward as an official proposal for an OWL standard for Topic Map implementation. At the very least, revisions are likely to be needed before such an event might take place.

References

[OWLDL 04] *OWL Web Ontology Language Reference,* World-Wide Web Consortium (W3C) Recommendation, 10 February 2004, available at http://www.w3.org/TR/2004/REC-owl-ref-20040210/

[TMDM 05] *Topic Maps – Part 2: DataModel* Draft ISO/IEC International Standard (ISO/IEC JTC 1/SC34 ISO/IEC FCD 13250-2), 10 January 2005, available at http://www.isotopicmaps.org /sam/sam-model/

[TM 02] *Topic Maps Second Edition*, ISO/IEC International Standard (ISO/IEC 13250*)*, 19 May 2002, available at http://www.y12.doe.gov/sgml/sc34/document/0322_files/iso13250-2nd-ed-v2.pdf

[XTM 01] *XML Topic Maps (XTM) 1.0,* TopicMaps.Org Specification, 06 August 2001, available at http://www.topicmaps.org/xtm/1.0/xtm1-20010806.html

[XMLS2 04] *XML Schema Part 2: Datatypes Second Edition*, World-Wide Web Consortium (W3C) Recommendation, 28 October 2004, available at http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/