TM/RDF Interoperability in Practice

A Tutorial

GOALS:
Understand topic maps, relation to RDF, and be able to convert data

Lars Marius Garshol
Development Manager, Ontopia
<larsga@ontopia.net>

Presented at ISWC 2004
2004-11-07
Who is talking?

- **Lars Marius Garshol**
  - Development manager at and co-founder of Ontopia
  - Co-author of the new ISO 13250 Topic Maps, parts 2 and 3
  - Co-editor of ISO 18048 Topic Map Query Language (TMQL)
  - Responsible for the Unicode support in the Opera web browser
  - Active open source developer in the XML community

- **Ontopia**
  - the leading topic map software vendor
  - Norwegian company headquartered in Oslo
  - main product: Ontopia Knowledge Suite (OKS)
Background for this tutorial

• Have been working full-time on Topic Maps since April 2000
  – standardization, software, applications, evangelization, ...
• RDF, Topic Maps, DAML, OIL, paper, December 2001
  – XML-based RDF-to-topic map mapping
• Topic Maps conversion toolkit based on RDF (2002-2003)
• Living with Topic Maps and RDF, paper, May 2003
  – RDF-based RDF-to-Topic maps mapping
  – TM-based TM-to-RDF mapping
• Implemented mappings in OKS and Omnigator (2003)
• User interface for creating mappings in Omnigator (2004)
Today's agenda

• Background
• Introduction to Topic Maps
• Comparing the models
• Approaches to conversions
• Understanding Topic Maps
• Converting topic maps to RDF
• Schema languages
• Query languages
• Future work
Preparations for the exercises

• Install Java, if you haven't already
  – JDK or JRE, version 1.3 or higher
• Install the Omnigator
• Put the example.rdf file in
  – OMNIGATOR/jakarta-tomcat/webapps/omnigator/WEB-INF/topicmaps
Background

Two technologies, one problem?
Some history
Why this tutorial?

- Topic Maps and RDF are considered by many to compete for the same space
  - Yet there is little understanding of how the two match up
  - There is also little communication between the two communities
- This is an attempt to
  - Explain how the two compare,
  - Position them relative to one another,
  - Build bridges between the communities, and
  - Teach a practical approach to interoperability
The big picture

TMQL  TMCL

Topic Maps

OWL
RDFS
SPARQL

RDF

XTM  HyTM  LTM

RDF/XML  n3
Technical comparison

• **Topic Maps and RDF**
  - are graph-based data models,
  - have well-defined identity tests and merging operators,
  - have XML-based interchange syntaxes (as well as human-friendly ones),
  - are standards, and
  - have (or will have) standardized schema and query languages

• **Differences**
  - RDF is lower-level than Topic Maps,
  - topic maps support reification, qualification/provenance/context, and n-ary relationships, and
  - Topic Maps distinguish different kinds of URI references
Comparison of goals and use

- **Topic Maps goals**
  - make information findable
  - make indexes mergeable
  - enable collocation of information
  - support “seamless knowledge”

- **Topic Maps uses**
  - portal infrastructure
  - classification/indexing
  - application integration
  - business process modelling
  - product data management
  - e-learning

- **RDF goals**
  - represent metadata on the web (RDF MS, Lassila & Swick)
  - unify metadata and data (MCF, Guha)
  - support data integration (Miller)
  - enable the Semantic Web (Berners-Lee, Miller, ...)

- **RDF uses**
  - portal infrastructure
  - application integration
  - document metadata
  - web agent applications
  - ???
Summing up the comparisons

- The technologies are similar, yet different
- The goals and visions are similar, yet different
- It is clear that the two are very close, yet significantly different
- In practice, they perform well at different things
- We will return to this
Where we stand today

• Topic Maps have been standardized and deployed
  – the standard is written and published, and the second edition nearly done
  – a number of commercial projects have been developed and deployed based on this standard
  – commercial and open source tools are available
  – a whole community has developed around the standard
  – work has begun on supporting standards

• RDF has been standardized and deployed
  – the situation is much the same as for topic maps

• In short, neither topic maps nor RDF will be going away soon
A brief history of topic maps

• Roots go back to early 1990’s
  – O'Reilly/DEC indexing project
  – Davenport Group → CAPH

• Adopted as an ISO work item in 1996
  – Original editors: Steve Newcomb, Michel Biezunski, and Martin Bryan

  – Syntax based on SGML, model based on HyTime

• Web Standard (XML Topic Maps 1.0, 2001)
  – XML version for use on the Web
  – Adopted by ISO, October 2001

• Second Edition
  – ISO 13250: 2003 (includes XTM)

• Revised Edition
  – Multipart standard including data model, query language and constraint language (2004-2005)
A brief history of RDF

