Conversion of WordNet to a standard RDF/OWL representation

Mark van Assem *, **Aldo Gangemi** †, **Guus Schreiber** *

* Vrije Universiteit Amsterdam
Department of Computer Science
{mark,schreiber}@cs.vu.nl
† Laboratory for Applied Ontology
ISTC-CNR, Rome
aldo.gangemi@istc.cnr.it

Abstract

This paper presents an overview of the work in progress at the W3C to produce a standard conversion of WordNet to the RDF/OWL representation language in use in the Semantic Web community. Such a standard representation is useful to provide application developers a high-quality resource and to promote interoperability. Important requirements in this conversion process are that it should be complete and should stay close to WordNet's conceptual model. The paper explains the steps taken to produce the conversion and details design decisions such as the composition of the class hierarchy and properties, the addition of suitable OWL semantics and the chosen format of the URIs. Additional topics include a strategy to incorporate OWL and RDFS semantics in one schema such that both RDF(S) infrastructure and OWL infrastructure can interpret the information correctly, problems encountered in understanding the Prolog source files and the description of the two versions that are provided (Basic and Full) to accommodate different usages of WordNet.

1. Introduction

Recently, WordNet (Fellbaum, 1998) has been adopted in the so-called Semantic Web research community. It is used mainly for annotation and retrieval in different domains such as cultural heritage (Hollink et al., 2003), product catalogs (Guarino et al., 1999) and travel photos.[1] It is also used to ground other vocabularies such as FOAF[2], as background knowledge in ontology alignment tools and other applications[3]. Application of WordNet for such tasks on the Semantic Web requires a representation of WordNet in RDF and/or OWL (Manola and Miller, 2004; Dean et al., 2004). There are several conversions available (from Word-Net's Prolog format to RDF/OWL) which differ in design choices and scope. It is expected that the demand for Word-Net in RDF/OWL will grow in the coming years, along with the growing number of Semantic Web applications.

The WordNet Task Force[4] of the W3C's Semantic Web Best Practices Working Group[5] aims at providing a standard conversion of WordNet. There are two main motivations that support the development of a standard conversion. Firstly, development through the W3C's Working Group process results in a peer-reviewed conversion that is based on consensus of the participating experts. In this process we have also based ourselves on existing conversions. Some of these conversions are not complete and we have taken different design decisions for reasons detailed later in the paper. The resulting standard provides application developers with a resource that has the desired level of quality for most common purposes. Secondly, a standard improves interoperability between applications and data. For example, a data set consisting of digital images annotated with

WordNet in one application can be imported into other image annotation and retrieval applications that support the W3C's WordNet (provided that the applications have compatible annotation schemas).

This paper provides an overview of the work in progress at the W3C. More details and discussion can be found in the current draft of the WG Note[6] and the mailing list of Semantic Web Best Practices WG[7]. The work is based on version 2.0 of Princeton's WordNet Prolog distribution.

The remainder of this paper is organized as follows. Section 2. briefly introduces WordNet's conceptual model and Prolog format. Section 3. then describes how the conversion was done and on which requirements its design is based. Section 4. goes into more detailed issues involving the conversion, such as the addition of OWL semantics. Finally, Section 5. describes open issues that are still debated or form future work.

2. Conceptual model and Prolog format

The three core concepts in WordNet are the synset, the word sense and the word. Words are the basic lexical units, e.g. "car", while a sense is a specific sense in which a specific word is used (e.g. car as a motorcar or car as a railcar). Synsets group word senses with a synonymous meaning, such as {car, auto, automobile, machine, motorcar} or {car, railcar, railway car, railroad car}. There are four disjoint types of synset, containing exclusively nouns, verbs, adjectives or adverbs. There is one specific type of adjective, namely an adjective satellite. Furthermore, WordNet defines seventeen relations, of which ten between synsets (hyponymy, entailment, similarity, member meronymy, substance meronymy, part meronymy, classification, cause, verb grouping, attribute) and five between word senses (derivational relatedness, antonymy, see also,

