Published on XML.com http://www.xml.com/pub/a/2002/09/11/topicmaps.html
See this if you're having trouble printing code examples

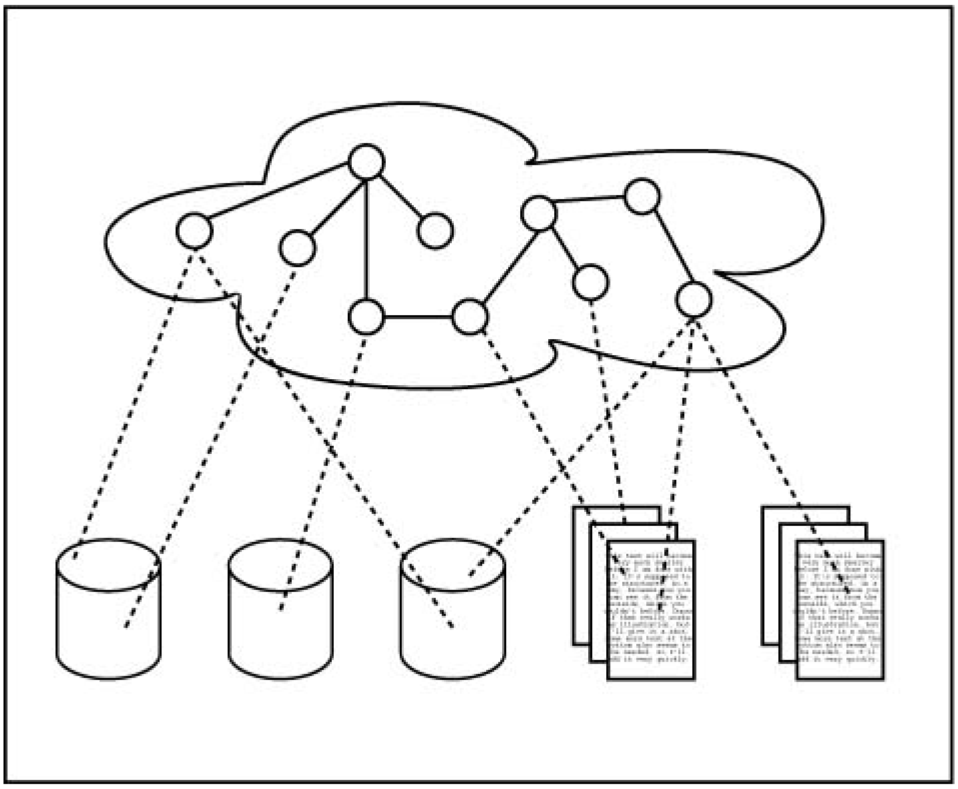
What Are Topic Maps?

By Lars Marius Garshol

Many years ago, I started looking into SGML and XML as a way to make information more manageable and findable, which was something I had been working on for a long time. It took me several years to discover that, although SGML and XML helped, they did not actually solve the problem. Later I discovered topic maps, and it seemed to me that here was the missing piece that would make it possible to really find what you were looking for. This article is about why I still think so.

What Topic Maps Do

When XML is introduced into an organization it is usually used for one of two purposes: either to structure the organization's documents or to make that organization's applications talk to other applications. These are both useful ways of using XML, but they will not help anyone find the information they are looking for. What changes with the introduction of XML is that the document processes become more controllable and can be automated to a greater degree than before, while applications can now communicate internally and externally. But the big picture, something that collects the key concepts in the organization's information and ties it all together, is nowhere to be found.



This is where topic maps come in. With topic maps you create an index of information which resides *outside* that information, as shown in the diagram above. The topic map (the cloud at the top) describes the information in the documents (the little rectangles) and the databases (the little "cans") by linking into them using URIs (the lines).

The topic map takes the key concepts described in the databases and documents and relates them together independently of what is said about them in the information being indexed. So when a document says "The maintenance procedure for part X consists of the following steps..." the topic map may say "Part X is of type Q and is contained in parts Y and Z and its maintenance procedure resides in document W". This means taking a step back from the details and focusing on the forest rather than the trees. Or, to put it another way, it means managing the meaning of the information, rather than just the information.

The result is an information structure that breaks out of the traditional hierarchical straightjacket that we have gotten used to squeezing our information into. A topic map usually contains several overlapping hierarchies which are rich with semantic cross-links like "Part X is critical to procedure V." This makes information much easier to find because you no longer act as the designers expected you to; there are multiple redundant navigation paths that will lead you to the same answer. You can even use searches to jump to a good starting point for navigation.