• **Ramanathan V. Guha**
  – created MCF (Meta Content Framework) at Apple '95-'96
  – common representation for metadata and data; used for HotSauce
  – model and text-based syntax submitted to IETF in '96

• **World Wide Web Consortium**
  – Guha and Bray created an XML syntax for MCF in June '97
  – work was already ongoing on PICS-NG, a general metadata standard
  – RDF = PICS-NG + XML-MCF
  – first working draft August '97, final recommendation February '99
  – second edition of core specifications early 2004
Timeline

- '91: SOFABED model
- '92: Topic navigation maps
- '93: MCF
- '94: PICS-NG
- '95: MCF-XML
- '96: RDF WD
- '97: RDF Rec
- '98: Standard finished
- '00: RDF Rec
- '01: ISO 13250:2003
- '02: OWL
- '03: XTM to ISO
- '04: XTM 1.0
- '05: TopicMaps.Org

Davenport Group

http://www.ontopia.net
Introduction to topic maps

The TAO of Topic Maps
The Findability Problem

• Ask yourself:
  – Is this problem really “new”? Didn’t it exist before the advent of computers?
  – How would you go about locating a specific piece of information in a book – short of reading it from cover to cover?

• Isn’t that what (back-of-book) indexes are for?
  – An index is an information retrieval device
  – Publishers have traditionally set great store by indexes:
    • “There is no book … so good that it is not made better by an index, and no book so bad that it may not by this adjunct escape the worst condemnation” (Sir Edward Cook)

• Indexes and maps
  – The task of the indexer is to chart the topics of the document and to present a concise and accurate map for the readers
    • “A book without an index is like a country without a map”
What is an Index, Really?

Madama Butterfly, 70-71, 234-236, 326
Puccini, Giacomo, 69-71
soprano, 41-42, 337
Tosca, 26, 70, 274-276, 326

topics (in fact, names of Topics)
page numbers (locators for Occurrences)
A More Complex Index

*Cavalleria Rusticana*, 71, 203-204

Mascagni, Pietro (composer)

*Cavalleria Rusticana*, 71, 203-204

Rustic Chivalry, see *Cavalleria Rusticana*

singers, 39-52

See also individual names

baritone, 46

bass, 46-47

soprano, 41-42, 337

tenor, 44-45

+ multiple indexes

+ other conventions

**Additional concepts:**

- *Index of names*
- *Index of places*
- *Index of subjects*

**topic types**

**occurrence types**

**topics with multiple names**

associations between topics
The Key Features Are

Topics
- named “subjects of discourse”
- may have multiple names
- may be typed

Associations
- relationships between topics

Occurrences
- information relevant to a topic
- may be typed
- pointed to via locators

These are also the key constructs in the topic map model!
Glossaries

- **bass:** The lowest of the male voice types. Basses usually play priests or fathers in operas, but they occasionally get star turns as the Devil.

- **diva:** Literally, “goddess” – a female opera star. Sometimes refers to a fussy, demanding opera star. See also prima donna.

- **first lady:** See prima donna.

- **Leitmotif** (German, “LIGHT-mo-teef”): A musical theme assigned to a main character or idea of an opera; invented by Richard Wagner.

- **prima Donna** (“PREE-mah DOAN-na”): Italian for “first lady”. The singer who plays the heroine, the main female character in an opera; or anyone who believes the world revolves around her.

- **soprano:** The female voice category with the highest notes and the highest paycheck.

- Glossaries have a different purpose than indexes:

- The purpose is not to provide pointers to every occurrence of a topic...

- ...but rather to provide one specific occurrence type – the definition

- Therefore, instead of using locators (page numbers) to point to the definition...

- ...the definition is simply placed in-line.

- It looks different on paper, but the underlying model is exactly the same
## Thesauri

<table>
<thead>
<tr>
<th>soprano</th>
<th>definition</th>
<th>The highest category of female (or artificial male) voice</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>broader terms</td>
<td>vocalist, singer</td>
</tr>
<tr>
<td></td>
<td>narrower terms</td>
<td>lyric soprano, dramatic soprano, coloratura soprano</td>
</tr>
<tr>
<td></td>
<td>related terms</td>
<td>mezzo-soprano, treble</td>
</tr>
</tbody>
</table>

*Note: The associations are typed!*
The Two-Layer Model of Topic Maps

- The core concepts of Topic Maps are based on those of the back-of-book index
- The same basic concepts have been extended and generalized for use with digital information
- Envisage a 2-layer data model consisting of
  - a set of information resources (below), and
  - a “knowledge map” (above)
- This is like the division of a book into content and index
(1) The Information Layer

- The lower layer contains the content
  - usually digital, but need not be
  - can be in any format or notation
  - can be text, graphics, video, audio, etc.

