Topic Map Authoring With Reusable Ontologies and Automated Knowledge Mining

Joshua **Reynolds**
W. Eliot **Kimber**

Abstract

Topic Maps and their supporting infrastructure are quickly achieving the level of maturity needed to make them useful as part of the basic information management toolkit. With increasing vendor support, standardization activities, and interest in the field of Knowledge Representation and Interchange, it is clear that Topic Maps are here to stay. Unfortunately all of this progress and interest in no way eases the formidable task of authoring Topic Maps.

Our experience indicates that XSLT works well for Topic Map generation over sets of XML resources. Markup, through it's design and implementation, frequently captures a good deal of semantic information, making it a perfect candidate for knowledge extraction. There are essentially two ways of extracting that knowledge into a Topic Map when those marked-up resources conform to a known schema (DTD, RELAX-NG, XSD, or even just an in-house convention). The first is hand authoring. This involves reading the document and using human reasoning to interpret the markup and it's content, then creating the Topic Map from this information. The second is to use the schema itself. By applying knowledge extraction techniques to the schema, we can use the same logic across an arbitrarily large set of conforming documents. As markup is easily machine processed, incorporating this reasoning in some sort of algorithmic form is clearly desirable. Going from markup (XML) to markup (XTM) makes XSLT the prime candidate for expressing this algorithm. Topic Map merging enables these generated XTMs to be combined with topical information that can't be extracted using a style-sheet. Although the former allows for more precision, the latter implies far less cost, both in terms of initial effort, as well as maintenance (only the style-sheet must be authored/maintained).

This paper provides a case study used to illustrate how to ease the task of Topic Map creation through a multi-stage modularized process. The first stage is hand authoring a relatively invariant "ontology" Topic Map. This consists of defining the ontology of types and associations that capture the data model for a particular subject domain. The assumption is that this ontology would be relatively stable over time, and a good candidate for reuse. The second is generating additional Topic Maps through an algorithmic process (XSLT) applied to XML document instances. The third is hand authoring those things not captured in the first two stages. This consists of the capture of information not directly discernible from the markup, or stored in non-XML resources. The resultant Topic Maps are merged giving a Topic Map that can be as rich as if completely hand authored. We present the source documents and code (stylesheets) used in an exploratory implementation of this approach, and lay out a more generalized approach to using this methodology. We finish by identifying possible issues with this approach, as well as enumerating alternatives, and stating the conclusions we were able to infer from our exploration.

Table of Contents

1. Introduction

One of the fundamental challenges in implementing a knowledge management solution based on Topic Maps is keeping the authorship and maintenance tasks at manageable level. The burden involved in these activities is directly dependent on the richness of your information set and the complexity of knowledge you wish to represent. If we are interested in using a Topic Map solution on an enterprise scale, we can assume such a rich and complex set of information resources. This, coupled with the additional fact that these resources may be dynamically changing, must be held in the forefront of our minds when planning our Topic Map based solution.

Hand authorship and manual maintenance of Topic Maps is certainly sufficient for a number of use cases. If a particular knowledge domain is being modeled without an eye towards occurrences in an external information base, hand authoring is appropriate and, until we have reasoning computers, automating such high level knowledge domain modeling will be difficult at best, impossible at worst[1]. If external occurrences are to be included in our Topic Map but the form in which they occur is stable then hand authorship may still be appropriate. However hand authorship quickly becomes too burdensome when dealing with a large number of information resources.

In tandem with the authorship question is the question of how we keep the Topic Maps current in the face of mutability of information resources. There are several ways that we could do this. In the case of occurrences, we could maintain a call-back listener on the resource that is addressed. Thus if the resource changes, the Topic Map author could be notified and the Topic Map updated. This is a viable solution for documents that are managed solely by the author, but is fairly labor intensive. This call-back listener could be a software artifact, or it could be a person charged with keeping things up to date. In either case the end result is that a human being must update the Topic Map manually.

Clearly it is desirable that we minimize the amount of human effort in both of these activities. A methodology for minimizing the amount of human activity would be one that would ultimately result in:

1. Less risk of Topic Map inaccuracy due to human error during authorship and maintenance.

2. Decreased amount of human time expended during a Topic Map's life cycle.

How can we maintain the power of Topic Maps, while automating as much of the authoring and maintenance process as possible?