---

[1] http://swordfish.rdfweb.org/discovery/2001/08/codepict/

[2] http://xmlns.com/foaf/0.1/

[3] http://esw.w3.org/mt/esw/archives/cat_applications_and_demos.html

[4] http://www.w3.org/2001/sw/BestPractices/WNET/tf

[5] http://www.w3.org/2001/sw/BestPractices/

---

[6] http://www.w3.org/2001/sw/BestPractices/WNET/wn-conversion.html

[7] http://lists.w3.org/Archives/Public/public-swbp-wg/

participle, pertains to). The remaining relations are "gloss" (between a synset and a sentence), and "frame" (between a synset and a verb construction pattern).

The Prolog distribution can be downloaded from the Word-Net website[8] and contains documentation of the source files. There are eighteen source files: one file that represents synsets, word senses and words, and then one for each of the seventeen relationships. The file with synsets contains Prolog facts such as:

```
s(100003009,1,''living\_thing'',n,1,1).
s(100003009,2,''animate\_thing'',n,1,0).
```

Each fact denotes exactly one word sense. The word senses with the same synset ID together form a synset. For example, the two facts above together form the synset with the ID 100003009. The first argument in the facts is an ID for the synset, the second gives a number to the word sense within the synset. The third argument is the lexical form (word) of the word sense and the fourth argument encodes the word sense's type (noun in this case). The fifth argument is the sense number, which gives a number to the sense in which the lexical form is used (e.g., the different senses of "car" each have a different number). The last argument is the tag count, which is the frequency of this word sense measured against a text corpus. Relations are identified by lists of facts like the following:

```
hyp(100002056,100001740).
mp(100004824,100003226).
ant(100017087,1,100019244,1).
```

The first identifies a hyponymy relation between two synsets, the second part meronymy between synsets, the third antonymy between two word senses (second and fourth argument are word numbers). Each relation either links synsets or word senses encoded as described in these examples. The documentation defines characteristics for each relationship, such as (anti-)symmetry, inverseness and value restrictions on the lexical groups (e.g. nouns, verbs) that may appear in relations. Most of these informally stated requirements can be formalized in OWL and are present in the conversion. Investigation of the Prolog source files and its documentation reveals several conflicts. For each conflict we have proposed a resolution which will be verified with Princeton. We list a few typical examples:

- the order of synset arguments of the member meronym relation is the opposite from what the documentation asserts;

- the documentation states that the hypernym relation has a reflexive relation (hyponym). The correct term is inverseness.

- the documentation states that derivational relatedness is reflexive, but here symmetry is meant.

- from the documentation it is not always clear if the symmetric relation is also present in the source file (e.g. if `der(A,B)` is in the file, is `der(B,A)` then

also present?). It is not clear if the relation also holds when only one of the symmetrical facts is present in the source.

The first three examples are situations in which we had to identify the correct OWL property characteristics for a property that represents a WordNet relation, while the documentation was erroneous or unclear. The last example is a situation in which it is unclear how we can detect erroneous entries in the source files, such that these entries can be skipped by our conversion program.

3. Conversion to RDF/OWL

This section first describes the steps taken in the design of the conversion. Then it describes some of these steps in further detail, namely the resulting schema and a comparison with existing conversions.

3.1. Process and Requirements

In the process of designing the conversion we took the following steps:

1. analysis of existing conversions;

2. formulate requirements;

3. analysis of source files and documentation;

4. design of RDF/OWL schema;

5. design of conversion program of Prolog data to RDF/OWL;

6. drafting of WG Note explaining requirements and design choices;

7. review of draft note and schema/data files by WG members and other parties.

Studying the existing conversions helped us to understand the different ways in which WordNet is used on the Semantic Web. In turn this helped to formulate and select between alternative requirements. Based on reviews the design choices and descriptions of them in the WG Note have been adapted; this process has been repeated over the course of several months.