- This is like the content of the book to which the back-of-book index belongs
(2) The Knowledge Layer

The upper layer consists of topics and associations

- **Topics** represent the subjects that the information is about
  - Like the list of topics that forms a back-of-book index

- **Associations** represent relationships between those subjects
  - Like “see also” relationships in a back-of-book index

![Diagram of knowledge layer]

- (born in) Puccini → Lucca
- (composed by) Lucca → Tosca
- (composed by) Tosca → Madame Butterfly
(3) Occurrences

- The two layers are linked together
  - **Occurrences** are relationships with information resources that are pertinent to a given subject
  - The links (or locators) are like page numbers in a back-of-book index

![Diagram showing the relationship between Puccini, Tosca, Madame Butterfly, and Lucca, with arrows indicating occurrences and relationships between the knowledge layer and information layer.]
Are associations directional?

Puccini compositions Tosca
No, They Use *Roles*

Diagram:

- **Puccini** (composer)
- **Tosca** (work)

Arrows indicate:
- Composed by
Association Role Types

N.B.
role == association role and
role type == association role type
N-ary associations

Parenthood

child

father

child

mother

Lars Marius

Knut

Bjørg
The Omnigator

- An Omnivorous Topic Map Navigator
  - The Omnigator will Eat Anything (provided it’s a topic map!)
  - Any Ontology: including your own
  - Just drop your own topic map into the Omnigator directory and away you go!
  - The Omnigator makes “reasonable sense” out of any “reasonably sensible” topic map

- And it’s Free!
  - Download it from the Ontopia web site
    - http://www.ontopia.net
  - Or view it online at
    - http://www.ontopia.net/omnigator

- Built using Ontopia’s flagship product
  - The Ontopia Knowledge Suite (OKS)
  - A complete Java toolkit for building topic map applications
  - Academic licenses available from sales@ontopia.net
How the Omnigator Works

- J2EE Web Server (e.g. Tomcat)
- topic map
- Omnigator
- Ontopia Knowledge Suite
- http
- <HTML> pages

http://www.ontopia.net
The topic map mindset

• Note how different the topic map outlook is from the RDF one
• There is no talk about metadata; focus is entirely on subjects
  – metadata is often represented in topic maps, but it’s not the focus
• Focus very much on findability
  – application integration, application data representation, metadata, etc all done and useful, but not in focus
  – the same applies to inferencing etc
Comparing the models

Things and symbols
Assertions
Assertions about assertions
Identity
Things

• The heart of RDF and topic maps is the same:
  - symbols representing real-world things
• Both RDF and topic maps consist of statements about these things
Terminology

- **Topic maps have topics**
  - all topics are equal (almost)
- **Topics represent subjects**
  - subjects are explicitly defined as “anything whatsoever, regardless of whether it exists or has any other specific characteristics, about which anything whatsoever may be asserted by any means whatsoever”

- **RDF has nodes**
  - URI references
  - blank nodes
  - literals
- **Nodes represent resources**
  - not necessarily documents

<table>
<thead>
<tr>
<th>Reference</th>
<th>Topic maps</th>
<th>RDF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symbol</td>
<td>Topic</td>
<td>Node</td>
</tr>
<tr>
<td>Thing</td>
<td>Subject</td>
<td>Resource</td>
</tr>
</tbody>
</table>
Assertions

- **RDF has one kind of assertion: the statement**
  - subject, predicate, object
- **Topic maps have three kinds**
  - (1) Names
  - (2) Occurrences
  - (3) Associations
(1) Names

- **foaf:name**: "Lars Marius Garshol"
- **dc:title**: "RDF/TM Interopera...", "Tutorial"
- **rdfs:label**: "Tutorial"

- **basename**: "Lars Marius Garshol"
(2) Concept-resource relationships

```
foaf:name

“Lars Marius Garshol”

foaf:homepage

http://www.garshol.priv.no

basename

“Lars Marius Garshol”

homepag

http://www.garshol.priv.no
```
(3) General relationships

- foaf:name: "Lars Marius Garshol"
- dc:creator: "RDF/TM Interopera..."
- dc:title: "RDF/TM Interopera..."
- creator: "Lars Marius Garshol"
- creator-of: "RDF/TM Interopera..."
- basename: "RDF/TM Interopera..."
 Assertions about assertions

• **This has many uses**
  – record metadata about assertions (created by, last modified, ...)
  – treat assertions as first-class things

• **There are also many common specialized uses**
  – provenance
  – authority
  – context
Assertions about assertions (2)

- RDF doesn't really do this
  - reification has no official support
  - it also very cumbersome
- However, people still use
  - reification,
  - contexts, and
  - named graphs
- Many tools have unofficial extensions for this
  - TriX syntax also does
  - SPARQL, too
- Topic maps have
  - reification, and
  - scope
- Reification is reification as we are used to it
  - less cumbersome than in RDF
- Scope is a set of topics
  - these together define a context
  - this supports provenance, context, and named graphs, plus more
Supporting Context through Scope

