# A Method for Converting Thesauri to RDF/OWL

Mark van Assem[1], Maarten R. Menken[1], Guus Schreiber[1], Jan Wielemaker[2], and Bob Wielinga[2]

[1] Vrije Universiteit Amsterdam, Department of Computer Science
{mark,mrmenken,schreiber}@cs.vu.nl
[2] University of Amsterdam, Social Science Informatics (SWI)
{wielemaker,wielinga}@swi.psy.uva.nl

**Abstract.** This paper describes a method for converting existing thesauri and related resources from their native format to RDF(S) and OWL. The method identifies four steps in the conversion process. In each step, decisions have to be taken with respect to the syntax or semantics of the resulting representation. Each step is supported through a number of guidelines. The method is illustrated through conversions of two large thesauri: MeSH and WordNet.

## 1 Introduction

Thesauri are controlled vocabularies of terms in a particular domain with hierarchical, associative and equivalence relations between terms. Thesauri such as NLM's Medical Subject Headings (MeSH) are mainly used for indexing and retrieval of articles in large databases (in the case of MeSH the MEDLINE/PubMed database containing over 14 million citations[1]). Other resources, such as the lexical database WordNet, have been used as background knowledge in several analysis and semantic integration tasks [2]. However, their native format, often a proprietary XML, ASCII or relational schema, is not compatible with the Semantic Web's standard format, RDF(S). This paper describes a method for converting thesauri to RDF/OWL and illustrates it with conversions of MeSH and WordNet.

The main objective of converting existing resources to the RDF data model is that these can then be used in Semantic Web applications for annotations. Thesauri provide a hierarchically structured set of terms about which a community has reached consensus. This is precisely the type of background knowledge required in Semantic Web applications. One insight from the submissions to the Semantic Web challenge at ISWC'03[2] was that these applications typically used simple thesauri instead of complex ontologies.

Although conversions of thesauri have been performed, currently no accepted methodology exists to support these efforts. This paper presents a method that

[1] http://www.ncbi.nlm.nih.gov/entrez/
[2] http://www-agki.tzi.de/swc/swc2003submissions.html

can serve as the starting point for such a methodology. The method and guidelines are based on the authors' experience in converting various thesauri. This paper is organized as follows. Section 2 provides introductory information on thesauri and their structure. In Sect. 3 we describe our method and the rationale behind its steps and guidelines. Sections 4 and 5 each discuss a case study in which the conversion method is applied to MeSH and WordNet, respectively. Additional guidelines that were developed during the case studies, or are more conveniently explained with a specific example, are introduced in these sections. Related research can be found in Sect. 6. Finally, Sect. 7 offers a discussion.

## 2 Structure of Thesauri

Many thesauri are historically based on the ISO 2788 and ANSI/NISO Z39.19 standards [3,1]. The main structuring concepts are terms and three relations between terms: Broader Term (BT), Narrower Term (NT) and Related Term (RT). *Preferred terms* should be used for indexing, while *non-preferred terms* are included for use in searching. Preferred terms (also known as *descriptors*) are related to non-preferred terms with Use For (UF); USE is the inverse of this relation. Only preferred terms are allowed to have BT, NT and RT relations. The Scope Note (SN) relation is used to provide a definition of a term (see Fig. 1).

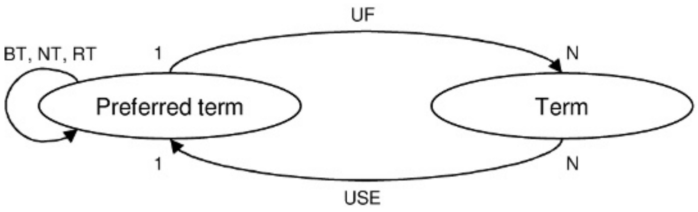

**Fig. 1.** The basic thesaurus relations. Scope note is not shown.

Two other constructs are *qualifiers* and *node labels*. Homonymous terms should be supplemented with a qualifier to distinguish them, for example "BEAMS (radiation)" and "BEAMS (structures)". A node label is a term that is not meant for indexing, but for structuring the hierarchy, for example "KNIVES By Form". Node labels are also used for organizing the hierarchy in either *fields* or *facets*. The former divides terms into areas of interest such as "injuries" and "diseases", the latter into more abstract categories such as "living" and "non-living" [3].

The standards advocate a *term-based* approach, in which terms are related directly to one another. In the *concept-based* approach [7], concepts are interrelated, while a term is only related to the concept for which it stands; i.e. a *lexicalization* of a concept [12]. The concept-based approach may have advantages such as improved clarity and easier maintenance [6].

