Information Retrieval and the Semantic Web

Mr.D.B.Mirajkar[1] Prof.D.G.Chougule[2] Mr.K. K. Awale[3] Mr.S. B. Sagare[4]

Abstract: *Information retrieval technology has been central to the success of the Web. For semantic web documents or annotations to have an impact, they will have to be com- patible with Web based indexing and retrieval technol- ogy. We discuss some of the underlying problems and issues central to extending information retrieval systems to handle annotations in semantic web languages. We also describe prototype systems to explore these ideas.*

1. Introduction

Information retrieval technology has been central to the success of the Web. Web based indexing and search systems such as Google and Yahoo have profoundly changed the way we access information. For the seman- tic web technologies [4][5] to have an impact, they will have to be compatible with Web search engines and in- formation retrieval technology in general. We discuss several approaches to using information retrieval systems with both semantic web documents and with text docu- ments that have semantic web annotations.

One vision of the Semantic Web is that it will be much like the Web we know today, except that docu- ments will be enriched by annotations in machine under- standable markup. These annotations will provide meta- data about the documents as well as machine interpret- able statements capturing some of the meaning of the documents' content. We describe initial experiments that demonstrate how existing IR systems can be coaxed into supporting this scenario using a technique we call *swan- gling* to encode RDF triples as word-like terms.

In an alternate vision, semantic web content will exist in separate documents that reference and describe the content of conventional web documents. Here too it may be desirable to use conventional systems such as Google to index and retrieve these documents. We discuss how the swangling technique can also be used to add asser- tions to RDF documents in a way that is compatible with many standard search engines.

A final approach to using IR engines for SWD docu- ments is to build custom indexing and retrieval engines specifically designed to work with semantic web docu- ments as opposed to conventional ones. We describe Swoogle, a prototype crawler-based search engines for RDF documents. This system allows users to retrieve indexed RDF documents based on the RDF classes and properties they use and also uses the Haircut information retrieval engine to retrieve documents using character- based n-grams.

The next section will motivate the ability to index and search for documents consisting of or annotated with semantic web content. Section Three will lay out the landscape of possible ways to adapt information retrieval systems to the Semantic Web and Section Four will de- scribe three different prototype systems we have built to explore the problem. The fifth section summarizes this work and speculates on what the future may bring.

2. Motivation

The Semantic Web has lived its infancy as a clearly de- lineated body of Web documents. That is, by and large researchers working on aspects of the Semantic Web knew where the appropriate ontologies resided and tracked them using explicit URLs. When the desired Se- mantic Web document was not at hand, one was more likely to use a telephone to find it than a search engine. This closed world assumption was natural when a hand- ful of researchers were developing DAML 0.5 ontolo- gies, but is untenable if the Semantic Web is to live up to its name. Yet simple support for search over Semantic Web documents, while valuable, represents only a small piece of the benefits that will accrue if search and infer- ence are considered together. We believe that Semantic Web inference can improve traditional text search, and that text search can be used to facilitate or augment Se- mantic Web inference. Several difficulties, listed below, stand in the way of this vision.

Current Web search techniques are not directly suited to indexing and retrieval of semantic markup. Most search engines use words or word variants as indexing terms. When a document written using some flavor of SGML is indexed, the markup is simply ignored by many search engines. Because the Semantic Web is expressed entirely as markup, it is thus invisible to them. Even when search engines detect and index embedded markup, they do not process the markup in a way that allows the markup to be used during the search, or even in a way that can distinguish between markup and other text.

Current Web search techniques cannot use semantic markup to improve text retrieval. Web search engines typically rely on simple term statistics to identify docu- ments that are most relevant to a query. One might con-

sider techniques such as thesaurus expansion or blind relevance feedback to be integration of inference into the retrieval process, but such inference is simple compared with what is possible using semantic markup. One would like the presence of semantic markup in either the query or the documents retrieved to be exploitable during search to improve that search.

Likewise, text is not useful during inference. To the extent that it is possible to automatically convert text to a semantic representation, such resulting representations can be used during inference. However, semantic interpretation is difficult at best, and unsolved in the general case. We would like a way to exploit relevant text during inference, without needing to analyze the semantics of that text.