• Topic Maps are about representing knowledge
• Knowledge is not absolute; it has a contextual aspect
• Context sensitivity is handled through the concept of **scope**
• **Scope makes it possible to**
  – Cater for the *subjectivity* of knowledge
  – Express *multiple viewpoints* in one knowledge base
  – Provide *personalized views* for different groups of users
  – Track the *source* or provenance of knowledge during merging
  – Record the *usage context* of a name (language, corporate culture, domain, ...)
  – Represent which *authority* claimed that something is true

• *(Scopes are defined as sets of topics)*
How Scope Works

• **Topics have “characteristics”**
  - Its **names** and **occurrences**, and the roles it plays in **associations** with other topics

• **Every characteristic is valid within some context (scope), e.g.**
  - the **name** “Norge” for the topic Norway in the scope “Norwegian”
  - a certain information **occurrence** in the scope “technician”
  - a given **association** is true in the scope (according to) “Authority X”

Filtering by scope
Identity

• Nodes, in topic maps and RDF, have global identity
  – that is, there are rules for when nodes *in different graphs* are the same
  – there are also rules for how to merge graphs
• In RDF there are three cases
  – blank nodes, which cannot be compared across graphs,
  – URI references, which are identical if the URIs are equal, and
  – literals, which have surprisingly complex rules for identity
• In topic maps there are also three methods
  – the rule is: topics are merged if they have the same subject
  – we need more space to discuss how this is determined, however
Identifying information resources

- Topics representing information resources are easy
- Attach a URI to the topic as the subject locator of the topic
- This means that the topic represents the resource
- That is, the resource is the subject of the topic

```xml
<topic id="ontopia-hp">
  <subjectIdentity>
    <resourceRef xlink:href="http://www.ontopia.net/"/>
  </subjectIdentity>
  ...
</topic>
```
Identifying non-resources

- **Topics representing other things are harder**
  - Concepts do not have URIs
- **Solution:**
  - Create a resource describing the concept
  - Refer to the concept as the description
- **Resource = Subject indicator**

```xml
<topic id="ontopia-hp">
  <subjectIdentity>
    <subjectIndicatorRef xlink:href="http://www.ontopia.net"/>
  </subjectIdentity>
...
</topic>
```
Subject locator ≠ subject indicator

- These two topics refer to the same resource, yet remain distinct
- This maintains a distinction between
  - information resources, and
  - everything else
- In RDF this distinction does not exist
  - using blank nodes and a tm:indicator property it can be approximated
  - however, properties must have URIs
  - this has been the source of much controversy ("identity crisis", etc)
Merging in topic maps

• When adding information to a topic map
  - topics which have the same subject locators or indicators are merged
  - duplicate information is removed

• Merging two topic maps behaves the same way
  - effectively the same as adding the information in both into a new map

• Thus, the same topic can have many identifiers
  - this expresses the fact that the identifiers identify the same thing
  - different from the RDF approach
Approaches to interoperability

Modelling topic maps in RDF
Model-level mappings
Vocabulary-level mappings
Modelling topic maps in RDF

• If RDF is lower-level than topic maps, perhaps topic maps can be modelled in RDF?
• This means creating an RDF vocabulary for representing topic maps
• Many people have tried this approach
  – *An RDF Schema for topic maps*, Lars Marius Garshol, 2002
• Basically, it works, but...
Modelling topic maps in RDF
Problems

• **This approach leads to data that**
  – is very heavyweight,
  – is cumbersome to use in RDF,
  – is formulated in terms of the topic map model, rather than the domain vocabulary,
  – interoperates poorly with other RDF data (uses TM vocabulary),
  – is awkward to query with RDQL (uses TM vocabulary),
  – cannot be modelled with RDFS/OWL (uses TM vocabulary)

• **In short, this approach is not the right one**
Finding my creations

SELECT ?c
WHERE (?m, <foaf:name>, “Lars ...”), (?c, <dc:creator>, ?m)

SELECT ?c
WHERE (?m, <tm:basename>, ?n),
(?n, <tm:value>, “Lars ...”),
(?r1, <tm:player>, ?m),
(?r1, <rdf:type>, <:creator>),
(?a, <tm:role>, ?r1),
(?a, <rdf:type>, <:created-by>),
(?a, <tm:role>, ?r2),
(?r2, <rdf:type>, <:creation>),
(?r2, <tm:player>, ?c)
The ideal conversion result