## 3 Method Description

The method is divided into four steps: (0) a preparatory step, (1) a syntactic conversion step, (2) a semantic conversion step, and (3) a standardization step. The division of the method into four steps is an extension of previous work [15].

### 3.1 Step 0: Preparation

To perform this step (and therefore also the subsequent steps) correctly, it is essential to contact the original thesaurus authors when the documentation is unclear or ambiguous. An analysis of the thesaurus contains the following:

– Conceptual model (the model behind the thesaurus is used as background knowledge in creating a sanctioned conversion);
– Relation between conceptual and digital model;
– Relation to standards (aids in understanding the conceptual and digital model);
– Identification of multilinguality issues.

Although we recognize that multilinguality is an important and complicating factor in thesaurus conversion (see also [8]), it is not treated in this paper.

### 3.2 Step 1: Syntactic Conversion

In this step the emphasis lies on the syntactic aspects of the conversion process from the source representation to RDF(S). Typical source representations are (1) a proprietary text format, (2) a relational database and (3) an XML representation. This step can be further divided into two substeps.

**Step 1a: structure-preserving translation.** In Step 1a, a *structure-preserving* translation between the source format and RDF format is performed, meaning that the translation should reflect the source structure as closely as possible. The translation should be complete, meaning that all semantically relevant elements in the source are translated into RDF.

**Guideline 1:** Use a basic set of RDF(S) constructs for the structure-preserving translation. Only use constructs for defining classes, subclasses, properties (with domains and ranges), human-readable `rdfs:label`s for class and property names, and XML datatypes. These are the basic building blocks for defining an RDF representation of the conceptual model. The remaining RDF(S) and OWL constructs are used in Step 2 for a semantically oriented conversion. However, one might argue that the application of some constructs (e.g. domains and ranges) also belongs to semantic conversion.

**Guideline 2:** Use XML support for datatyping. Simple built-in XML Schema datatypes such as `xsd:date` and `xsd:integer` are useful to supply schemas with information on property ranges. Using user-defined XML Schema datatypes is still problematic[3]; hopefully this problem will be solved in the near future.

**Guideline 3:** Preserve original naming as much as possible. Preserving the original naming of entities results in more clear and traceable conversions. Prefix duplicate property names with the name of the source entity to make them unique. The meaning of a class or property can be explicated by adding an `rdfs:comment`, preferably containing a definition from the original documentation. If documentation is available online, `rdfs:seeAlso` or `rdfs:isDefinedBy` statements can be used to link to the original documentation and/or definition.

**Guideline 4:** Translate relations of arity three or more into structures with blank nodes. Relations of arity three or more cannot be translated directly into RDF properties. If the relation's arguments are independent of each other, a structure can be used consisting of a property (with the same name as the original relation) linking the source entity to a blank node (representing the relation), and the relation's arguments linked to the blank node with an additional property per argument (see examples in Sect. 4).

**Guideline 5:** Do not translate semantically irrelevant ordering information. Source representations often contain sequential information, e.g. ordering of a list of terms. These may be irrelevant from a semantic point of view, in which case they can be left out of the conversion.

**Guideline 6:** Avoid redundant information. Redundant information creates representations which are less clear and harder to maintain. An example on how to avoid this: if the Unique Identifier (UI) of a resource is recorded in the `rdf:ID`, then do not include a property that also records the UI.

**Guideline 7:** Avoid interpretation. Interpretations of the meaning of information in the original source (i.e., meaning that cannot be traced back to the original source or documentation) should be approached with caution, as wrong interpretations result in inconsistent and/or inaccurate conversions. The approach of this method is to postpone interpretation (see Step 2b).

Instead of developing a new schema (i.e., thesaurus metamodel), one can also use an existing thesaurus schema, such as the SKOS (see Sect. 3.4), which already defines "Concept", "broader", etc. This may be a simpler approach than to first develop a new schema and later map this onto the SKOS. However, this

[3] http://www.w3.org/2001/sw/WebOnt/webont-issues.html#I4.3-Structured-Datatypes

is only a valid approach if the metamodel of the source and of SKOS match. A drawback is that the naming of the original metamodel is lost (e.g. "BT" instead of "broader"). For thesauri with a (slightly) different metamodel, it is recommended to develop a schema from scratch, so as not to lose the original semantics, and map this schema onto SKOS in Step 3.