For presentational purposes we describe the results of these steps in a different order: this Section explains the requirements that resulted, while the next Subsection explains our RDF/OWL schema. The last Subsection explains the comparison to other conversions. The other steps are not described in this paper. The requirements that have been established for the RDF/OWL version are:

1. it should be a full conversion;

2. it should be convenient to work with;

3. it should as much as possible reflect the original structure of WordNet (i.e. avoid interpretation); and

4. it should provide OWL semantics while still being intepretable by pure RDFS tools (i.e. the OWL semantics can be used but can also be ignored).

The first requirement implies that all queries that are possible on the original source are also possible on the RDF/OWL version. The second requirement means that design choices should also take into account how the representation format (in this case RDF/OWL) is used in practice and what kinds of operations are difficult to perform on it. By its nature this is a vague requirement that can only be measured against user's perceptions. The third requirement specifies that we simply want to change the representation format of WordNet without changing its conceptual model. The conversion should stay agnostic to possible interpretations of e.g. WordNet relations. For example, the hyponym relation is sometimes interpreted as similar or equal to the `rdfs:subClassOf` relation. The last requirement is based on the fact that, while OWL provides greater expressivity, RDF(S) infrastructure is more commonly used by the Semantic Web community. Schemas in OWL cannot be processed by RDFS infrastructure. In order to satisfy both types of users either two separate WordNet versions (one in RDF(S) and one in OWL) are required, or one version that can be processed by both RDF(S) and OWL tools. Note that there may be tension between the requirements. For example, while one RDF/OWL structure may reflect the WordNet structure more appropriately than another, this structure may be less convenient to work with. In such cases a suitable trade-off needs to be made between the requirements.

3.2. Classes and properties

```
Synset
    AdjectiveSynset
        AdjectiveSatelliteSynset
    AdverbSynset
    NounSynset
    VerbSynset
WordSense
    AdjectiveWordSense
        AdjectiveSatelliteWordSense
    AdverbWordSense
    NounWordSense
    VerbWordSense
Word
    Collocation
```

Figure 1: The class hierarchy of the conversion.

Defining a class hierarchy for WordNet is relatively straightforward. The basic classes are Synset, WordSense and Word, mirroring the entities defined in the conceptual model described earlier. Logical subset divisions below Synset and WordSense are based on the lexical type (adjective, adverb, noun, verb). A subset of words are collocations (e.g. mix-up and eye_contact), for which a separate class is introduced. Without this class it would be impossible to separate collocations from other Words, because most RDF query languages e.g. SPARQL (Prud'hommeaux and Seaborne, 2006) have no facilities for detecting single characters in strings. In principle the Synset subclasses are not necessary because their instances can be constructed from queries on the other classes and properties, but as they will be frequently needed they have been provided for convenience.

The properties defined for this conversion are listed in Table 1. The definition of properties given the classes is also

Property	Domain	Range	Prolog clause
synsetContainsWordSense	Synset	WordSense	s
word	WordSense	Word	s
lexicalForm	Word	xsd:string	s
synsetId	Synset	xsd:string	s
tagCount	Synset	xsd:integer	s
frame	VerbWordSense	xsd:string	fr
gloss	Synset	xsd:string	g
hyponymOf	Synset	Synset	hyp
entails	Synset	Synset	ent
similarTo	Synset	Synset	sim
memberMeronymOf	Synset	Synset	mm
substanceMeronymOf	Synset	Synset	ms
partMeronymOf	Synset	Synset	mp
classifiedByTopic	Synset	Synset	cls
classifiedByUsage	Synset	Synset	cls
classifiedByRegion	Synset	Synset	cls
causes	Synset	Synset	cs
sameVerbGroupAs	Synset	Synset	vgp
attribute	Synset	Synset	at
adjectivePertainsTo	Synset	Synset	per
adverbPertainsTo	Synset	Synset	per
derivationallyRelated	WordSense	WordSense	der
antonymOf	WordSense	WordSense	ant
seeAlso	WordSense	WordSense	sa
participleOf	WordSense	WordSense	ppl
classifiedBy	Synset	Synset	cls
meronymOf	Synset	Synset	mm,ms,mp