• Ideally, conversions should produce result data formulated in the same vocabulary as the source data
• That is, if the source data is defined in terms of
  – document, title, creator, person, email address
• ...it would be nice if the result also were, instead of being defined as
  – topic, base name, association, topic, occurrence
• Also, tools applicable to the source data should, if similar tools exist for the target technology, be equally applicable to the result data
• With that in mind, let's look at a different approach
Model-level mappings

- Given how close topic maps and RDF are, it seems it should be possible to create model-level mappings
  - That is, it should be able to relate RDF model constructs into topic map constructs
  - Converting RDF into topic maps would then be simple, given some basic conventions
- To some extent this works
  - RDF nodes and topics are almost the same
- Reality check:
  - what does the first statement below map to? base name? occurrence?
  - and the second? association? occurrence?
Vocabulary-level mappings

• However, if we knew the meaning of ex:foo and ex:bar we would be able to do the mapping
  – We would know whether ex:foo is a name or an occurrence
  – Similarly, whether ex:bar is an occurrence or an association

• So, what if we created vocabulary mappings?
  – for each property we could declare what it mapped to
  – the declarations could be RDF statements

• This is what the RTM (RDF-to-TM) vocabulary does
Approach taken
How the mapping works

• For each statement, create a topic for the subject
• If the subject has a URI, set that as the subject indicator
• Look at the <rtm:maps-to> property on the predicate to see what to map it to
• Other properties supply more information, as shown on right

<table>
<thead>
<tr>
<th>Mapping to</th>
<th>Information needed</th>
<th>Legal node types</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Scope</td>
<td>Literal</td>
</tr>
<tr>
<td>Occurrence</td>
<td>Scope, type</td>
<td>Literal, URI</td>
</tr>
<tr>
<td>Association</td>
<td>Scope, type, roles</td>
<td>URI, blank</td>
</tr>
<tr>
<td>Subject locator</td>
<td></td>
<td>URI</td>
</tr>
<tr>
<td>Subject indicator</td>
<td></td>
<td>URI</td>
</tr>
</tbody>
</table>
The RTM vocabulary

- The namespace is http://psi.ontopia.net/rdf2tm/#
- Core property: rtm:maps-to
  - domain: rdf:Property, range: rtm:Construct
  - values
    - rtm:basename
    - rtm:occurrence
    - rtm:association
    - rtm:instance-of
    - rtm:subject-identifier
    - rtm:subject-locator
    - rtm:source-locator
- Additional property: rtm:type
  - domain: rdf:Property, range: rdfs:Resource
  - type of created topic characteristic defaults to topic representing property
  - this property can be used to override the default
The RTM vocabulary (2)

- **The rtm:in-scope property**
  - domain: rdf:Property, range: rdfs:Resource
  - repeatable; used to attach fixed scoping topics to the created characteristic
  - mainly used to attach scope to base names

- **The rtm:subject-role property**
  - domain: rdf:Property, range: rdfs:Resource
  - required for properties mapped to associations
  - sets the role played by the topic that is the subject of the RDF statement

- **The rtm:object-role property**
  - domain: rdf:Property, range: rdfs:Resource
  - required for properties mapped to associations
  - sets the role played by the topic that is the value of the RDF statement
Exercise #1

- Start the Omnigator
- Go to http://localhost:8080/omnigator/
  - then click on example.rdf
- Use the RDF2TM plug-in to configure the mapping
  - it must be turned on: go to Manage, then Plug-ins
- Then reload the file to see the results of your mapping
- Keep tuning the mapping until you get it right
- Look at the mapping file to learn how it works
  - OMNIGATOR/jakarta-tomcat/webapps/omnigator/WEB-INF/topicmaps/mapping.rdff
Exercise #2

• Extend the mapping.rdff file with mappings for the FOAF vocabulary
  – manually, that is
  – note that for associations you must supply the role types...

• Key properties
  – foaf:name
  – foaf:mbox
  – foaf:homepage
  – foaf:knows

• Copy foaf.rdf to the topicmaps directory
• Go to “Manage”, press “Refresh sources”
• Then click foaf.rdf to see the result
Learning topic maps

The XTM and LTM syntaxes
Topic Map Syntaxes

- **HyTM (HyTime Topic Maps)**
  - Topic Maps as originally defined by ISO
  - An architecture expressed in terms of SGML and HyTime
  - *Will be left out of the next version of ISO 13250*

- **XTM (XML Topic Maps)**
  - Defined by TopicMaps.Org in 2001 and later adopted by ISO
  - Uses XML and Xlink
  - Will become ISO 13250-2: Topic Maps – XML Syntax
  - *Easy to understand but very verbose*

- **LTM (Linear Topic Map Notation)**
  - Defined by Ontopia in 2001
  - Also supported by TM4J and Perl::XTM
  - *A simple text syntax for rapid prototyping*
<?xml version="1.0" encoding="ISO-8859-1"?>
<topicMap
   xmlns="http://www.topicmaps.org/xtm/1.0/"
   xmlns:xlink="http://www.w3.org/1999/xlink"
>