**Step 1b: explication of syntax.** Step 1b concerns the *explication* of information that is implicit in the source format, but intended by the conceptual model. The same set of RDF(S) constructs is used as in Step 1a. For example, the AAT thesaurus [11] uses node labels (called "Guide Terms" in AAT), but in the AAT source data these are only distinguished from normal terms by enclosing the term name in angle brackets (e.g. <KNIVES by Form>). This information can be made explicit by creating a class `GuideTerm`, which is an `rdfs:subClassOf` the class `AATTerm`, and assigning this class to all terms with angle brackets. Other examples are described in Sects. 4 and 5.

### 3.3 Step 2: Semantic Conversion

In this step the class and property definitions are augmented with additional RDFS and OWL constraints. Its two substeps are aimed at explication (Step 2a) and interpretation (Step 2b). After completion of Step 2a the thesaurus is ready for publication on the Web as an "as-is" RDF/OWL representation.

**Step 2a: explication of semantics.** This step is similar to Step 1b, but now more expressive RDFS and OWL constructs may be used. For example, a `broaderTerm` property can be defined as an `owl:TransitiveProperty` and a `relatedTerm` property as an `owl:SymmetricProperty`.

A technique that is used in this step is to define certain properties as specializations of predefined RDFS properties, e.g. `rdfs:label` and `rdfs:comment`. For example, if a property `nameOf` is clearly intended to denote a human-readable label for a resource, it makes sense to define this property as a subproperty of `rdfs:label`. RDFS-aware tools will now be able to interpret `nameOf` in the intended way.

**Step 2b: interpretations.** In Step 2b specific *interpretations* are introduced that are strictly speaking not sanctioned by the original model or documentation. A common motivation is some application-specific requirement, e.g. an application wants to treat a `broaderTerm` hierarchy as a class hierarchy. This can be stated as follows: `broaderTerm rdfs:subPropertyOf rdfs:subClassOf`. Semantic Web applications using thesauri will often want to do this, even if not all hierarchical links satisfy the subclass criteria. This introduces the notion of metamodeling. It is not surprising that the schema of a thesaurus is typically a metamodel: its instances are categories for describing some domain of interest.

**Guideline 8:** CONSIDER TREATING THE THESAURUS SCHEMA AS A METAMODEL. The instances of a thesaurus schema are often general terms or concepts, that occur as classes in other places. RDFS allows one to treat instances as a classes:

simply add the statement that the class of those instances is a subclass of `rdfs:Class`. For example, an instance $i$ is of class $C$; class $C$ is declared to be an `rdfs:subClassOf rdfs:Class`. Because instance $i$ is now also an instance of `rdfs:Class`, it can be treated as a class.

The above example of treating broader term as a subclass relation is similar in nature.

A schema which uses these constructions is outside the scope of OWL DL. Application developers will have to make their own expressivity vs. tractability trade-off here.

The output of this step should be used in applications as *a specific interpretation of the thesaurus*, not as a standard conversion.

### 3.4 Step 3: Standardization

Several proposals exist for a standard schema for thesauri.[4] Such a schema may enable the development of infrastructure that can interpret and interchange thesaurus data. Therefore, it may be useful to map a thesaurus onto a standard schema. This optional step can be made both after Step 2a (the result may be published on the web) and Step 2b (the result may only be used in an application-specific context). Unfortunately, a standard schema has not yet been agreed upon. As illustration, the schema of the W3C Semantic Web Advanced Development for Europe (SWAD-E) project[5] is mapped to MeSH in Sect. 4.

The so-called "SKOS" schema of SWAD-E is concept-based, with class `Concept` and relations `narrower`, `broader` and `related` between Concepts. A Concept can have a `prefLabel` (preferred term) and `altLabels` (non-preferred terms). Also provided is a `TopConcept` class, which can be used to arrange a hierarchy under special concepts (such as fields and facets, see Sect. 2). TopConcept is a subclass of Concept. Note that because SKOS is concept-based, it may be problematic to map term-based thesauri.

## 4 Case One: MeSH

This section describes how the method has been applied to MeSH (version 2004[6]). The main source consists of two XML files: one containing so-called *descriptors* (228 MB), and one containing *qualifiers* (449 Kb). Each has an associated DTD. A file describing additional information on descriptors was not converted. The conversion program (written in XSLT) plus links to the original source and output files of each step can be found at `http://thesauri.cs.vu.nl/`. The conversion took two persons approximately three weeks to complete.