The approach that we have taken in our exploration is an intuitive one. Allow humans to do that which they can do best, which is the very high-level ontology modeling, and leverage computers to do the rest, which is processing large quantities of information. The rules for how this information is transformed into Topic Maps are still defined by people, but only once, in the form of an algorithm, instead of repeatedly applying human reasoning. If anything is missed a third group of Topic Maps can still be defined to fill in the gaps. By using the merging capabilities of Topic Maps the end result is a system that is much more robust in the face of changing information sets, can easily deal with new documents being added in, and is far more stable than a purely hand authored solution.

Our XML-based [XML]exploration will be as follows: We will create a static ontology that captures the basic taxonomy and relationships of our business domain, then define the rules for automatically extracting the knowledge

[1]One possible approach to automation of occurrence authoring is the CyC [CYC] system. The CyC system is an artificial intelligence system whose knowledge model closely mirrors the Topic Map paradigm. The CyC knowledge base attempts to encode “common sense,“ giving the CyC engine the potential ability to automatically associate topics with occurrences. While we have done some small experiments on this idea we have not done enough to prove or disprove this possibility.

from our repository of XML documents, add any knowledge not captured in the first two steps via hand authored Topic Maps, and merge these into the final result.

For the sake of this paper, we assume that the reader has a working knowledge of the Topic Map paradigm as well as the XTM [XTM] syntax. If not it is highly recommended that you read the XTM standard.

2. Capturing Knowledge

The question of what it means to capture knowledge is a complex one on which there is still much debate. As the question touches on the realm of the philosophical it is unlikely that this debate will ever cease. For the sake of this exploration and paper, we scope ourselves to the codification and capture of units of information and the relationships amongst them. The process that we are using to do this can be broken down into three distinct phases:

1. Creating a Static Ontology.

2. Creating rules for extracting knowledge from the information resource we are interested in.

3. Hand authoring anything not covered in the first two phases.

2.1. Static Ontology

Our approach begins with identifying the static ontology of the business domain we are interested in. This is done as an activity separate from adding specific occurrences and resources into our knowledge base. This aids both in limiting the scope of human authorship and encapsulating the high-level business domain in a stand-alone document, insulating it from unnecessary change. This would also offer a means of effective knowledge sharing in the absence of a Topic Map schema. A compelling use case is two separate organizations doing the initial domain analysis, creating an agreed-upon static ontology, and using that ontology to create their knowledge model, complete with occurrences. This would enable meaningful merging of their Topic Maps.

Another compelling use for static ontologies is the modularization of more basic ontologies, allowing for quicker creation of business-specific ones. This has the additional benefit of creating a shared group of ontologies that not only codifies a particular area of knowledge expertise, but allows for consistency of modeling across different organizations and faster time to delivery for more specific domain ontologies.

What is entailed in creating the static ontology? At least one domain expert should be involved in the process, but it is essentially a mapping of the taxonomy of the business domain and the reification of the relationships between the taxonomic members. XTM is our encoding methodology of choice, owing largely to its simplicity and expressive power.

2.2. Automated Knowledge Mining

The second step in our Topic Map creation process is what we refer to as automated knowledge mining. For our exploration we have scoped ourselves to mining only information marked up in XML, but the process could be applied to a large classes of information resources.[2]

For this paper we are assuming that the XML that will be mined has a high amount of semantic markup and is authored against a semantically-modeled DTD or schema. It is far simpler to leverage the structure of semantic markup than markup which focuses only on formatting when looking to author an ontology over such resources. This phase of the study involves the analysis of the information resources in the context of the static ontology, identifying topic occurrences and relationships among topics, and transforming this identification into an actual Topic Map. Since our target storage for this is XTM, XSLT [XSLT] is immediately indicated to automate this transformation.

2.3. Customization

Any knowledge that we wish to capture that is not easily extracted must be hand coded. This is essentially filling in any blank spaces left by the static ontology and the generated Topic Maps. Since this is the most obvious time-

[2]For information and ideas on how this could be applied see [AKM]

intensive area, it is desirable to minimize this portion as much as possible. It has been our finding that the richer the information model is in intuitive semantics, the easier it is to generate and make explicit the knowledge contained within. The amount of hand coding necessary to arrive at an arbitrarily complete Topic Map is inversely proportional to the semantic richness of the markup.