There is no current standard for creating or manipulat- ing documents that contain both HTML text and semantic markup. There are two prime candidates for such hybrid documents. First, semantic markup might be embedded directly in an HTML page. Unfortunately, while we call approaches like RDF and OWL semantic *markup,* they are typically used not as markup but rather as stand-alone knowledge representation languages that are not directly tied to text. Furthermore, embedding RDF-based markup in HTML is non-compliant with HTML standards up to and including HTML 4.0. This issue is currently under study by a W3C task force [23].

The second way to bind HTML to semantic markup is to create a pair of documents, one containing HTML, the other containing the corresponding semantic markup. The two files are bound by placing in each a pointer to the URI of the other, either by URI naming convention, or by concurrent retrieval (*i.e.,* as part of a single transac- tion). While this method makes it difficult to associate semantic markup with specific components of the HTML page, it is possible to implement using today's standards. Whichever approach is taken to binding semantic markup to HTML, the current lack of a standard has made it dif- ficult to exploit the relationship between the two.

One of the stated objectives of the semantic web is to enhance the ability of both people and software agents to find documents, information and answers to queries on the Web. While there has been some research on infor- mation retrieval techniques applied to documents with markup [1][2][3][7][13], combining retrieval with ontol- ogy browsing [9], the role of explicit ontologies in in- formation retrieval tasks [19], and on question answering as a retrieval task [18], much of it can be seen as incre- mental extensions to familiar paradigms. Our goal is more ambitious and offers, we think, a new paradigm for information retrieval that mixes and interleaves search, retrieval and understanding.

To explore the tight integration of search and infer- ence, we propose a framework designed to meet the fol- lowing desiderata:

- The framework must support both retrieval-driven and inference-driven processing.
- Retrieval must be able to use words, semantic markup, or both as indexing terms.
- Web search must rely on today's broad coverage, text-based retrieval engines.
- Inference and retrieval should be tightly coupled; improvements in retrieval should lead to improve- ments in inference, while improvements in inference should lead to improvements in retrieval.
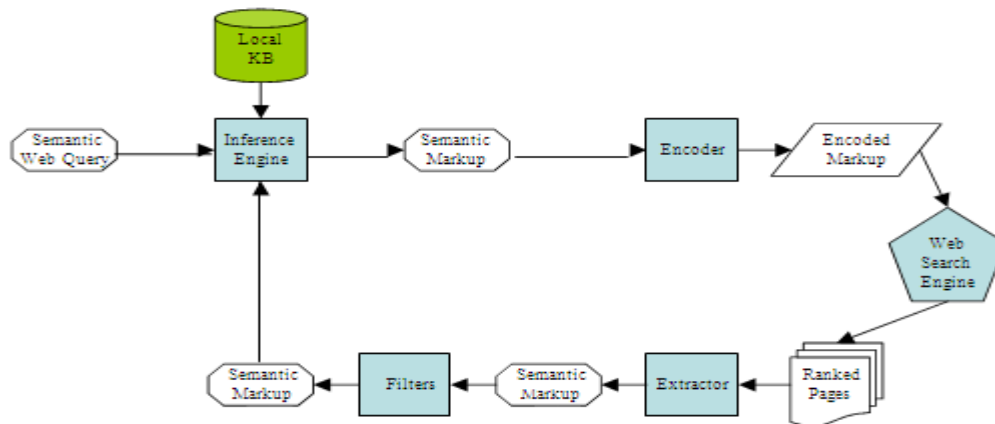

Figure 1. Integration of inference and retrieval over semantic markup. Arrows represent data flow.

In the following subsections, we first describe the por- tions of the framework that use semantic markup, then show how text processing can be mixed in to increase system capabilities and improve performance.

2.1 Processing of Semantic Markup

Imagine we are concerned only with retrieval and infer- ence over semantic markup. We would like the ability

to operate some sort of inference engine, to identify facts and rules needed by the inference engine to reach its de- sired conclusions, to search the Semantic Web for such facts and rules, and to incorporate the results of the search into the inference process. Figure 1 shows the basic architecture of such a system.