The most common use for topic maps right now is to build web sites that are entirely driven by the topic map, in order to fully realize the their information-finding benefits. The topic map provides the site structure, and the page content is taken partly from the topic map itself, and partly from the occurrences. This solution is perfect for all sorts of portals, catalogs, site indexes, and so on. Since a topic map can be said to represent knowledge about the things it describes, topic maps are also ideal as knowledge management tools.

This is by no means all topic maps can be used for, however. They can also be used to organize the content in content management systems (instead of the simple folder hierarchies and property-value metadata often used today), they integrate information from diverse sources (using merging), they can drive expert systems, and much much more. (This article will focus on information-finding, however, since it is an introductory article with limited scope and length. See Marc de Graauw's article about topic maps in B2B exchange for a different view on what topic maps can do.)

by Lars Marius Garshol

How topic maps work

By now you are probably wondering how all this works. The answer is surprisingly simple. At the heart of topic maps are *topics* (these are the circles in the diagram), which represent the things the topic map is about. In a topic map about XML, for example, one might expect to find topics representing 'the XML Recommendation', 'the W3C', and 'Tim Bray'.

The next step is the relationships between the topics, which in topic maps is modeled with *associations* (the lines between the topics). Associations are typed, which means that we can say that the relationship between the XML Recommendation and the W3C is one we might call 'publishing', while the relationship between the Recommendation and Tim Bray is one of 'authorship'.

Associations have one unusual feature. Each topic involved in the association is said to play a *role*, which is defined by its *role type*. So in the 'authorship' association Tim Bray plays the role of 'author' while the XML Recommendation plays the role of 'work'. This means that the statements 'Tim Bray wrote the XML Recommendation' and 'the XML Recommendation was written by Tim Bray' are *the same statement* in topic maps. It is impossible to say the one without at the same time saying the other, and the association can be traversed in either direction. Associations need not be restricted to two topics. Relationships like 'Tim Bray represents Textuality in the W3C' can be expressed using an association with three roles, and it's no more difficult to express than than simpler associations.

The last main feature of topic maps is what is known as *occurrences*, which are information resources relevant to a topic. For "Tim Bray", occurrences might be his home page, a portrait, CV, etc. Since occurrences may also be typed, these different kinds of resources can be distinguished. This means that when a user comes to a topic and wants more information about it, the user not only gets a set of links, but also knows what makes each link interesting.

The last point to note is that topics may also have types, and reasonable types for the example topics might be 'standard', 'standards body', and 'person'. Types in topic maps, however, are themselves topics, which means that anyone creating a topic map can choose what topic types, association and role types, and occurrence types they want to use. As a result, the model is infinitely extensible and adaptable and can capture just about any kind of information.

One of the benefits of the topic maps approach is that usually, when you are creating a topic map for a set of existing data (documents or databases), you will find that a number of important concepts are touched on in the dataset, without actually having identities of their own. One example of this might be if you were to topic map the W3C web site. It is a fairly well structured site, but let's say you are looking for information on XML Base. You can easily find the specification, and using the search you can find various pages that mention it. And that's it. With a topic map, on the other hand, you'd have a topic representing 'XML Base', the concept. Going to that topic would present you with information like 'XML Base is an XML vocabulary' (topic type), 'XML Base uses XML namespaces' (association), 'XML Base is used by XHTML' (association), 'the XML Base specification is here' (occurrence), and so on. This makes it easier to find what you are looking for, and also to learn about it once you have found it.

Making a topic map

Before we look at the practicalities, a little background is needed. Topic maps are an ISO standard, published as ISO/IEC 13250 in 2000. That standard defines the basic model and an SGML-based syntax for it, which uses HyTime for linking, and is therefore known as HyTM. When the standard was published it was clear that something more web-optimized was needed, and so an ad-hoc organization known as TopicMaps.Org was formed to create a topic map syntax based on XML and URIs.

TopicMaps.Org published its XML Topic Maps (XTM) 1.0 specification in early 2001, and in October of the same year that syntax was accepted into the second edition of ISO 13250 as an annex. Today, XTM is the main topic map syntax and is supported by nearly all topic map tools. In this article we use XTM as the example syntax.

To create a topic map for the example above we can start by defining topics for the three topic types. This is done as follows:

```
<topicMap xmlns="http://www.topicmaps.org/xtm/1.0/"
          xmlns:xlink="http://www.w3.org/1999/xlink">

  <topic id="person">
    <baseName>
      <baseNameString>Person</baseNameString>
    </baseName>
  </topic>

  <topic id="standards-body">
    <baseName>
      <baseNameString>Standards body</baseNameString>
    </baseName>
  </topic>

  <topic id="standard">
    <baseName>
      <baseNameString>Standard</baseNameString>
    </baseName>
  </topic>

</topicMap>
```