3. Case Study: Case Law, Topic Maps, and XSLT

The main focus of our exploration involved a corpus of legal documents all of which reside in the domain of case law. Our only requirement was that we create a basic Topic Map over the body of law.

It was clear from our initial attempt hand coding a Topic Map over our chosen domain (even over a small set of information resources) would be far too burdensome, both in initial authorship and ongoing maintenance. Though it would be reasonable to expect that such a document base would be relatively static over time, even the initial authorship was deemed to daunting to do manually. Although Case Law itself aggregates and is not dynamically changing, this does not preclude the documents changing due to an evolving, more complete information model. An additional requirement then is the need to be able to add new conforming case law documents into the knowledge base quickly and easily. This is what drove us to look to automation as an aid in the creation and maintenance of our Topic Maps and the approach that is detailed hereafter.

3.1. Static Ontology for the Legal Domain

In defining the static ontology we took as modular an approach as possible, trying to seek out clear boundaries for reuse. This led us to a clear partitioning of the solution space. We defined a number of distinct topic sets forming a rough hierarchy of re-usable topic map “modules.” The top module set is generic across all business domains and serves to provide a common vocabulary for characterizing topic characteristics and associations. The second set deals solely with the legal domain, capturing those concepts and relationships that are common to the realm of case law.

The generic module set consists of two topic map documents: fundamental subjects and fundamental characteristics. The Fundamental Subjects map defines a small set of subjects that are universal to all topics. The Fundamental Characteristics map defines subjects that represent fundamental topic characteristic types.

The Fundamental Subjects reflect those subjects needed by the case law topic map but that are not specific to case law. They consist of the following topics:

Abstract Data Type	A subject that represents an abstract data type, as distinct from business objects or characteristic types. Is an instance of the XTM-defined subject “Class”.
Business Object	A subject that represents some sort of business object: a real-world thing (including computer constructs) involved in a business process. Is an instance of the XTM-defined subject “Class”.
Author	An entity credited with having produced a document. Is an instance of Business Object.
Plural name	A plural form of a base name.
Definition	Characterizes an occurrence that serves to define the subject of the topic. Usually used with resourceData to enable quick display of a relatively short definition of the topic. Is an instance of the XTM-defined subject “Occurrence Type”.
Description	An occurrence that serves to describe a topic in order to convey the general purpose of the topic. Is an instance of the XTM-defined subject “Occurrence Type”.
Authored By	An association between a document and the entity that authored it (or is credited with having authored it). Authored-by is an instance of Association Type.
Document	The role of "document" within an authored-by/author-of relationship. Document is an instance of Association Role.

Author	The role of "author" within an authored-by/author-of relationship. Author is an instance of Association Role.

The Fundamental Characteristics reflect the characteristic types needed by the case law topic map but that are, again, not specific to case law. They consist of the following topics:

Typing Topic, Characteristic Type	A topic whose only or primary purpose is to characterize topic characteristics by their functional, rather than semantic, role. Characteristic types are primarily used to enable specialized user interface and processor behavior. This topic is itself an instance of the XTM-defined subject “Class”, meaning that this subject represents a class in the data modeling sense.
Association Type	A topic that defines a type of association. An association should define at least two distinct roles. Is an instance of Typing Topic.
Association Role	A topic that defines a role within an association. Is an instance of Typing Topic.
Occurrence Type	A topic that defines a type of occurrence. Occurrence types are typically used to trigger specific user interface or retrieval actions or behavior. Is an instance of Typing Topic.
Name Type	A topic that defines a type of name. Name types are typically used to trigger specific user interface or retrieval actions or behavior. Is an instance of Typing Topic.