Input to the system is some sort of Semantic Web query. If the user's goal is retrieval, this might simply be semantic markup encoding the concepts being sought (*e.g.,* using XML-QL [10] or XIRQL [15]). Alternatively, if the goal is inference, the query might be a statement the system is to prove. In either case, the query is submitted to the inference engine. For retrieval, the inference engine may choose to perform limited forward chaining on the input (as a text retrieval engine might perform thesaurus expansion). For proof, the inference engine will generate a partial proof tree (or more accu- rately, one in a sequence of partial proof trees), using its local knowledge base to the extent possible. The infer- ence engine produces a description of the semantic markup to be sought on the Web.

Because we want to use a traditional Web search en- gine for the retrieval, we cannot simply use the output of the inference engine as a search query. Rather, we must first encode the semantic markup query as a text query that will be recognized by a search engine. We call this process *swangling,* for 'Semantic Web mangling.'[1] Technical details about swangling, and its application to Web pages prior to indexing, are discussed further below in Section 4. The result is a bag of words, recognizable as indexing terms by the target Web search engine(s), that characterize the desired markup.

The query is submitted to one or more Web search engines. The result will be a ranked list of Web pages, which either contain semantic markup themselves, or refer to companion pages that do. Some number of these pages must be scraped to retrieve their semantic markup. Control over how many pages to scrape, and over whether to scrape additional pages or to issue a new Web query, resides with the inference engine.

---

[1] Mangling is the technical term for a technique used in C++ and other object-oriented compilers in which the types of a method's arguments and return value are encoded in the in- ternal function name.

Only some of the semantic markup retrieved through this process will be useful for the task at hand. Some will not come from an appropriate trusted authority. Some will be redundant. Some will be irrelevant. Thus, before it is asserted into the inference engine's knowledge store, the semantic markup gleaned from each page must be filtered. The result will be a collection of facts and rules, which are likely to further the inferences being pursued, or serve as valuable relevance feedback terms. These facts and rules are passed to the inference engine, which may then iterate the entire process.

2.2 Using Text

The process described in the previous subsection makes no use of text, except to the extent that the result of markup swangling is a set of text terms. However, there is no reason that we cannot include appropriate text in the Web query. Adding text will influence the order- ing of search results, possibly biasing them toward pages that will be most useful for the task at hand. Figure 2 shows how text can be included in the framework. First, a text query can be sent directly to the search engine (augmented by swangled markup, if such is available). Second, the extractor can pull text as well as markup out of retrieved pages. As with semantic markup, extracted text may be filtered or transduced in various ways before being used. Potentially useful filters include translation, summarization, trust verification, *etc.*

Incorporation of extracted text into the query of a sub- sequent round of processing corresponds to blind rele- vance feedback. The framework therefore provides a way to include both text and semantic markup as relevance feedback terms, even when the original query is homoge- neous.

3. Prototype systems

We have explored the problems and approaches to solv- ing them through three prototype systems. While these systems do not exhaust the space of possibilities, they have challenged us to refine the techniques and provided valuable experience.

The first prototype, *OWLIR*, is an example of a system that takes ordinary text documents as input, annotates them with semantic web markup, swangles the results and indexes them in a custom information retrieval sys- tem. OWLIR can then be queried via a custom query interface that accepts free text as well as structured at- tributes.

Swangler, our second prototype, is a system that anno- tates RDF documents encoded in XML with additional RDF statements attaching swangle terms that are indexi-ble by Google and other standard Internet search engines. These documents, when available on the web, are dis- covered and indexed by search engines and can be re- trieved using queries containing text, bits of XML and swangle terms.

Our third prototype is Swoogle, a crawler-based in- dexing and retrieval system for RDF documents. It dis- covers RDF documents and adds metadata about them to its database. It also inserts them into a special version of the HAIRCUT information retrieval engine [21] that uses character n-grams as indexing terms.

3.1 OWLIR

OWLIR [23] is an implemented system for retrieval of documents that contain both free text and semantic markup in RDF, DAML+OIL or OWL. OWLIR was designed to work with almost any local information re- trieval system and has been demonstrated working with two–HAIRCUT [21] and WONDIR. In this section we briefly describe the OWLIR system; readers are referred to Shah [23] for additional details.