<!-- topics, associations, and mergeMap elements go here -->

</topicMap>
/* topics, associations, and occurrences go here */
Exercise #3

• Create an LTM file in the topicmaps directory
  – choose whatever name for it you like, but make the extension .ltm
  – write a nice comment at the top
• Go to the Manage page, press Refresh Sources
• Enter the topic map and see that it is empty
Topics: XTM Syntax

<topic id="italy">
...
</topic>

<topic id="puccini">
...
</topic>
Topics: LTM Syntax

[italy]

[puccini]
Exercise #4

• Create two topics in the topic map
  – one for yourself
  – one for the class 'person'
• Use the reload plug-in in the Omnigator to see the result
<topic id="la-scala">
  <baseName>
    <baseNameString>Teatro alla Scala</baseNameString>
    <variant>
      <parameters>
        <subjectIndicatorRef xlink:href="http://www.topicmaps.org/xtm/1.0/core.xtm#display"/>
      </parameters>
      <variantName>
        <resourceData>La Scala</resourceData>
      </variantName>
    </variant>
    <variant>
      <parameters>
        <subjectIndicatorRef xlink:href="http://www.topicmaps.org/xtm/1.0/core.xtm#sort"/>
      </parameters>
      <variantName>
        <resourceData>scala, teatro alla</resourceData>
      </variantName>
    </variant>
  </baseName>
</topic>
Topic Names: LTM Syntax

[topic-id = basename; sortname?; dispname?]

[la-scala = "Teatro alla Scala"; "scala, teatro alla"; "La Scala"]
Exercise #5

- Add a name for yourself, with a sort name if you like
- Also add a name for the 'person' class
- Reload and view
Topic Types: XTM Syntax

<topic id="opera">
  ...
</topic>

<topic id="tosca">
  <instanceOf>
    <topicRef xlink:href="#opera"/>
  </instanceOf>
</topic>

<topic id="boito">
  <instanceOf>
    <topicRef xlink:href="#composer"/>
  </instanceOf>
  <instanceOf>
    <topicRef xlink:href="#librettist"/>
  </instanceOf>
</topic>
Topic Types: LTM Syntax

[topic-id : topic-type = basename; sortname?; dispname?]

tosca : opera

boito : composer librettist
Exercise #6

• Make yourself an instance of the 'person' class
• Reload and view
<!- Refer to a resource as subject: -->
<topic id="foo">
  <subjectIdentity>
    <resourceRef xlink:href="http://www.ontopia.net"/>
  </subjectIdentity>
  <baseName>
    <baseNameString>The Ontopia Website</baseNameString>
  </baseName>
</topic>

<!- Refer to a subject indicator: -->
<topic id="bar">
  <subjectIdentity>
    <subjectIndicatorRef xlink:href="http://www.ontopia.net/about.html"/>
  </subjectIdentity>
  <baseName>
    <baseNameString>Ontopia</baseNameString>
  </baseName>
</topic>
Subject identity: LTM Syntax

[topic-id = names %subject-locator-URL]
[topic-id = names @subject-indicator-URL]

/* Refer to a resource as subject: */
[foo = "The Ontopia Website" %"http://www.ontopia.net"
]

/* Refer to a subject indicator: */
[bar = "Ontopia" @"http://www.ontopia.net/"
Exercise #7

• Add a subject indicator for the person topic
  – http://xmlns.com/foaf/0.1/Person

• Reload the topic map to see the result
<topic id="la-boheme">
  <occurrence>
    <instanceOf><topicRef xlink:href="#homepage"/></instanceOf>
    <resourceRef>
      xlink:href="http://www.opera.it/Opere/La-Boheme/La-Boheme.html"/>
    </resourceRef>
  </occurrence>
  <occurrence>
    <instanceOf><topicRef xlink:href="#premiere-date"/></instanceOf>
    <resourceData>1896 (1 Feb)</resourceData>
  </occurrence>
</topic>
Occurrences: LTM Syntax

{topic-id, occurrence-type, (URL | data)}

{la-boheme, homepage,
  "http://www.opera.it/Opere/La-Boheme/La-Boheme.html"}
{la-boheme, premiere-date, [[1896 (1 Feb)]]}
Exercise #8

- Add topics 'phone' and 'mbox'
- Give them subject indicators in the FOAF namespace
- Then add for yourself
  - an mbox occurrence with your email address as a mailto: URI
  - a phone occurrence with your phone number in it as a string
- Reload topic map and view
Associations: XTM Syntax

<association>
  <instanceOf>
    <topicRef xlink:href="#composed-by"/>
  </instanceOf>