The Case Law Base topic map defines those subjects specific to case law that are invariant and that are then used to characterize instance-specific subjects generated from the actual case law. These subjects reflect a one-time analysis of case law in order to define the business objects and association types inherent in case law. The development of these topics is a data modeling activity. The result could have been expressed in any number of ways, including as a UML [UML] static data model, a traditional taxonomy, or as a topic map. For this activity it of course made the most sense to express the resulting data model as a topic map. But note that, for example, we could have created a UML model as the authoritative definition of the case law ontology and then created a topic map that used the UML types as their subject identities. In any case, the resulting topic map uses instance-of and subtype-of relationships to establish type/subtype relationships and does not define any occurrences other than definition- and description-type occurrences (which simply serve as documentation for the types the subjects represent), meaning that the topic map is informationally equivalent to the corresponding UML data model.

All of the subjects in the Case Law Base topic map are either instances of Business Object or instances of other types within Case Law Base. These subjects represent the kinds of things one will find in case law: dates, case names, case law, citation, appellant, appellee, client, attorney, judge, etc. These subjects are all then used via is-a relationships from subjects that represent instances of these things in specific cases.

The associations in Case Law Base describe the types of relationships that can exist among the case law business objects. All of the subjects that serve to define associations are instances, directly or indirectly, of Association Type or Association Role. Most of these relationships have to do with capturing the authorship of the various documents involved in a case, e.g. Legal Opinion.

3.2. Extracting Knowledge from Case Law

In the implementation of this Topic Map application, the development of the generational portion and the static ontology were actually executed in parallel. It is our feeling that the lessons we were learning in one area drove our approach in the other and that this feedback led to a fairly well designed system. Our approach for creating our final Topic Map was to use XSLT and some reasonable assumptions to generate a Topic Map instance for each case law document and use some merging code that we authored to combine each of these generated Topic Maps with our static portions. This relied on us being able to leverage as much of the semantics from the document itself as possible.

This is an excerpt from one of the case law documents that served as the building block for our knowledge base.

```
<CASELAW>
  <CASELAW_DATA>
    <STATE>Tennessee</STATE>
    <COURT_TYPE type="supreme"/>
    <YEAR year="1998"/>
    <KEYWORD>Sexual harassment</KEYWORD>
    <KEYWORD>Workers' compensation</KEYWORD>
  </CASELAW_DATA>
  <PROCESSING_INFO>
    <TAG_DATE>05/23/02</TAG_DATE>
    <ORIGIN><SUPP>TNDE</SUPP></ORIGIN>
  </PROCESSING_INFO>
  <CASE_NAME>ANDERSON v. SAVE-A-LOT, LTD.</CASE_NAME>
  <OFFICIAL_CITATION><CITE>989 S.W.2d 277</CITE> (Tenn. 1999)</OFFICIAL_CITATION>
  <PARTY_GROUP>
    <PARTY>Bernice Anderson</PARTY>,
    <PARTY_TYPE>Plaintiff</PARTY_TYPE>,
  </PARTY_GROUP>
  <V/>
  <PARTY_GROUP>
    <PARTY>Save-a-Lot, Ltd.</PARTY>, a Supervalue Company, d/b/a
    <PARTY>Save-a-Lot Foods</PARTY>, and
    <PARTY>Liberty Mutual Insurance Company</PARTY>,
    <PARTY_TYPE>Defendants-Appellants</PARTY_TYPE>.
  </PARTY_GROUP>
  <COURT>TNSC<SUB_COURT>at Jackson.</SUB_COURT></COURT>
  <DECIDED_DATE><DATE>January 25, 1999</DATE>.</DECIDED_DATE>
  <OTHER_DATE>Rehearing Denied March 1, 1999.</OTHER_DATE>
  <ATTORNEY_GROUP><ATTORNEY role="attorney">Erich M. Shultz</ATTORNEY>, Memphis,
    for Plaintiff.</ATTORNEY_GROUP>
  <ATTORNEY_GROUP><ATTORNEY role="attorney">Jack A. Childers, Jr.</ATTORNEY>,
    Bateman, Gibson &amp;amp; Childers, Memphis, for Defendants.</ATTORNEY_GROUP>
  <MAJORITY_OPINION>
    <ORGANIZATION LEVEL="99">
      <JUDGE_BOLDITAL>OPINION</JUDGE_BOLDITAL>
    </ORGANIZATION>
    <PARA INDENT="YES"><OPINION_JUDGE>DROWOTA</OPINION_JUDGE>, J.</PARA>
    <PARA PID="1">In this workers' compensation case, we consider for the first
      time whether an employee who has been sexually harassed by a supervisor in
      the course of employment may recover workers' compensation benefits from the
```

As you can see there is a good deal of semantic information that is extractable.

