Jumpstarting the Semantic Web

Mark Watson.

Version 0.3 January 14, 2005

Abstract

Who needs the Semantic Web? Difficult to say right now because the required meta-data to support the Semantic web is not in place so we do not see any “killer applications” for the Semantic Web. The first step in laying the required groundwork for the Semantic Web is the introduction of RDF to online web sites and to documents available through known URIs.

The problem is really: how do we make it easy for content producers to add RDF to content? There are good RDF authoring tools available; one of my favorites is RDFAuthor by Libby Miller and Damian Steer. For web pages, it is not too time consuming to use an RDF authoring tool to create RDF, export it to XML, and edit the exported XML into the HTML for a web page. Like most activities, once you do this a few times it gets a lot easier.

A larger problem is creating Semantic Web meta-data for general documents (word processing, technical figures, etc.).

Another promising approach is to automatically add metadata (for example, Erik Larson “A Client Side Approach to Building the Semantic Web”). As ongoing research at my company KnowledgeBooks.com, I am experimenting with explicit ontology’s for news stories and handcrafted code to extract RDF tuples from text containing news articles. While this effort is in the early research phase, the goal is to automatically generate RDF

tuples from a corpus of news articles and then to apply automated reasoning (first order and descriptive logics) for question answering, etc.

While it involves much cost and labor, I think that one practical application of the Semantic Web will be in the development of high-value ontologies that would be used as an interface for search and information retrieval (see the last section of this paper).

Semantic Web Metadata for Documents

A principle requirement for the Semantic Web is the adherence to standards. While the standardization process for web services (using XML, SOAP, WSDL, UDDI, etc.) has broad industry acceptance and participation, the problem of standardized document formats remains a challenge.

Hopefully people who own Microsoft stock will not hate me too much for this saying this, but: Microsoft's practice of frequently changing binary document formats is a major impediment to the successful development of the Semantic Web. Recent changes in Microsoft Office to use XML for document storage are unconvincing since binary data is, I believe, still used.

I offer two solutions for publishing documents to the Semantic Web: one solution attempts to live with proprietary Microsoft Office documents and the other is a proposal for extending the open source OpenOffice program suite directly support the Semantic Web.

Living with Proprietary Microsoft Documents

Individuals and organizations that desire to publish Microsoft Office documents to the Semantic Web need some way to associate RDF meta-data with proprietary formatted documents. The simplest technique that comes to mind is to simply add a file with the extension ".rdf" for every Microsoft Office document to be published with the same root file names as the original documents and stored in the same directories. Then, a proprietary application can periodically scan specific file systems for Microsoft Office/RDF file pairs and expose the RDF appropriately on the web so that RDF aware web spiders can use the RDF to enable semantic searches.

Extending OpenOffice.org to Support the Semantic Web

Ideally, all documents should be stored in XML files using available XML Schemas. Ideally, RDF markup should be part of documents, and not stored in an associated file. Since XML documents form a tree structure, we have the opportunity to think about not only placing RDF elements near the top of the tree (so they apply to the entire document) but also placing more specific RDF elements in sub trees to allow more specific description of content in chapters, sections, and perhaps even at the paragraph level.

Newer versions of OpenOffice.org support an XML output file format so I believe that it is a natural extension to extend OpenOffice.org to support adding RDF meta-data both at the top level for a document (or, for example, technical figures made with sdraw) and in sub elements.

Automated Generation of Metadata

Automatic generation of metadata is currently impossible for arbitrary content. I believe that in the short term, with hard work, automatic metadata generation can be made to work for narrow topics. This is analogous to expert system development: narrow classes of problems can be solved with a lot of manual labor building rule-based systems.

To write metadata extraction software, I believe that the following steps are required:

1. Choose a limited domain (e.g., news stories dealing with finance).
2. Either develop a tailored ontology for this domain or reuse an existing ontology if one is available.
3. Use natural language processing (NLP) and text mining techniques to extract data tuples from text using the ontology developed in step 2.
4. Develop an automated reasoning system tailored to the ontology developed in step 2 for question answering, etc.

As an example, consider an ontology for representing knowledge about financial news. This ontology should define a vocabulary for:

1. Data on a typical company.
2. One company buying another company.
3. Data contained in a typical financial report.
4. A company issuing a financial report.
5. The stock price of a company on a given data/time.
6. Data contained in a typical analyst's report.
7. Analyst's report on a company.
8. etc., etc., etc.

In artificial intelligence and knowledge management, there is usually "no free ride". Application domains must be limited and you can expect a lot of hard work to craft systems that function well in specific domains. That said, there is (perhaps) some hope that as we use new languages like OWL for specifying ontologies and knowledge, that we might also standardize not only standard ontologies for specific subjects but also develop standard NLP/text-mining libraries that are associated with standard ontologies. In this way, we can work together to eventually "cover" most areas of human knowledge with:

1. Specific ontologies
2. Extraction libraries to create data for specific ontologies in the form or RDF/OWL tuples

3. Standards for reasoning about specific ontologies

The challenge is great because it is not enough to develop small and independent systems that work independently (but not together) for specific domains of human knowledge. We need a standard architecture so that we can all use each other's work.

Using an ontology as an interface for search and information retrieval

I have been thinking about an idea mentioned in Antoniou's and van Harmelen's "Semantic Web Primer" (a nice book, BTW): use ontologies like (in an object oriented sense) **an interface** to information. Good Java style calls for coding to interfaces so underlying implementation classes can be modified, swapped for other classes implementing the same interface, etc. Consider a similar strategy for dealing with mass amounts of information that uses different vocabularies, formats, etc.:

1. create an ontology for information that is of high value to your business
2. identify sources of raw data for the topics covered in this ontology
3. write custom connectors to the raw data sources that convert vocabulary, understand the data source formats, etc.
4. search and information retrieval is through the ontology

This is an alternative of the theme in the last section of automatic metadata creation. Here I am proposing that an ontology would be created for a high value application. A series of custom data connectors would have to be written (an expensive operation) to map individual raw data sources in to the terminology of the ontology. We will look at an example:

It would be of high economic value to be able to query many diverse news sources using a single programming API. Probably the least expensive and quickest way to prototype a system like this would be to:

1. At least temporarily, forgo OWL and simply define an ontology by specifying allowed classes and properties using RDF Schema.
2. Identify a few sources for news stories on the web and for each source, write a custom connector to fetch and convert information into RDF data using the RDF Schema developed in step 1.
3. Use an off the shelf RDF repository like Sesame to store and query the RDF data store using the RQL query language (or use the Sesame extensions in the SeRQL query language)

Developing a useful system like this proposal would be a lot of work, but certainly possible. Challenges will be converting data in unstructured text to the precise representation specified in the application specific RDF Schema. For example: converting human names like "John Smith", "Mr. John Smith", etc. into a strict <title>,

<first_name>, <middle_name>, <last_name> form. Similar problems will be normalizing how place names, monetary amounts, etc. are stored.

Both software agents and human users of this ontology would always deal with data in a known format and in a standard vocabulary.