Table 1: Overview of properties in the WordNet schema. The "Prolog clause" column indicates the Prolog clause(s) used to generate instances of the properties.

relatively straightforward. For example, each WordNet relationship has a counterpart property in the model. Some peculiarities are explained below. The table splits the properties into four categories: properties that

- connect the main classes to each other;

- describe an attribute of these classes in the form of XML Schema Datatypes (e.g. `tagCount`)

- represent WordNet relations between Synsets (e.g. `hyponymOf`);

- represent WordNet relations between WordSenses (e.g. `antonymOf`),

- two superproperties that were introduced for relationship properties (e.g. `meronymOf`).

This last category of properties is again for querying convenience. Some WordNet relations were split into different properties. For example, the `cls` operator has three arguments: two synset IDs and one character encoding that the first synset is classified into the second synset as a topic, a specific usage or a specific region. This was mapped to three properties between synsets: `classifiedByTopic`, `classifiedByUsage` and `classifiedByRegion`.

The choice for the property `synsetContainsWordSense` represents a trade-off between requirements two and three. The original source defines a *word number* for each word sense in the synset (see Section 2.). Although it is possible to define a strict sequence in RDF using `rdf:Seq`, this is often avoided because `rdf:Seq` is awkward to process. As we did not find evidence that the word numbers are intended to signify a strict sequence, we decided to interpret this structure as a set. Sets can either be represented with the equally

awkward to process `rdf:Bag` or by a custom property that represents a one-to-many membership between Synset and WordSenses. We chose the second approach for the sake of convenience. We also do not record the word number, as this would only be useful information if the word number indeed signifies a sequence. Retaining it would require definition of a 3-aried relation between Synset, WordSense and the number, which requires a non-standard representation in RDF. We would like to avoid such structures.

The property `synsetId` is also introduced for convenience. Although the synset ID is also present in a `Synset`'s URI, retrieving it requires additional string parsing of the URI. This need can be removed by recording the synset ID separately in the `synsetId`.

3.3. Comparison to existing conversions

Our conversion is based on our own analyses as well as studying existing conversions. To the best of our knowledge there are four other conversions: by Brickley, Decker & Melnik, the University of Neuchatel and the University of Chile. Below we treat the major differences with these conversions in turn.

Firstly, we have chosen not to convert the hyponym relationship into `rdfs:subClassOf`, as done by Dan Brickley[9]. We argue that not all hyponyms can be interpreted in that way. An attempt to provide a consistent semantic translation of hyponymy has been done by (Gangemi et al., 2003), but in this work we we explicitly avoid semantic interpretation of WordNet relations (see Requirement three). Secondly, different from the conversions by Brickley and Decker & Melnik[10], our version represents all WordNet relationships. Thirdly, we represent `Words` and `WordSenses` as separate entities. The OWL conversion by the University of Neuchatel[11] only represents word senses. Fourthly, we also split some relations into sub-relations. For example, the Prolog relationship "per" denotes (a) a relation between an adjective and a noun or adjective or (b) a relation between an adverb and an adjective. We convert "per" into `adjectivePertainsTo` and `adverbPertainsTo`.

The conversion of Neuchatel is close to ours. It provides an OWL-based conversion in which relation characteristics such as (anti-)symmetry, inverseness and value restrictions are formalized. The other conversions are in RDFS only. However Neuchatel omits relations "derivation" and "classification", does not provide sub-relations and inverses for all relationships, and as already noted it does not represent Words as as separate objects.

The motivation for representing `Words` separately in our conversion is that when they do not have a URI it is impossible to refer to words directly. This may be important when one would like to annotate or reason with a word instead of a wordsense or synset. Another reason is that future integration of WordNet with other multilingual sources may require mappings between words.