[4] http://www.w3.org/2001/sw/Europe/reports/thes/thes_links.html

[5] http://www.w3.org/2001/sw/Europe/reports/thes/1.0/guide/

[6] http://www.nlm.nih.gov/mesh/filelist.html

### 4.1 Analysis of MeSH

The conceptual model of MeSH is centered around Descriptors, which contain Concepts [9]. In turn, Concepts consist of a set of Terms. Exactly one Concept is the preferred Concept of a Descriptor, and exactly one Term is the preferred Term of a Concept. Each Descriptor can have Qualifiers, which are used to indicate aspects of a particular Descriptor, e.g. "ABDOMEN" has the Qualifiers "pathology" and "abnormalities". Descriptors are related in a polyhierarchy, and are meant to represent broader/narrower *document retrieval sets* (i.e., not a subclass relation). Each Descriptor belongs to one (or more) of fifteen Categories, such as "Anatomy" and "Diseases" [10]. The Concepts contained within one Descriptor are also hierarchically related to each other.

This model is inconsistent with the ISO and ANSI standards, for several reasons. Firstly, the model is concept-based. Secondly, Descriptors contain a set of Concepts, while in the standards a Descriptor is simply a preferred term. Thirdly, Qualifiers are not used to disambiguate homonyms.

### 4.2 Converting MeSH

**Step 1a: structure-preserving translation.** In the XML version of MeSH, Descriptors, Concepts, Terms and Qualifiers each have a Unique Identifier (UI). Each Descriptor also has a TreeNumber (see [1]). This is used to indicate a position in a polyhierarchical structure (a Descriptor can have more than one TreeNumber), but this is implicit only. Relations between XML elements are made by referring to the UI of the relation's target (e.g. <SeeRelatedDescriptor> contains the UI of another Descriptor). In Step 1a, this is converted into instances of the property `hasRelatedDescriptor`. The explication of TreeNumbers is postponed until Step 1b.

Most decisions in Step 1a concern which XML elements should be translated into classes, and which into properties. The choice to create classes for Descriptor, Concept and Term are clear-cut: these are complex, interrelated structures. A so-called <EntryCombination> relates a Descriptor-Qualifier pair to another Descriptor-Qualifier pair. Following Guideline 4, two blank nodes are created (each representing one pair) and related to an instance of the class EntryCombination. As already mentioned, relations between elements in XML MeSH are made by referring to the UI of the target. However, each such relation also includes the *name* of the target. As this is redundant information, the name can be safely disregarded.

**Guideline 9:** Give preference to the relation-as-arc approach over the relation-as-node approach. In the relation-as-arc approach, relations are modeled as arcs between entities (RDF uses "properties" to model arcs). In the relation-as-node approach, a node represents the relation, with one arc relating the source entity to the relation node, and one arc relating the relation node to the destination entity [7]. The relation-as-arc approach is more natural to the RDF model, and also allows for definition of property semantics (symmetry, inverseness, etc.) in OWL.

It is not always possible to follow Guideline 9, e.g. in the case of MeSH <ConceptRelation>. Although the straightforward choice is to create a property that relates two Concepts, this is not possible as each ConceptRelation has an additional attribute (see Guideline 4). Again, a blank node is used (the relation-as-node approach).

**Guideline 10:** Create proxy classes for references to external resources if they are not available in RDF. Each Concept has an associated SemanticType, which originates in the UMLS Semantic Network[7]. This external resource is not available in RDF, but might be converted in the near future. In MeSH, only the UI and name of the SemanticType is recorded. One could either use a datatype property to relate the UI to a Concept (again, the redundant name is ignored), or create SemanticType instances (empty proxies for the actual types). We have opted for the latter, as this simplifies future integration with UMLS. In this scenario, either new properties can be added to the proxies, or the existing proxies can be declared `owl:sameAs` to SemanticType instances of a converted UMLS.

**Guideline 11:** Only create `rdf:ID`s based on identifiers in the original source. A practical problem in the syntactical translation is what value to assign the `rdf:ID` attribute. If the original source does not provide a unique identifier for an entity, one should translate it into blank nodes, as opposed to generating new identifiers. A related point is that if the UI is recorded using `rdf:ID`, additional properties to record an entity's UI would introduce redundancy, and therefore shouldn't be used.

**Guideline 12:** Use the simplest solution that preserves the intended semantics. In XML MeSH, only one Term linked to a Concept is the preferred term. Some terms are permutations of the term name (indicated with the attribute `isPermutedTermYN`), but unfortunately have the same UI as the Term from which they are generated. A separate instance cannot be created for this permuted term, as this would introduce a duplicate `rdf:ID`. Two obvious solutions remain: create a blank node or relate the permuted term with a datatype property `permutedTerm` to Term. In the first solution, one would also need to relate the node to its non-permuted parent, and copy all information present in the parent term to the permuted term node (thus introducing redundancy). The second solution is simpler and preserves the intended semantics.