The following is an excerpt from the style sheet that we used to generate the topics that for an attorney.

```
<xsl:template name="gen-attorney">
  <xsl:variable name="att-name"><xsl:call-template name="normalize-name">
      <xsl:with-param name="name">
        <xsl:value-of select="./text()"/>
      </xsl:with-param>
    </xsl:call-template>
  </xsl:variable>
  <topic>
    <xsl:attribute name="id"><xsl:value-of select="$att-name"/></xsl:attribute>
    <instanceOf>
      <topicRef>
        <xsl:attribute name="xlink:href">
          <xsl:value-of select="$basedir-url"/>
          <xsl:text>/conference/topic-maps/caselaw-base.xtm#</xsl:text>
          <xsl:value-of select="@role"/>
        </xsl:attribute>
```

```
      </topicRef>
    </instanceOf>
    <occurrence>
      <resourceRef>
        <xsl:attribute name="xlink:href">
          <xsl:value-of select="$occurrence-loc-url"/>
        </xsl:attribute>
      </resourceRef>
      </occurrence>
    <baseName>
      <baseNameString><xsl:value-of select="text()"/></baseNameString>
    </baseName>
  </topic>
</xsl:template>
```

In the beginning of the template we generate what we assume to be the unique name in the scope of the documents that we are processing. We capture that there is an occurrence of the topic for this attorney in the document we are currently processing. Additional templates create associations specific to the document of case law that this attorney participates in.

In the final merged Topic Map instance, we can see that this attorney has occurrences in several documents and that this is accurately reflected.

Note

The id's here are auto-generated as part of the merge process.

```
<topic id="x1h89mfbf0-eb">
<instanceOf>
  <topicRef xlink:href="#x1h89mfbf0-3a"/>
</instanceOf>
<baseName id="x1h89mfbf0-4c">
  <baseNameString>Attorney</baseNameString>
</baseName>

...

<topic id="x1h89mfbf0-5f4">
<instanceOf>
  <topicRef xlink:href="#x1h89mfbf0-eb"/>
</instanceOf>
<baseName id="x1h89mfbf0-5f6">
  <baseNameString>Jack A. Childers, Jr.</baseNameString>
</baseName>
<occurrence id="x1h89mfbf0-5f5">
  <resourceRef
  xlink:href=
  "file:///home/joshuar/projects/tm_explore/conference/samples/989SW2D277_TNDE.xml"/>
</occurrence>
<occurrence id="x1h89mfbf0-5f6">
  <resourceRef
  xlink:href=
  "file:///home/joshuar/projects/tm_explore/conference/samples/988SW2D145_TNDE.xml"/>
</occurrence>
</topic>
```

3.3. Putting the pieces together.

Our current generation of the fully merged Topic Map goes through the following process.

1. We generate a case-law instance specific Topic Map for each of our case law documents.

2. We hand author a Topic Map that is a shell that names the fundamental types Topic Map, the case-law specific ontology Topic Map, each of our generated Topic Maps, and any additional Topic Maps that should be included in the process. The naming takes the form of mergeMap elements that identify each of the constituent Topic Maps.

3. We apply some custom code that provides the basic merging functionality that we require for our purposes.

For the merging functionality we needed we implemented a relatively simple DOM [DOM] application that resolves the mergeMap references to create a single, unnormalized (unmerged) Topic Map instance. This instance can then be processed by any Topic Map engine to do any required topic merging (for our experiments we have been using TM4J's [TM4J] merge utility).

4. Findings

4.1. Limitations

The system that we have developed does have some significant limitations. These can be subdivided into limitations that are specific to this case study and those that exist across all Topic Map applications. For our application we have identified the following limitations.

1. Scalability

   We are currently generating and assembling our final Topic Map in batch. This means that if one XML instance changes, or a new one is added, we must redo the entire process.

2. Topic Uniqueness

   We are currently being fairly cavalier in our assumptions about name uniqueness in certain elements mapping into unique Topics. For example, the current code would be insufficient to deal with two attorneys in the set of case law that have the exact same name.