The conversion of University of Chile[12] uses the same basic model (Synset, WordSense and Word) and is as complete as ours, but it does not use OWL. The development of this conversion actually ran in parallel to our own without either party being aware of it, so it served as an additional reference point to which we could validate our modeling choices instead of an inspiration.

4. Detailed design issues

This section explains the following more detailed design issues: the combination of RDFS and OWL semantics in one conversion, the two available versions (Basic and Full) and the formatting of URIs and its relation with online querying of WordNet.

4.1. RDFS and OWL semantics

An explicit requirement to the conversion is that it should define the appropriate OWL semantics. Therefore, we defined restrictions, property characteristics and disjointness statments between classes. The required information for the first two modeling decisions comes directly from the Prolog source documentation. For example, each Prolog fact in the senses file denotes exactly one word sense and also states a synset ID. Hence we can conclude that a WordSense always belongs to exactly one Synset and that a Synset always has at least one WordSense. We modeled this with a restriction on Synset (each Synset is associated to at least one WordSense) and a restriction WordSense (each WordSense belongs to exactly one Synset). Furthermore, the classes Synset, WordSense and Word are declared disjoint from each other, as are their subclasses. OWL allows the definition of property characteristics such as symmetry, inverseness and transitivity. For each relation we investigated which characteristics can be added. In most cases this is simple as the characteristic is already described in the source. Sometimes the characteristic is implicit in the documentation as is the case in the `sameVerbGroup` relationship:

> "The vgp operator specifies verb synsets that are similar in meaning and should be grouped together when displayed in response to a grouped synset search."

This implies that if `vgp(A,B)` then also `vgp(B,A)`, which makes this a symmetric relation. Sometimes background information is necessary as in the case of hyponymy. The documentation does not state that if `hyponym(A,B)` and `hyponym(B,C)` then also `hyponym(A,C)` as in: `hyponym("leading lady", "actress")` and `hyponym("actress", "actor")` then also `hyponym("leading lady", "actor")`. We used our knowledge of the semantics of the hyponym relation to derive that this relation is transitive. Furthermore, we have provided inverse properties for all WordNet relationships.

Another requirement of this conversion is that it should be interpretable by both RDF(S) and OWL-aware software. To satisfy this requirement we defined each class as both an `rdfs:Class` and an

---

[9] http://lists.w3.org/Archives/Public/www-rdf-interest/1999Dec/0002.html, http://xmlns.com/2001/08/wordnet/

[10] http://www.semanticweb.org/library/

[11] http://taurus.unine.ch/GroupHome/knowler/wordnet.html

[12] http://wordnet.princeton.edu/~agraves/

`owl:Class`, and each property both an `rdf:Property` and either an `owl:DatatypeProperty` or `owl:ObjectProperty`. This solution allows us to distribute the RDFS/OWL schema as one file. The RDF(S) software can safely ignore any information it cannot interpret (e.g. the class restrictions) if RDF(S) users take the following into account:

- for transitive properties RDFS users have to construct the transitive closure of the graph themselves or write software that deals with transitivity while querying the data;

- the WordNet data does not explicitly contain the inverse of e.g. hyponymOf. The inverse statement is only implied with the OWL statement `hyponymOf owl:inverseOf hypernymOf`. In other words, querying the hypernymOf property will return no results when using software that is not OWL-aware. Therefore, RDFS users should not use the inverse properties because they do not yield query results. Because querying for `X hypernymOf Y` is just a syntactic variant of querying for `Y hyponymOf X` RDFS users do not have less information than OWL users.