This gives us three topics suitable for use as topic types in our topic map. The `baseName` elements give the topics names that can be used to display the topics. The next step is to add one topic to be used as an occurrence type and our three instance topics, complete with names and occurrences. (The fragment below should be inserted inside the `topicMap` element above. The order of `topic` elements in topic maps is irrelevant.)

```
  <topic id="xml-rec">
    <instanceOf>
      <topicRef xlink:href="#standard"/>
    </instanceOf>
    <baseName>
      <baseNameString>The XML Recommendation</baseNameString>
    </baseName>
  </topic>

  <topic id="tim-bray">
    <instanceOf>
      <topicRef xlink:href="#person"/>
    </instanceOf>
    <baseName>
      <baseNameString>Tim Bray</baseNameString>
    </baseName>
  </topic>

  <topic id="homepage">
    <baseName>
      <baseNameString>Homepage</baseNameString>
    </baseName>
  </topic>

  <topic id="w3c">
    <instanceOf>
      <topicRef xlink:href="#standards-body"/>
    </instanceOf>
    <baseName>
      <baseNameString>World Wide Web Consortium</baseNameString>
    </baseName>
    <occurrence>
      <instanceOf>
        <topicRef xlink:href="#homepage"/>
      </instanceOf>
      <resourceRef xlink:href="http://www.w3.org"/>
    </occurrence>
  </topic>
```

The first two `topic` elements create topics for the XML Recommendation and Tim Bray, making them instances of the "standard" and "person" topic types we defined earlier. Notice how `instanceOf` is used to provide the class and `topicRef` is used to point to the topic that defines the class. Then we define the occurrence type "homepage" and finally a topic for the W3C, which is of type "standards-body" and even has a "homepage" occurrence. The `resourceRef` element inside the `occurrence` gives the URI of the resource that is the occurrence.

Finally, we are ready to create topics for the association and role types and create the corresponding associations in order to complete the topic map. The fragment below does just this.

```
<topic id="authorship">
  <baseName>
    <baseNameString>Authorship</baseNameString>
  </baseName>
</topic>

<topic id="author">
  <baseName>
    <baseNameString>Author</baseNameString>
  </baseName>
</topic>

<topic id="work">
  <baseName>
    <baseNameString>Work</baseNameString>
  </baseName>
</topic>

<association>
  <instanceOf>
    <topicRef xlink:href="#authorship"/>
  </instanceOf>

  <member>
    <roleSpec>
      <topicRef xlink:href="#author"/>
    </roleSpec>
    <topicRef xlink:href="#tim-bray"/>
  </member>

  <member>
    <roleSpec>
      <topicRef xlink:href="#work"/>
    </roleSpec>
    <topicRef xlink:href="#xml-rec"/>
  </member>
</association>
```

In order to keep this example at a reasonable length only the first association is provided here. It is of type "authorship" and has one topic playing the role of "author" (Tim Bray) and another playing the role of "work" (the XML Recommendation). The involvement of topics in the association is described using the `member` element type, and role types are defined with `roleSpec`. The participating topic is pointed to by the `topicRef` directly inside the `member`.

And that's it, really. This is a complete, although trivial, topic map. With a topic map browser you can actually browse through it, do queries on it, and so on. (See the links section at the bottom.) I've made a complete version for your convenience as well.

by Lars Marius Garshol

How to Use Topic Maps

There are two quetions here -- "how do I create a topic map?" and "how do I build an application once I have my topic map?" For creating a topic map there are four main approaches:

- Have humans author the topic maps manually. This usually gives very high-quality and rich topic maps, but at the cost of human labor. This is appropriate for some projects, while prohibitively expensive for others.
- Automatically generate the topic map from existing source data. This can give very good results if the existing data are well-structured (sound familiar?); if not, there are various natural-language processing tools that might help.
- Automatically produce the topic map from structured source data like XML, RDBMSs, LDAP servers, and more specialized applications.

This all sounds very fancy, but to produce a simple topic map (or even a fancy one) you don't need anything more than a text editor, and for automatic generation XSLT stylesheets can be used perfectly well. This won't be enough for all uses, of course, and therefore there is specialized software for topic map editing and automatic generation of topic maps.

The next step is the question of where the topic map is going to live, that is, be stored and maintained. For simpler applications storing XTM documents in files suffices, but for most real applications some form of database storage will be necessary. Most topic map implementations support some form of database storage, using various approaches.

Generally, the heart of every topic map application is what is known as the *topic map engine*, which is roughly equivalent to an RDBMS database engine, but designed for topic maps. This component knows how to import and export XTM (and other topic map syntaxes), store, update, and query topic maps, and so on. The engine will handle the storage, and any updates will happen through it. For example, applications that implement a topic map-driven portal will sit on top of the engine and use it to access the topic map. More advanced topic map implementations have special frameworks that simplify the work of creating such applications.