OWLIR defines ontologies, encoded in DAML+OIL, allowing users to specify their interests in different events. These ontologies are also used to annotate the event announcements. Figure 3 shows a portion of the OWLIR Event Ontology, which is an extension to the ontologies used in ITTalks [8]. Events may be academic or non-academic, free or paid, open or by invitation. An event announcement made within the campus is identi- fied as an instance of one of the natural kind of events or subcategories. Instances of subcategories are inferred to be a subtype of one of the natural kind of events.

Text Extraction. Event announcements are currently in free text. We prefer that these documents contain se- mantic markup. We take advantage of the AeroText™ system to extract key phrases and elements from free text documents. Document structure analysis supports exploi- tation of tables, lists, and other elements to provide more effective analysis.

We use a domain user customization tool to fine-tune extraction performance. The extracted phrases and ele- ments play a vital role in identifying event types and add- ing semantic markup. AeroText has a Java API that pro- vides access to an internal form of the extraction results. We have built DAML generation components that access this internal form, and then translate the extraction results into a corresponding RDF triple model that uses DAML+OIL syntax. This is accomplished by binding the Event ontology directly to the linguistic knowledge base used during extraction.

Inference System. OWLIR uses the metadata infor- mation added during text extraction to infer additional semantic relations. These relations are used to decide the scope of the search and to provide more relevant re- sponses. OWLIR bases its reasoning functionality on the use of DAMLJessKB [17]. DAMLJessKB facilitates reading and interpreting DAML+OIL files, and allowing
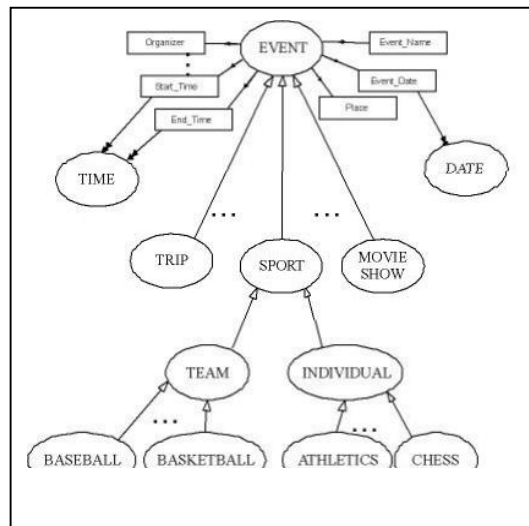

Figure 3. OWLIR annotations use terms from a DAML+OIL ontology of classes and properties that are useful in describing campus events.

the user to reason over that information. The software uses the SiRPAC RDF API to read each DAML+OIL file as a collection of RDF triples and Jess (Java Expert Sys- tem Shell) [14] as a forward chaining production system to apply rules to those triples.

DAMLJessKB provides basic facts and rules that fa- cilitate drawing inferences on relationships such as Sub-

classes and Subproperties. We enhance the existing DAMLJessKB inference capabilities by applying domain specific rules to relevant facts. For example, DAMLJessKB does not import facts from the ontology that is used to create instances; this limits its capacity to draw inferences. We have addressed this issue by importing the base Event ontology and providing relevant rules for reasoning over instances and concepts of the ontology. This combination of DAMLJessKB and domain specific rules has provided us with an effective inference engine.

As an example of the swangling process used in OWLIR, consider the markup, expressed here in RDF N3 notation, describing a movie with the title "Spiderman":

_j:00255 a owlir:movie; dc:title "Spiderman".

OWLIR has domain-specific rules that are used to add information useful in describing an event. One rule is triggered by a description of a movie event where we know the movie title. This rule requests that the Internet Movie Database (IMDB) agent seek additional attributes of this move, such as its genre. The results are added as triples, such as the following one (also in N3).

_:j00255 owlir:moviegenre "action".