4.2. WordNet Full and WordNet Basic

The complete WordNet in RDF/OWL version described here consists of different files and is over 150 MB uncompressed RDF/XML in size. The required memory footprint when loading all files into software such as SWI-Prolog's Semantic Web library [13] may be double that amount (figures vary for different software). To mitigate memory shortage problems and/or improve query response times we have made a separate file for each WordNet relation. The required footprint can be dimished by loading only those files/relations that are required by the application at hand. WordNet can be used for a task known as sense disambiguation: the annotation of lexical forms in texts with a synset's ID (or, on the Semantic Web, its URI) to record the meaning of the lexical form (see e.g. (Ide and Véronis, 1998)). The disambiguation process consists of selecting the appropriate synset. In the sense disambiguation task (and others in which only the Synsets and their associated lexical forms are of interest) the WordSenses and Words instances add memory footprint which is not used. To keep the footprint small for such applications we provide Word-Net Basic. WordNet Basic has separate schema and instances files. The schema is a stripped-down version of the Full schema, as it does not have classes `WordSense` and `Word`, and the properties to connect them to `Synset`. It has one additional property in comparison to Full, namely `senseLabel`. The instance RDF file that belongs to WordNet Basic does not have instances of WordSense and Word, and each Synset has a set of senseLabels. The sense-Labels are filled with the content of the lexical forms that are connected to a Synset through WordSense and Word in the Full version.

The instance files for the WordNet relations can be loaded individually as for the Full version, with the caveat that only

---

[13]http://www.swi-prolog.org/packages/semweb.html

properties (relations) between Synsets are sensible to load. The relations `antonymOf`, `seeAlso`, `participleOf` and `derivationallyRelated` are between Word-Senses, and can therefore not be used in WordNet Basic.

4.3. Formatting of URIs and online querying

We have chosen to introduce identifiers for the instances of classes Synset, WordSense and Word. We use a base URI + a locally unique ID. Three kinds of entities need a URI: instances of the classes Synset, WordSense and Word. Instead of generating any unique ID we have tried to use IDs derived from information in the source and also tried to make them human-readable. Because the IDs have distinct syntactic patterns, it is easy to identify the type of the resource (Synset, WordSense or Word) by examining the URI. The patterns are described below.

Local IDs of Synset instances are composed of the synset ID, the lexical form of the first word sense in the synset and the lexical group symbol. Thus human readers can derive the lexical group of the word senses in the synset and get an idea about the kinds of words in the synset. For example:

`http://wordnet.princeton.edu/wn/107909067-bank-n/`

For WordSenses the word + its lexical group + the sense number is used. Example:

`http://wordnet.princeton.edu/wn/bank-noun-1/`

For the URI for Words we use the lexical form, which is unique within English, plus the prefix "word-". For example:

`http://wordnet.princeton.edu/wn/word-bank/`

The prefix is required to prevent clashes between the property and class names of the schema and the words. For example, the URIs for the class "Word" and the property "antonym" would be the same as the URIs for the words "word" and "antonym". Another option would be to put the schema in a different namespace than the data, but that results in additional management for users and the maintainers of the WordNet RDF/OWL version. The prefix approach avoids this drawback.

Some words contain slashes which have been converted into underscores when generating URIs. This is done to prevent the slashes to be interpreted as the character used to separate hierarchical components in URIs (Internet Engineering Task Force, 2005). For example, the URI for the word "read/write_memory" becomes:

`http://wordnet.princeton.edu/wn/read_write_memory`

There are two options in formatting the relationship between the namespace and the local part, usually termed "slash" URIs and "hash" URIs after the symbol used to connect the two parts (either / or #). The disadvantage of hash URIs is that when a HTTP GET is done (e.g. for the first example above) the browser will return the whole document

located at `http://wordnet.princeton.edu/wn`. The reason for this is that servers do not receive the fragment identifier (Jacobs and Walsh, 2004). Because WordNet is very large this is not a desirable option. The alternative is to use slash URIs. This choice implies that a decision needs to be made on which statements a server hosting WordNet should return when an HTTP GET is done for resources with a URI such as http://wordnet.princeton.edu/wn/107909067-bank-n/. Possible choices are:

- a graph that contains a pre-defined set of properties if the resource has values for them (e.g. rdf:type, rdfs:subClassOf);

- all statements connected to the resource with some offset, e.g. everything connected in at most two steps;

- the Concise Bounded Description of the URI (Stickler, 2005);

- the Symmetric Concise Bounded Description of the URI (Stickler, 2005).