  <member>
    <roleSpec>
      <topicRef xlink:href="#composer"/>
    </roleSpec>
    <topicRef xlink:href="#puccini"/>
  </member>

  <member>
    <roleSpec>
      <topicRef xlink:href="#work"/>
    </roleSpec>
    <topicRef xlink:href="#tosca"/>
  </member>

</association>
Associations: LTM Syntax

\textit{assoc-type} ( role-player : role-type, role-player : role-type )

\textit{composed-by} ( puccini : composer, tosca : work )
\textit{born-in} ( puccini : person, lucca : place )

\textit{composed-by} ( puccini, tosca )
\textit{born-in} ( puccini, lucca )

Note: When omitted, the role type will be assumed to be identical to the type of the role playing topic. This can be a useful short-hand, but it is not always what you want.
Exercise #9

• Add a new person, someone you know
• Add a new topic 'knows'
• Create an association between you and the person you know
  – you can both play the role 'person'
• Reload to see the result
From topic maps to RDF

From house to bricks
Going the other way

• Given that we must add information to go from RDF to topic maps, one should expect that going the other way would be painless
• Not so
• There are a few things one needs to know
  – which properties are used to represent names?
  – what to do with scope?
  – what direction should binary associations take?
• However, this can be solved in a similar way
  – attach the necessary information as declarations
  – represent them in the topic map itself
Rough algorithm

• Each topic becomes an RDF node
  – if it has a subject locator, that becomes the URI
  – if not, but it has a subject indicator, that becomes the URI
  – if it has neither it becomes a blank node
  – additional subject indicators are handled with owl:sameAs

• Base names become RDF statements
  – the property is chosen using the name-property association for the topic type

• Occurrences become RDF statements
  – the property is the occurrence type

• Binary associations become RDF statements
  – the preferred-role association is used to decide which role player is the subject;
    if missing one is chosen at random

• N-ary associations become RDF nodes
  – type recorded with rdf:type
  – each role becomes a statement; property is role type; player is value
Reification and scope

- This is the hardest bit to handle
- Current solution
  - reification represented using RDF reification
  - scope represented using RDF reification and a special tm2rdf:scope property
- The problem with this is that it's horribly verbose
  - options in Ontopia exporter to enable/disable this
- Interestingly, various RDF extensions would solve the scope part
  - however, RDF/XML does not support this...
The TMR vocabulary

- **Namespace:** http://psi.ontopia.net/tm2rdf/#
- **name-property**
  - name-property(rdf-Property : type, rdfs-Label : property)
- **preferred-role**
  - preferred-role(subclass-of : association-type, subclass : role-type)
Exercise #10

• Add a topic 'name', and create subject indicator for it in the FOAF namespace
• Add an association making 'name' the name property for 'person' topics
• Export the result as RDF using the 'Export' plug-in, and study the result
Schema and ontology languages

*RDF Schema*

*OWL*

*TMCL*
Schema-level interoperability

• We've now covered data interoperability, which is the most important part
• However, schema interoperability is also important
  – Life becomes much simpler if schemas do not have to be rewritten by hand
• The topic map standard corresponding to RDFS and OWL is Topic Maps Constraint Language (TMCL)
  – This standard is not yet finalized, however
• We will consider how to preserve RDF schema information in topic maps
  – The opposite will have to wait until TMCL is ready
A quick look at TMCL

• The TMCL standard is still being worked on
• Declarative constraints in a specialized syntax
  – these are restrictions on what may be said in the topic map
  – violations are returned in a structured form
• Can be used to
  – validate a topic map
  – filter a topic map (ie, return only what is valid)
  – learn about the structure of the topic map
• Only the basic part has been worked on so far
  – OWL-like features may be added once the core is in place
Converting RDF Schema

• The classes are easy: we can just let them be converted directly
• The properties need to be mapped, however
  – rdfs:label  Maps to base name
  – rdfs:comment  Maps to occurrence
  – rdfs:subClassOf  Maps to association (should have its type translated)
  – rdfs:domain  Maps to association
  – rdfs:range  Maps to association
  – rdfs:isDefinedBy  Maps to occurrence
  – rdfs:seeAlso  Maps to occurrence
  – rdfs:subPropertyOf  Maps to association
• This mapping preserves the RDFS structure in a topic map, but requires validation logic to be implemented on the topic map side
RDF Schema validation using queries

- A crude way to implement RDFS validation in TMs is using queries
- Queries can be written to find
  - all topic characteristics belonging to a topic of a type that is not in the domain of the characteristic's type
  - all topic characteristics whose value does not match the range of the characteristic's type
- This is quite limited, of course, but then RDFS is quite limited
A quick look at OSL

- **Ontopia Schema Language** – Ontopia's short-term schema language
- **Simple constraint language reminiscent of DTDs**
  - constraints on topic classes and association classes
  - cardinality constraints on characteristics and their types and scopes
  - no ordering, no conditionals, no data typing
- **Has proven itself to work, but not seen wide use**
Converting RDF Schema to OSL