This triple is then expanded with wildcards to generate seven terms, which are added to the document prior to indexing:

j00255.owlir.umbc.edu/event/moviegenre.action
.owlir.umbc.edu/event/moviegenre.action j00255..action
j00255.owlir.umbc.edu/event/moviegenre.* j00255.*.*
.owlir.umbc.edu/event/moviegenre.
**.action

We conducted experiments with OWLIR to see if se- mantic markup within documents could be exploited to improve retrieval performance. We measured precision and recall for retrieval over three different types of document: text only; text with semantic markup; and text with semantic markup that has been augmented by inference. We used two types of inference to augment docu- ment markup: reasoning over ontology instances (*e.g.,* deriving the date and location of a basketball game); and

reasoning over the ontology hierarchy (*e.g.,* a basketball game is a type of sporting event). For example, extracting the name of a movie from its description allows details about the movie to be retrieved from the Internet Movie Database site. A query looking for movies of the type *Romantic Genre* can thus be satisfied even when the ini- tial event description was not adequate for the purpose.

We generated twelve hybrid (text plus markup) que- ries, and ran them over a collection of 1540 DAML+OIL-enhanced event announcements.

Unstructured data (e.g., free text)	Structured data with in- ferred data	Structured data plus free text
25.9%	66.2%	85.5%

Table 1. Mean average precision over twelve hybrid queries given to OWLIR.

Indexed documents contain RDF Triples and RDF Triple Wildcards. This gives users the flexibility to represent queries with RDF Triple wildcards. DAML+OIL cap- tures semantic relationships between terms and hence offers a better match for queries with correlated terms.

These experiments were run using the WONDIR in- formation retrieval engine. Preliminary results are shown in Table 1 and in Shah *et al.* [23]. Retrieval times for free text documents and documents incorporating text and markup are comparable. Including semantic markup in the representation of an indexed document increases in- formation retrieval effectiveness. Additional performance benefits accrue when inference is performed over a document's semantic markup prior to indexing. While the low number of queries at our disposal limits any con- clusions we might draw about the statistical significance of these results, we are nonetheless strongly encouraged by them. They suggest that developing retrieval tech- niques that draw on semantic associations between terms will enable intelligent information services, personalized Web sites, and semantically empowered search engines.

4. Discussion

Our experience in building and evaluating these systems has helped us to understand some of the dimensions inherent in adapting information retrieval to the semantic web. We will briefly describe them as well as some of the related issues and decisions that arise.

The first dimension involves what kind of documents we expect, i.e., RDF documents encoded in XML (or perhaps N3 or some other standard encoding) or text documents with embedded RDF markup. Swoogle and Swangler are designed to work only on well formed RDF documents whereas OWLIR can handle compound documents with both text and RDF intermixed.

The second dimension concerns how the semantic web markup is processed – as structured information with an underlying data/knowledge model or as text with little or no associated model. OWLIR and Swangler treat markup as structured information and perform inferences over it following the semantics of RDF and OWL. The resulting data is ultimately reduced to swangle terms which, while a lossy transformation, still preserves much of the information. Swoogle has components on both ends of this spectrum. It stores metadata about RDF documents in its database in a way completely faithful to its structure and meaning. This allows it to retrieve documents based on the set of classes, properties and indi- viduals mentioned in them or implied by the semantic model. In this way, Swoogle treats an RDF documents as a "bag of URIs" just as a conventional IR systems treats a text document as a "bag or words". Swoogle also treats RDF documents (in their canonical XML encod- ing) as text documents which are indexed by the HAIR- CUT retrieval engine.

The final dimension delineates systems using conven- tional retrieval components and infrastructure from those that use specialized IR systems to handle semantic web documents. Swangler was designed with goal of ena- bling Google and other Internet search engines to index semantic web documents. OWLIR and Swoogle, on the other hand, use special retrieval engines adapted to han- dle the task of indexing and retrieving documents with RDF markup.

In the remainder of this section, we will introduce and discuss some additional issues that have surfaced in our work.

4.1 Tokenization

Most search engines are designed to use words as tokens. There are two immediate issues that present themselves when considering the conversion of RDF triples into swangle terms that look like indexing terms to a Web search engine – which triples should be selected for swangling and what techniques should be used to swan- gle a selected triple.