**Step 1b: explication of syntax.** In Step 1b, three explications are made. Firstly, the TreeNumbers are used to create a hierarchy of Descriptors with a `[Descriptor] subTreeOf [Descriptor]` property. Secondly, the TreeNumber starts with a capital letter which stands for one of fifteen Categories. The

[7] http://www.nlm.nih.gov/pubs/factsheets/umlssemn.html

class `Category` and property `[Descriptor] inCategory [Category]` are introduced to relate Descriptors to their Catogories. Thirdly, the ConceptRelations are translated into three properties, `brd`, `nrw` and `rel`, thus converting from a relation-as-node to a relation-as-arc approach (see Guidelines 12 and 9). This requires two observations: (a) the values NRW, BRD and REL of the attribute `relationName` correspond to narrower, broader and related Concepts; and (b) the `relationAttribute` is not used in the actual XML, and can be removed. Without the removal of the `relationAttribute`, the arity of the relation would have prevented us from using object properties.

Some elements are not explicated, although they are clear candidates. These are XML elements which contain text, but also implicit information that can be used to link instances. For example, a Descriptor's <RelatedRegistryNumber> contains the ID of another Descriptor, but also other textual information. Splitting this information into two or more properties changes the original semantics, so we have chosen to create a datatype property for this element and copy the text as a literal value.

**Step 2a: explication of semantics.** In Step 2a, the following statements are added (a selection):

– The properties `brd` and `nrw` are each other's inverse, and are both transitive, while `rel` is symmetric;
– A Concept's `scopeNote` is an `rdfs:subPropertyOf` the property `rdfs:comment`;
– All properties describing a resource's name (e.g. `descriptorName`) are declared an `rdfs:subPropertyOf` the property `rdfs:label`;
– Each of these name properties is also an `owl:InverseFunctionalProperty`, as the names are unique in the XML file. Note that this may not hold for future versions of MeSH;
– All properties recording a date are an `owl:FunctionalProperty`;
– The XML DTD defines that some elements occur either zero or once in the data. The corresponding RDF properties can also be declared functional;
– As a Term belongs to exactly one Concept, and a Concept to exactly one Descriptor, `[Concept] hasTerm [Term]` as well as `[Descriptor] hasConcept [Concept]` is an `owl:InverseFunctionalProperty`;

Unfortunately, the relation represented by class `EntryCombination` cannot be supplied with additional semantics, e.g. that it is an `owl:SymmetricProperty` (see Guideline 9).

**Step 2b: interpretations.** In Step 2b, the following *interpretations* are made, following Guideline 8. Note that these are examples, as we have no specific application in mind.

– `brd` is an `rdfs:subPropertyOf rdfs:subClassOf`;
– `Descriptor` and `Concept` are declared `rdfs:subClassOf rdfs:Class`.

**Step 3: standardization.** In Step 3, a mapping is created between the MeSH schema and the SKOS schema. The following constructs can be mapped (using `rdfs:subPropertyOf` and `rdfs:subClassOf`):

- `mesh:subTreeOf` onto `skos:broader`;
- `mesh:Descriptor` onto `skos:Concept`;
- `mesh:hasRelatedDescriptor` onto `skos:related`;
- `mesh:descriptorName` onto `skos:prefLabel`.

There is considerable mismatch between the schemas. Descriptors are the central concepts between which hierarchical relations exist, but it is unclear how MeSH Concepts and Terms can be dealt with. SKOS defines datatype properties with which terms can be recorded as labels of Concepts, but this cannot be mapped meaningfully onto MeSH' Concept and Term classes. For example, `mesh:conceptName` cannot be mapped onto `skos:prefLabel`, as the former's domain is `mesh:Concept`, while the latter's domain is `skos:Concept` (skos:Concept is already mapped onto mesh:Descriptor). Furthermore, the `mesh:Category` cannot be mapped onto `skos:TopCategory`, because TopCategory is a subclass of `skos:Concept`, while `mesh:Category` is not a subclass of `mesh:Descriptor`.

## 5 Case Two: WordNet

This section describes how the method has been applied to WordNet release 2.0. The original source consists of 18 Prolog files (23 MB in total). The conversion programs (written in Prolog) plus links to the original source as well as the output files of each step can be found at `http://thesauri.cs.vu.nl/`. The conversion took two persons approximately three weeks to complete. Note that Step 3 for WordNet is not discussed here for reasons of space, but is available at the forementioned website.