- Possible, and actually quite easy
- `rdfs:Class` becomes a topic class
  - properties whose domain is that class get class constraints
  - `rtm:maps-to` is used to work out what type of constraint to create
  - `rtm:in-scope` could be used to add scope rules
- Properties that map to `rtm:association` become association classes
  - role constraints set using `rtm:subject-role` and `rtm:object-role`
Converting OWL

- OWL consists of many different kinds of statements which must be handled differently
- Metadata
  - this is just instance-level statements, and can be converted easily
- Restrictions
  - value, cardinality, and disjointness; these can be converted, too
  - some of it might go into the RDF Schema-to-OSL conversion
- Semantic annotations
  - these are more problematic; not all are needed in topic maps
Query languages

RDQL
TMQL
tolog
The state of standardization

- **W3C Data Access WG**
  - working on RDF query language

- **Current draft: SPARQL**
  - based on RDQL
  - very simple graph-matching language
  - inferencing to be done in level below querying

- **ISO 18048: TMQL**
  - to become standard topic map query language

- **No draft yet**
  - to be based on tolog
  - basic outline already known
  - graph-matching with inference
  - also path capabilities
  - also output construction ability a la XQuery
tolog

- **Ontopia's topic map query language**
  - Basically Datalog adapted for use with topic maps
  - Has been in use for ~2 years
  - Drives a number of commercial applications
  - Also implemented in the open source TM4J engine

- **Basic features**
  - Graph matching (Datalog clauses)
  - AND/OR/NOT support in query body (and rule bodies)
  - Inference rules
  - Aggregate functions (a la SQL)
  - Sorting (a la SQL)

- **Coming extensions**
  - String predicates and comparator predicates
Standards family harmonization

- SPARQL and tolog share a common core
  - the graph matching part
- RDF querying can be done in tolog
- In theory it is possible to create a common standard
- In practice the political will to do so appears to be wholly missing
Cross-query example

• A cross TM/RDF query:
  - using foaf for "http://xmlns.com/foaf/0.1/" as rdf
    xc for "http://psi.ontopia.net/xmlconf/#" as indicator
  select $B from
    foaf:mbox($A, "mailto:larsga@ontopia.net"),
    foaf:knows($A, $B),
    foaf:mbox($B, $BMAIL),
    xc:email($BTM, $BMAIL),
    xc:employed-by($BTM : xc:employee, $C : xc:employer),
    xc:homepage($C, "http://www.empolis.com")?

• Note the use of the email address to do the join across the TM/RDF boundary
Translating SPARQL queries with RTM

SELECT ?c
WHERE (?m, <foaf:name>, "Lars ..."), (?c, <dc:creator>, ?m)

SELECT $C
FROM
  name($M, "Lars Mar..."),
  dc:creator($M : creator, $C : creation)?
What the future holds

SWBPD
OWL/TMCL
Query languages
Within the Semantic Web Best Practices [...] Working Group a Task Force is being set up to work on TM/RDF interoperability
- most likely it will build on the RTM approach
- not clear yet exactly what the scope of the work will be
- there may be 2-3 TFs, we don't know yet

The task force is currently looking for participants
- http://www.w3.org/2001/sw/BestPractices/RDFTM/
- to join, you must be a member of W3C and SWBPD WG
- most likely it will be created during the upcoming November 18 telecon
OWL/TMCL

• OWL is done, so there's unlikely to be any TM-related changes to it
• TMCL is being created, but with a different focus
  – focus is much more on validation
  – also some on declarative semantics
  – once the core is done we may work on adding OWL-like features
  – mapping to OWL is likely to be known when TMCL is created
Query languages

- SPARQL and TMQL are being created in parallel at the moment
- There is unlikely to be any work on interoperability or harmonization here
  - Main reason: lack of political will
- However, it's not clear whether interoperability work really is needed
Thank you!

- The slides from this talk

- About topic maps
  - [http://www.topicmap.com](http://www.topicmap.com) (English)
  - [http://www.knowledge-synergy.com](http://www.knowledge-synergy.com) (Japanese)

- Topic maps standardization
  - [http://www.isotopicmaps.org](http://www.isotopicmaps.org)

- About RDF and topic maps (RTM, TMR, +++)
  - [http://www.ontopia.net/topicmaps/materials/tmrdf.html](http://www.ontopia.net/topicmaps/materials/tmrdf.html)

- The Omnigator
  - [http://www.ontopia.net/omnigator/](http://www.ontopia.net/omnigator/)

- Questions
  - <larsga@ontopia.net> (English)
  - <motom@green.ocn.ne.jp> (Japanese)