What to swangle. Some search engines, such as Google, limit query size. Care must be taken to choose a set of triples that will be effective in finding relevant documents. Some triples carry more information that others. For example, every instance is a type of owl:thing, so adding triples asserting owl:thingness will not be very helpful, especially if the query size is limited. OWL and RDF descriptions typically contain anonymous nodes (also know as "blank nodes") that represent exis- tentially asserted entities. Triples that refer to blank nodes should probably be processed in a special way, since including the "gensym" tag that represents the blank node carries no information. It might be possible to develop a statistical model for OWL annotations on documents similar to statistical language models. Such a model could help to select triples to include in a query.

How to swangle. In the OWLIR system we explored one approach to swangling triples. More experimenta- tion is clearly needed to find the most effective and effi- cient techniques for reducing a set of triples to a set of tokens that a given information retrieval system will ac- cept. The simplest approach would be to decompose each triple into its three components and to swangle these separately. This loses much of the information, of course. OWLIR followed an approach which preserved more information. Each triple was transformed into seven patterns, formed by replacing zero, one or two of its components with a special "don't care" token. Each of the seven resulting tokens was then reduced to a single word-like token for indexing

4.3 Dealing with search engines

Control. The basic cycle we've described involves (re)forming a query, retrieving documents, processing some of them, and repeating. This leaves us with a deci- sion about whether to look deeper into the ranked result set for more information to use in reforming our query, or to reform the query and generate a new result set. The choice is similar to that faced by an agent in a multiagent system that must decide whether to continue reasoning with the information it has or to ask other agents for more information or for help with the reasoning [20]. We need

some metric that estimates the expected utility of proc- essing the next document in the ranked result set.

Spiders. Web search engines typically do not process markup. So, we need a way to give a search engine spider a preprocessed (swangled) version of a Web page when it tries to spider it for indexing. This can be easily accomplished if we have control of the HTTP server that serves a page – it checks to see if the requesting agent is a spider. If so, it returns the swangled version of the page, otherwise it returns the original source page. The preprocessing can be done in advance or on demand with caching.

Offsite annotation. The technique described above depends on having control over all of the servers associated with a Semantic Web page. If this is not the case, some work arounds are needed. One option is to mirror the pages on a server that does automatic swangling. The pages should have a special annotation (*e.g.,* in RDF) that asserts the relationship between the source and mirrored pages.

Search engine limitations. Web based search engines have limitations that must be taken into account, including how they tokenize text and constraints on queries. We would like swangled terms to be accepted as indexable terms by typical search engines. The two retrieval systems we used in OWLIR were very flexible in what they accepted as a token; tokens could be of arbitrary length and could include almost any non-whitespace characters. Many commercial systems are much more constrained. With Google, for example, we were advised to keep the token length less than 50 and to include only lower and uppercase alphabetic characters. Many commercial systems also limit the size of a query to a maxi- mum number of terms. Google, for example, currently has a limit of ten terms in a query. These limitations, as well as others, affect how we have to interface to a given retrieval engine.

5. Conclusion

The Semantic Web will contain two kinds of documents. Some will be conventional text documents enriched by annotations that provide metadata as well as machine interpretable statements capturing some of the meaning of the documents' content. Information retrieval over collections of these documents offers new challenges and new opportunities. We have presented a framework for integrating search and inference in this setting that sup- ports both retrieval-driven and inference-driven process- ing, uses both text and markup as indexing terms, ex- ploits today's text-based Web search engines, and tightly binds retrieval to inference. While many challenges must be resolved to bring this vision to fruition, the benefits of pursuing it are clear. The Semantic Web is also likely to contain documents whose content is entirely encoded in an RDF based markup language such as OWL. We can use the swangling technique to enrich these documents to terms that capture some of their meaning in a form that can be indexed by conventional search engines. Finally,
there is also a role for specialized search engines that are designed to work over collections of RDF documents.

Reference

[1]   Abiteboul, S., Quass, D., McHugh, J. Widom, J. and Wiener, J. 'The Lorel query language for semistructured data.' *International Journal on Digital Libraries* 1, pages 68-88, April 1997.