Additionally, all of the case citations in our sample documents were text only without any sort of specific linking syntax. Obviously if this were present it would be trivial to add the XSLT code to take advantage of that and create a richer more compelling Topic Map. As such, these citation relationships fall to the third portion of the process, and must be hand coded and maintained.

Of course any Topic Map application that seeks to serve as a viable solution for enterprise scale must address the issue of adding and removing topic information as well as merging and unmerging Topic Maps into a Topic Map layer. These issues are outside the scope of this case study.

4.2. Conclusion

In the end we were able to automate a large portion of the creation of a Topic Map over an arbitrarily large set of case law documents. The static components of the system are relatively stable, easing the maintenance over time, and with a sufficient amount of semantic markup, the generational portion provides a sufficient amount of detail to satisfy our initial goals of creating a usable Topic Map over a set of marked up Case Law. In attempting to do this in the quickest most logical possible fashion we were inevitably led to implement and design the system this paper has discussed.

Note

All source documents and supporting code are available from the ISOGEN web site, at http://www.isogen.com/downloads/index.htm

4.3. Areas of Further Exploration

One area where we did not explore our options for optimal reuse is the hand authoring customization component. For the purposes of our study, we deemed it sufficient to identify that as the third phase in a complete process, but in a production implementation of this system this would be the an area which would require real analysis. It is our observation that, in the early stages of implementation, the requirements of the hand authoring activity could actually drive the semantic modeling of the markup, leading to an improved and more robust information model.

An additional extension of functionality would be to generate more complete addressing into the source documents. At generation time, the occurrences could add an xpointer [XPTR] that addressed the element itself.

Bibliography

[AKM] Automated Knowledge Mining. White Paper available at http://www.isogen.com/downloads/index.htm

[CYC] OpenCyc, the open source version of Cyc(r) technology. Information available at http://www.opencyc.org/

[DOM] Document Object Model (DOM)Level 2 Core Specification Version 1.0. http://www.w3.org/TR/DOM-Level-2Core/ .

[JAVA] Java programming language. An interpreted language. http://java.sun.com/

[SGML] ISO 8879:1986, Standard Generalized Markup Language (SGML). Published by International Organization for Standardization (ISO). Not available online.

[TM4J] Topic Maps for Java http://www.tm4j.org/

[UML] Unified Modeling Language (UML) http://www.omg.org/uml/

[XML] Extensible Markup Language (XML) 1.0 (Second Edition). http://www.w3.org/TR/REC-xml

[XPATH] XML Path Language (XPath) Version 1.0. http://www.w3.org/TR/xpath.

[XPTR] XPointer Framework W3C Working Draft 10 July 2002 http://www.w3.org/TR/xptr-framework.

[XSLT] XSL Transformations (XSLT) Version 1.0, http://www.w3.org/TR/xslt

[XTM] XML Topic Maps (XTM) Version 1.0, http://www.topicmaps.org/xtm/1.0/

Biography

Joshua **Reynolds**
ISOGEN International, LLC
Dallas
United States of America

Joshua is a formally trained mathematician who has been working with computers and thinking deep thoughts for many years. He has extensive experience addressing content management problems in a highly versioned/linked problem domain and dealing with the complexities that arise. When addressing any technical challenges, his goals are to surmount them using a solid extensible architecture and implement any solutions using test-driven development and whatever tools are appropriate. His tool-set includes UML, CORBA, Pattern Based Design, XML/SGML, Java, C++, and Python among others.

W. Eliot **Kimber**
ISOGEN International, LLC
Dallas

United States of America

Eliot is a long-time contributor to the XML and SGML community, with over 15 years of experience in industrial-strength generic markup applications. He is a founding member of the XML Working Group, a co-editor of the HyTime standard (ISO 10744), and co-editor of the Standard Music Description Language draft standard (ISO 10743). For the last 9 years (the last 6 with ISOGEN International) Eliot has worked as an information systems consultant specializing in developing standards-based systems for managing documents with a focus on technical documentation and publishing systems. Eliot speaks and writes frequently on XML and related subjects. Eliot is a devoted husband and dog owner who enjoys snowboarding and body boarding on those rare occasions when he finds himself at the coast or in the mountains.