But Wait, There's More

There are three additional features in topic maps that you really need to know about. The first of these is *scope*, which can be attached to any name, occurrence, or association in a topic map. Basically, scope ca be attached to anything you can say in a topic map. Scope allows you to qualify a statement, but still express it.

Imagine you are making a topic map about languages, and basing it on the ISO 639 and Ethnologue lists of language codes. In that case you might want to record that ISO 639 assigns English to the Germanic language group, while Ethnologue considers it a West Germanic language. This can be done by scoping the association between English and Germanic with a topic representing ISO 639, and the association between English and West Germanic with a topic representing Ethnologue. Similarly, one might use scope to record that what Ethnologue calls Maldivian, ISO 639 calls Divehi.

Users can then choose to see all information in all scopes, or only those in particular scopes, basically tailoring their view of the world as they want to see it. One common use of scope is to create topic maps where the topics have names in multiple languages, allowing topic maps to be converted between languages at the press of a button.

Another interesting feature of topic maps is the use of URIs to identify subjects. A topic may have any number of *subject identifiers* (URIs) which identify the subject the topic is about. These URIs should point to resources which describe the subject to a human; the resources are known as *subject indicators*. This allows subjects to be uniquely identified across topic maps and the entire web. For example, the URI `http://www.topicmaps.org/xtm/1.0/core.xtm#superclass-subclass` uniquely identifies the subclassing association type.

This unambiguous identification of subjects is used in topic maps to *merge* topics that, through these identifiers, are known to have the same subject. Basically, what happens is that two topics with the same subject are replaced by a new topic that has the union of the characteristics (names, occurrences, and associations) of the two originals. There is in fact a well-defined procedure for automatically merging topic maps based on this rule. The combination of globally unique identifiers and the merging procedure makes integration of diverse information sources and reuse of information very much easier.

One aspect related to this is the *published subjects* activity of OASIS, which is developing guidelines for how to create, publish, and maintain subject indicators intended for wide usage. One example of this is well-known URIs for all the countries in the world (based on the ISO 3166 country codes), which will allow us to tell that when you say 'Norway' in one topic map and I say 'Norge' in another, we mean the same thing. More on this in a future article.

Putting Topic Maps in Context

A question you may be asking yourself at this point is how something like topic maps fits into the larger family of XML standards. As I hinted at the beginning, topic maps are really an add-on to XML, something that adds extra value beyond what XML itself can do. You can in fact use topic maps without using XML at all. In a very real way, the two standards are similar without competing. They both have data models, interchange syntaxes, query languages, schema languages, and so on. Being developed for different purposes and doing different things well they can peacefully coexist and complement one another.

The relationship between RDF and topic maps is less obvious, however. Structurally, they are very similar, and their semantics are very close, although the distinctions in topic maps between base names, occurrences, and associations do not exist in RDF. At first glance it may appear that they are nearly the same, but on closer inspection it turns out that their respective communities think of the technologies in very different ways, and that features such as scope and merging actually make them rather different after all. Again, the conclusion seems to be that they are good for different things, and that there is room for both.

So what should be used where? Generally, use XML for interchange and document contents, RDF for fine-grained metadata, and topic maps for making information findable and anything that is mostly about relationships.

Conclusion

So, to sum up, topic maps make information findable by giving every concept in the information its own identity and providing multiple redundant navigation paths through the information space. These paths are semantic, and all points on the way are clearly identified with names and types that tell you what they are. This means you always know where you are, which prompted Charles

Goldfarb to call topic maps "the GPS of the information universe."

Topic maps also help by making it possible to relate together information that comes from different sources through merging and published subjects. A future article will discuss this.

Tools and references

The following is a very partial list of places where you can find useful information about topic maps.

- The TAO of topic maps, the classic introduction to topic maps.
- TM4J is an open source topic map engine project in Java.
- Perl XTM is an open source topic map engine in Perl.
- tmproc is an open source topic map engine in Python.
- The Omnigator is a free (as in beer) topic map browser that can display any topic map. There's also an online demo.
- easytopicmaps.com is a wiki site about topic maps.
- topicmap.com is a useful site about topic maps.
- XTM 1.0 is currently the most important specification. It has now been incorporated in the second edition of the topic map ISO standard.
- isotopicmaps.com tells you where the topic map standards are headed next.
- TOPICMAPMAIL is a general mailing list about topic maps, suitable for newbies.
- LTM, the Linear Topic Map notation, is a text-based syntax for topic maps that is easier to read and write for humans than XTM.
- Jan Algermissen maintains a registry of publicly available topic maps.

If you want to start creating your own topic maps I recommend downloading the Omnigator and writing an XTM or LTM file (see above). You can then browse your topic map in the Omnigator immediately, without any programming or stylesheets.