### 5.1 Analysis of WordNet

WordNet [2] is a concept-based thesaurus for the English language. The concepts are called "synsets" which have their own identifier. Each synset is associated with a set of lexical representations, i.e. its set of synonyms. The synset concept is divided into four categories, i.e. nouns, verbs, adverbs and adjectives. Most WordNet relations are defined between synsets. Example relations are hyponymy and meronymy.

There have been a number of translations of WordNet to RDF and OWL formats. Dan Brickley[8] translated the noun/hyponym hierarchy directly into RDFS classes and subclasses. This is different from the method we propose, because it does not preserve the original source structure. Decker and Melnik[9] have created a partial RDF representation, which does preserve the original

[8] http://lists.w3.org/Archives/Public/www-rdf-interest/1999Dec/0002.html
[9] http://www.semanticweb.org/library/

structure. The conversion of the KID Group at the University of Neuchatel[10] constitutes an extension of representation both in scope and in description of semantics (by adding OWL axioms). We follow mainly this latter conversion and relate it to the steps in our conversion method. In the process we changed and extended the WordNet schema slightly (and thus also the resulting conversion).

### 5.2 Converting WordNet

**Step 1a: structure-preserving translation.** In this step the baseline classes and properties are created to map the source representation as precisely as possible to an RDF representation:

– Classes: `SynSet`, `Noun`, `Verb`, `Adverb`, `Adjective` (subclasses of `SynSet`), `AdjectiveSatellite` (subclass of `Adjective`);
– Properties: `wordForm`, `glossaryEntry`, `hyponymOf`, `entails`, `similarTo`, `memberMeronymOf`, `substanceMeronymOf`, `partMeronymOf`, `derivation`, `causedBy`, `verbGroup`, `attribute`, `antonymOf`, `seeAlso`, `participleOf`, `pertainsTo`.

Note that the original WordNet naming is not very informative (e.g. "s" represents synset). For readability, here we use the `rdfs:label`s that have been added in the RDF version. All properties except for the last four have a synset as their domain. The range of these properties is also a synset, except for `wordForm` and `glossaryEntry`. Some properties have a subclass of `SynSet` as their domain and/or range, e.g. `entails` holds between `Verbs`.

The main decision that needs to be taken in this step concerns the following two interrelated representational issues:

1. Each synset is associated with a set of synonym "words". For example, the synset `100002560` has two associated synonyms, namely `nonentity` and `nothing`. Decker and Melnik represent these labels by defining the (multi-valued) property `wordForm` with a literal value as its range (i.e. as an OWL datatype property). The Neuchatel approach is to define a word as a class in its own right (`WordObject`). The main disadvantage of this is that one needs to introduce an identifier for each `WordObject` as it does not exist in the source representation, and words are not unique (homonymy).
2. The last four properties in the list above (`antonymOf`, etc.) do not represent relations between synsets but instead between *particular words* in a synset. This also provides the rationale for the introduction of the class WordObject in the Neuchatel representation: antonymOf can now simply defined as a property between WordObjects.

We prefer to represents words as literal values, thus avoiding the identifier problem (see Guideline 11). For handling properties like `antonymOf` we defined a helper class `SynSetWord` with properties linking it to a synset and a word. For each subclass of SynSet, an equivalent subclass of `SynSetWord` is introduced (e.g. `SynSetVerb`). A sample representation of an antonym looks like this:

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

```
<SynSetWord>
  <inSynSet rdf:resource="#100017087"/>
  <relevantWord>natural_object</relevantWord>
  <antonymOf>
    <SynSetWord>
      <inSynSet rdf:resource="#100019244"/>
      <relevantWord>artifact</relevantWord>
    </SynSetWord>
  </antonymOf>
</SynSetWord>
```

In this example, the word `natural object` in synset 100017087 is an antonym of the word `artifact` in synset 100019244.

**Step 1b: explication of syntax.** The source representation of WordNet does not contain many implicit elements. The only things that need to be added here are the notions of hypernymy and holonymy (three variants). Both are only mentioned in the text and are apparently the inverse[11] of respectively the hyponym relation and the three meronym variants. Consequently, these four properties were added to the schema.

**Step 2a: explication of semantics.** In this step additional OWL axioms can be introduced to explicate the intended semantics of the WordNet classes and properties. A selection:

– `Noun`, `Verb`, `Adverb`, `and Adjective` together form disjoint and complete subclasses of `SynSet`;
– `hyponymOf` and `hypernymOf` are transitive properties;
– `hyponymOf` only holds between nouns or verbs;[12]
– `hyponymOf/hypernymOf` and the three variants of `meronymOf/holonymOf` are inverse properties;
– `verbGroup` and `antonymOf` are symmetric properties.