[2]   Arocena, G. and Mendelzon, A. 'WebOQL: Re- structuring documents, databases and webs.' In *Interna- tional Conference on Data Engineering,* pages 24-33. IEEE Computer Society, 1998.

[3]   Bar-Yossef, Z., Kanza, Y., Kogan, Y., Nutt, W. and Sagiv, Y.. 'Quest: Querying semantically tagged docu- ments on the World Wide Web.' In *Proc. of the 4th Workshop on Next Generation Information Technologies and Systems*, volume NGITS'99, Zikhron-Yaakov (Is- rael), July 1999.

[4]   Berners-Lee, T. and Fischetti, M. Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by its Inventor. Harper, San Francisco. 1999.

[5]   Berners-Lee, T., Hendler, J. and Lassila, O. 'The Semantic Web.' *Scientific American*, May 2001.

[5a]   Brin, Sergey and Lawrence Page, The anatomy of a large-scale hypertextual Web search engine, Proceedings of the 7th international conference on World Wide Web, Elsevier Science Publishers B. V. pp 107-117, Brisbane, Australia, 1998.

[6]   Buneman, P., Khanna, S. and Tan, W-C. 'Why and Where: A Characterization of Data Provenance.' *Interna- tional Conference on Database Theory (ICDT)* 2001.

[7]   Chinenyanga, T. and Kushmerick, N. 'Elixir: An expressive and efficient language for XML   information retrieval.' In *SIGIR Workshop on XML and Information Retrieval*, 2001.

[8]   Cost, R. S., Finin, T., Joshi, A., Peng, Y., Nicholas, C., Soboroff, I., Chen, H., Kagal, L., Perich, F., Zou, Y., and Tolia, S. 'ITTALKS: A Case Study in the Semantic Web and DAML+OIL.' *IEEE Intelligent Systems*17(1):40-47, 2002.

[9]   Davies, J., Weeks, R. and Krohn, U. 'QuizRDF: Search technology for the Semantic Web.' In *WWW2002 Workshop on RDF and Semantic Web Applications,* Ha- waii, 2002.

[10]   Deutsch, A.,Fernandez, M., Florescu, D., Levy, A. and Suciu, D. 'XML-QL: A query language for XML.' In *Proceedings of the Eighth International World Wide Web Conference*, 1999.

[11]   Ding, L,. Lina Zhou, and Tim Finin, 'Trust Based Knowledge Outsourcing for Semantic Web Agents,' *2003 IEEE/WIC International Conference on Web Intel- ligence (WI 2003)*, October 2003, Halifax, Canada.

[12]   Kopena, J. and Regli, W., 'DAMLJessKB: A tool for reasoning with the Semantic Web.' *IEEE Intelligent Sys- tems* 18(3), May/June, 2003.

[13]   Kwok, C., Etzioni, O. and Weld, D. 'Scaling ques- tion answering to the Web. ' In *Proceedings of WWW10,* Hong Kong, 2001.

[14]   Mayfield, J. 'Ontologies and text retrieval.' *Knowl- edge Engineering Review* 17(1):71-75. 2002.

[15]   Mayfield, J., Finin, T., Narayanaswamy, R., Shah, C., MacCartney, W.   and Goolsbey, K. 'The Cycic Friends Network: Getting Cyc agents to reason together.' *Proceedings of the CIKM Workshop on Intelligent Information Agents*. 1995.

[16]   Mayfield, J., McNamee, P. and Piatko, C.'The JHU/APL HAIRCUT system at TREC-8.' *The Eighth Text Retrieval Conference (TREC-8)*, pages 445-452, November 1999.

[17]   Mayfield, J. and Tim Finin, Information retrieval on the Semantic Web: Integrating inference and retrieval, SIGIR Workshop on the Semantic Web, Toronto, 1 August 2004

[18]   Reagle, J. (ed.), *RDF in XHTML*. W3C Task ForceDocument, May 2003.

[19]   Shah, U., Finin, T., Joshi, A., Cost, R. S. and Mayfield, J.'Information Retrieval on the Semantic Web.' *10th Interna- tional Conference on Information and Knowledge Management*, November 2002.