The difference between the two last ones is that the Symmetric CBD not only includes statements for which the URI is the subject, but also those for which the URI is the object. We have chosen for the CBD of the URI because it "constitutes a reasonable default response to the request 'tell me about this resource'" (Stickler, 2005).

5. Open Issues

A remaining technical issue is how to support different versions of WordNet in RDF/OWL and how to define the relationship between them. Currently there is a proposal to give each WordNet version a separate namespace to prevent URI clashes between different versions. However, this does not solve the problem of how and under which circumstances to relate e.g. synsets of different versions to each other. The WordNet TF has not focused on solving this problem.

Another open issue is how to integrate WordNet with sources in other languages. Although again the TF did not focus on solving this problem as it is out of scope, we have tried to take this into account in our design, e.g. by making Words separate entities with their own URI. This allows them to be referenced directly and related to structures representing words in other RDF/OWL sources.

The actual strategy for hosting the conversion is also under discussion. For example, it is possible to redirect HTTP GETs to Princeton's server to another where the actual CBD's are computed on demand. Another strategy is to pre-compute the CBD's and store them on a Princeton server.

Furthermore, feedback from Princeton is required to check whether we have solved conflicts between source and documentation correctly. Feedback from the user community is also required to check whether this conversion provides the necessary usability.

6. References

M. Dean, A. Th. Schreiber, S. Bechofer, F. van Harmelen, J. Hendler, I. Horrocks, D. MacGuinness, P. Patel-Schneider, and L. A. Stein. 2004. OWL Web Ontology Language Reference. W3C Recommendation, World Wide Web Consortium, 10 February. Latest version: http://www.w3.org/TR/owl-ref/.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.

A. Gangemi, N. Guarino, C. Masolo, and A. Oltramari. 2003. Sweetening WORDNET with DOLCE. *AI Magazine*, 24(3):13–24.

Nicola Guarino, Claudio Masolo, and Guido Vetere. 1999. Ontoseek: Content-based access to the web. *IEEE Intelligent Systems*, 14(3):70–80, May/June.

L. Hollink, A. Th. Schreiber, J. Wielemaker, and B. J. Wielinga. 2003. Semantic annotation of image collections. In S. Handschuh, M. Koivunen, R. Dieng, and S. Staab, editors, *Knowledge Capture 2003 – Proceedings Knowledge Markup and Semantic Annotation Workshop*, pages 41–48.

Nancy Ide and Jean Véronis. 1998. Introduction to the special issue on word sense disambiguation: the state of the art. *Computational Linguistics*, 24(1):2–40, March.

Internet Engineering Task Force. 2005. Uniform Resource Identifier (URI): Generic Syntax. Request for comments, The Internet Society, January. http://www.ietf.org/rfc/rfc3986.txt.

Ian Jacobs and Norman Walsh. 2004. Architecture of the World Wide Web, Volume One. W3C Recommendation, World Wide Web Consortium, December. Latest version: http://www.w3.org/TR/webarch/.

Frank Manola and Eric Miller. 2004. RDF Primer. W3C Recommendation, World Wide Web Consortium, February. Latest version: http://www.w3.org/TR/rdf-primer/.

Eric Prud'hommeaux and Andy Seaborne. 2006. SPARQL Query Language for RDF. W3c working draft, World Wide Web Consortium, February. Latest version: http://www.w3.org/TR/rdf-sparql-query/.

Patrick Stickler. 2005. CBD - Concise Bounded Description. W3c member submission, World Wide Web Consortium, June. http://www.w3.org/Submission/2005/SUBM-CBD-20050603/.