In addition, we defined the properties `wordForm` and `glossaryEntry` as subproperties of respectively `rdfs:label` and `rdfs:comment`.

From the WordNet documentation it is clear that these properties have this type of intended semantics. The alternative for defining these as subproperties would have been to use `rdfs:label` and `rdfs:comment` directly in the RDF representation, thus dropping the original names. This makes the traceability of the conversion less clear.

[11] The WordNet documentation uses the term "reflexive", but it is clear that inverseness is meant.

[12] In the Neuchatel representation an intermediate class `NounsAndVerbs` is introduced to express these constraints. This is not needed, as OWL supports local property restrictions which allow one, for example, to state that the value range of the `hyponymOf` property for `Noun` must be `Noun`.

**Step 2b: interpretations.** We have used WordNet heavily in semantic annotation of images (see e.g. [5]). In that application we used the WordNet hierarchy as an RDFS subclass tree by adding the following two metastatements:

– `wn:SynSet rdfs:subClassOf rdfs:Class`;
– `wn:hyponymOf rdfs:subPropertyOf rdfs:subClassOf`

Tools such as Triple20[13] will now be able to visualize the synset tree as a subclass tree. The repercussions of this type of metamodeling for RDF storage and retrieval are discussed in [14].

## 6 Related Research

Soualmia *et al.* [13] describe a migration of a specialized, French version of MeSH to an OWL DL representation. Their goal is to improve search in a database of medical resources. We mention a few of their modeling principles. Firstly, they "clean" the taxonomy by distinguishing between part-of and is-a relationships (the former type are translated into a `partOf` property, the latter into `rdfs:subClassOf`). Secondly, qualifiers are translated into properties, and their domains are restricted to the union of the descriptors on which they may be applied. The properties are hierarchically organized using `rdfs:subPropertyOf` according to the qualifier hierarchy.

Wroe *et al.* [16] describe a methodology to migrate the Gene Ontology (GO) from XML to DAML+OIL. Their goal is to "support validation, extension and multiple classification" of the GO. In each step, the converted ontology is enriched further. For example, three new part_of relations are introduced. Also, new classes are added to group part_of instances under their parent components, which enables visualization of the hierarchy. MeSH and the KEGG enzyme database are used to enrich class definitions so a new classification for Gene enzyme functions can be made by a reasoner. Additional modeling of class restrictions allowed the same reasoner to infer 17 is-a relationships that were omitted in the original source.

Goldbeck *et al.* [4] describe a conversion of the NCI (National Cancer Institute) Thesaurus from its native XML format to OWL Lite. Their goal is to "make the knowledge in the Thesaurus more useful and accessible to the public". A mapping of XML tags onto OWL classes and properties is defined, based on an analysis of the underlying conceptual model and NCI's thesaurus development process. `rdf:ID`s are created using a transformation of the original concept names (spaces removed, illegal characters substituted). This is a reasonable approach, under the assumption that names are indeed and will remain unique (see Guideline 11).

There are two main differences with our work. Firstly, the forementioned projects do not separate between "as-is" conversion and enrichment steps, as our method does. Therefore, the conversions may only be usable for the project's

[13] http://www.swi-prolog.org/packages/Triple20/

own goals. Secondly, we try to generalize over specific conversions and aim to define a more general conversion process.

In the SWAD-Europe project a schema is being developed to encode RDF thesauri. Additionally, work is in progress to produce a guideline document on converting multilingual thesauri [8]. The development of a standard schema influences our standardization step, and guidelines on converting multilingual thesauri may be incorporated to broaden the scope of our method.

## 7 Discussion

This paper has presented a method to convert existing thesauri to RDF(S) and OWL, in a manner that is sanctioned by the original documentation and format. Only in a separate step may interpretations be made for application-specific purposes.

Two additional aims of converting existing resources to the RDF model may be identified. Firstly, the quality of a thesaurus may be improved using the semantics of RDF(S), as some thesauri use relations with weak or unclear semantics, or apply them in an ambiguous way [12]. Secondly, converted thesauri can be checked using standard reasoners, identifying e.g. missing subsumption and inverse relations (e.g. BT/NT).

Recently the W3C has installed the Semantic Web Best Practices and Deployment (SWBPD) Working Group[14], which aims to provide guidelines for application developers on how to deploy Semantic Web technology. This method may serve as input for and might be extended by this Working Group.

Several issues remain as future research in developing a methodology. Firstly, translating between the source model and the RDF model is a complex task with many alternative mappings, especially for thesauri that do not conform to or define extensions of the ISO and ANSI standards. Secondly, more guidelines are required on how to convert multilingual thesauri. Thirdly, a standard thesaurus schema is required to perform step three of our method. It is clear that the current SWAD-E proposal does not completely cover complex thesauri such as MeSH. An open question is whether the proposal might be extended or that this type of thesaurus is simply outside the scope, as MeSH deviates from the ISO and ANSI standards.

**Acknowledgements.** This work was partly supported by the IST project IST-2002-507967 “HOPS” and the CHIME project, part of the NWO ToKeN2000 programme. The authors are grateful for the useful discussions with and feedback of their colleagues.

## References

1. ANSI/NISO. Guidelines for the construction, format, and management of monolingual thesauri. Ansi/niso z39.19-2003, 2003.

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

2. Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database.* Bradford Books, 1998.
3. International Organization for Standardization. Documentation - guidelines for the establishment and development of monolingual thesauri. Iso 2788-1986, 1986.
4. Jennifer Goldbeck, Gilberto Fragoso, Frank Hartel, James Hendler, Bijan Parsia, and Jim Oberthaler. The National Cancer Institute's Thesaurus and Ontology. *Journal of Web Semantics*, 1(1), Dec 2003. URL: http://www.mindswap.org/papers/WebSemantics-NCI.pdf.
5. L. Hollink, A. Th. Schreiber, J. Wielemaker, and B. J. Wielinga. 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, 2003.
6. Douglas Johnston, Stuart J. Nelson, Jacque-Lynne A. Schulman, Allan G. Savage, and Tammy P. Powell. Redefining a thesaurus: Term-centric no more. In *Proceedings of the 1998 AMIA Annual Symposium*, 1998.
7. Alistair Miles and Brian Matthews. Review of RDF thesaurus work. Deliverable 8.2, version 0.1, SWAD-Europe, 2004. URL: http://www.w3c.rl.ac.uk/SWAD/deliverables/8.2.html.
8. Alistair Miles, Brian Matthews, and Michael Wilson. RDF encoding of multilingual thesauri. Deliverable 8.3, version 0.1, SWAD-Europe, 2004. URL: http://www.w3c.rl.ac.uk/SWAD/deliverables/8.3.html.
9. U.S. National Library of Medicine. Introduction to MeSH in XML format, November 2001. URL: http://www.nlm.nih.gov/mesh/xmlmesh.html.
10. U.S. National Library of Medicine. MeSH tree structures, May 2004. URL: http://www.nlm.nih.gov/mesh/intro_trees.html.
11. T. Peterson. *Introduction to the Art and Architecture Thesaurus.* Oxford University Press, 1994. See also: http://www.getty.edu/research/tools/vocabulary/aat/.
12. Dagobert Soergel, Boris Lauser, Anita Liang, Frehiwot Fisseha, Johannes Keizer, and Stephen Katz. Reengineering thesauri for new applications: the AGROVOC example. *Journal of Digital Information*, 4(4), March 2004.
13. L.F. Soualmia, C. Goldbreich, and S.J. Darmoni. Representing the mesh in owl: Towards a semi-automatic migration. In *Proceedings of the First International Workshop on Formal Biomedical Knowledge Representation (KR-MED 2004)*, pages 81–87, Whistler, Canada, June 2004.
14. J. Wielemaker, A. Th. Schreiber, and B. J. Wielinga. Prolog-based infrastructure for rdf: performance and scalability. In D. Fensel, K. Sycara, and J. Mylopoulos, editors, *The Semantic Web - Proceedings ISWC'03, Sanibel Island, Florida*, volume 2870 of *LNCS*, pages 644–658, Berlin/Heidelberg, October 2003. Springer Verlag.
15. Bob Wielinga, Jan Wielemaker, Guus Schreiber, and Mark van Assem. Methods for porting resources to the semantic web. In C. Bussler, J. Davies, D. Fensel, and R. Studer, editors, *Proceedings of the First European Semantic Web Symposium (ESWS2004)*, number 3053 in LNCS, pages 299–311, Heraklion, Greece, May 2004. Springer-Verlag.
16. C.J. Wroe, R. Stevens, C.A. Goble, and M. Ashburner. A methodology to migrate the Gene ontology to a description logic environment using DAML+OIL. In *Proceedings of the 8th Pacific Symposium on Biocomputing (PSB 2003)*, pages 624–635, Lihue, Hawaii, USA, January 2003.