A Method to Convert Thesauri to SKOS

Mark van Assem[1], Véronique Malaisé[1], Alistair Miles[2], and Guus Schreiber[1]

[1] Vrije Universiteit Amsterdam, Department of Computer Science
{mark, vmalaise, guus}@cs.vu.nl
[2] CCLRC Rutherford Appleton Laboratory,
Business and Information Technology Department,
Oxfordshire, OX11 0QX, UK
A.J.Miles@rl.ac.uk

Abstract. Thesauri can be useful resources for indexing and retrieval on the Semantic Web, but often they are not published in RDF/OWL. To convert thesauri to RDF for use in Semantic Web applications and to ensure the quality and utility of the conversion a structured method is required. Moreover, if different thesauri are to be interoperable without complicated mappings, a standard schema for thesauri is required. This paper presents a method for conversion of thesauri to the SKOS RDF/OWL schema, which is a proposal for such a standard under development by W3Cs Semantic Web Best Practices Working Group. We apply the method to three thesauri: IPSV, GTAA and MeSH. With these case studies we evaluate our method and the applicability of SKOS for representing thesauri.

1 Introduction

Thesauri and thesauri-like resources such as MeSH [5] and the Art and Architecture Thesaurus [9] are controlled vocabularies developed by specific communities, often for the purpose of indexing (annotation) and retrieval (search) of resources (images, text documents, web pages, video, etc.). They represent a valuable means for indexing, retrieval and simple kinds of reasoning on the Semantic Web. Most of these resources are represented in databases, as XML files, or some other special-purpose data format. For deployment in Semantic Web applications an RDF/OWL representation is required. Thesauri can be converted to RDF/OWL in different ways. One conversion might define a thesaurus metamodel which represent terms as instances of a class `Term`, while another converts them into literals contained in a property `term`. This can introduce structural differences between the conversions of two thesauri which have the same semantics. Using a common framework for the RDF/OWL representation of thesauri (and thesauri-like resources) either enables, or greatly reduces the cost of (a) sharing thesauri; (b) using different thesauri in conjunction within one application; (c) development of standard software to process them (because there is no need to bridge structural differences with mappings). However, there is a significant amount of variability in features of thesauri, as exemplified by the case studies presented here. The challenge for a common metamodel such as SKOS is to capture the essential features of thesauri and provide enough extensibility to enable specific, locally-important thesaurus features to be represented.

The SKOS Core Guide [6] and the SKOS Core Vocabulary Specification [7] are currently Working Drafts for W3C Working Group Notes. They present the basic metamodel consisting of an RDF/OWL schema, an explanation of the features that the properties and classes of the schema represent. Guidelines and examples for extending SKOS Core are given by a proposed draft appendix to the SKOS Core Guide[3] and another draft proposes additional properties for representing common features in thesauri[4]. Because they are at the proposal stage they have no formal status within W3C process as yet. For the purpose of this paper we take these four documents to represent the SKOS metamodel and guidelines. Together they define (in a non-formal way) what constitutes a "correct" SKOS RDF document. SKOS models a thesaurus (and thesauri-like resources) as a set of `skos:Concept`s with preferred labels and alternative labels (synonyms) attached to them (`skos:prefLabel`, `skos:altLabel`). Instances of the Concept class represent actual thesaurus concepts can be related with `skos:broader`, `skos:narrower` and `skos:related` properties. This is a departure from the structure of many existing thesauri that are based on the influential ISO 2788 standard published in 1986, which has *terms* as the central entities instead of concepts. It defines two types of terms (preferred and non-preferred) and five relations between terms: broader, narrower, related, use and use for. Use and use for are allowed between preferred and non-preferred terms, the others only between preferred terms [2]. More recent standards such as ANSI/NISO Z39-19 acknowledge that terms are "lexical labels" representing concepts, but are still term-based format [1]. Often it is possible to convert a term-based thesaurus into a concept-based one, but sometimes information is lost (examples appear in the paper). The standards (including SKOS) allow polyhierarchies, i.e. a term/concept can have more than one `broader` term/concept.

Careful analysis of a thesaurus may still not result in an error-less, interoperable conversion to SKOS. To help ensure the quality and utility of conversions a structured method is required. This paper addresses a methodological research question: given the SKOS metamodel for thesauri, can a step-wise method be developed that assists in converting thesauri to this metamodel in a correct manner? The method should be able to guide the development of a deterministic program (i.e. does not require human intervention) that generates correct SKOS RDF for a specific thesaurus. We address the research question by first by examining existing thesaurus conversion methods in Section 2. Secondly, we develop our method by refining an applicable existing method in Section 3. Thirdly, we apply our method to three thesauri in Sections 4 through 6. Fourthly, we evaluate our method and the SKOS metamodel in Section 7.

[3] http://isegserv.itd.rl.ac.uk/cvs-public/~checkout~/skos/drafts/appextensions.html

[4] http://www.w3.org/2004/02/skos/extensions/spec/

2 Existing Thesaurus Conversion Methods

This section discusses existing methods to convert thesauri. We distinguish conversion methods for specific thesauri, method that convert thesauri to ontologies and methods that convert any thesaurus to RDF/OWL.

A first stream of research presents methods to convert one specific thesaurus from its native format to RDF/OWL, such as for MeSH [11] and the NCI thesaurus [3]. Although the steps and techniques developed for these methods are useful in thesaurus conversion, it is not clear if they can be applied to other thesauri because only features that appear in the specific thesaurus are covered. We do not consider these methods when choosing a method to base ours on.

A second stream of research presents methods with the goal to convert any thesaurus into an *ontology*, such as the work of Soergel et al. [10]. A major difference between thesauri and ontologies is that the latter feature logical is-a hierarchies, while in thesauri the hierarchical relation can represent anything from is-a to part-of. Their method has three steps: (1) define ontology metamodel; (2) define rules to convert a traditional thesaurus into the metamodel, introducing more specific kinds of relationships; and (3) manual correction. The main requirement of the method is to refine the usual thesaurus relationships into more specific kinds of relationships such as “causes”, “hasIngredient” and “growsIn”. The method does not target a specific output format, although hints are given for conversion to RDFS. It is not clear if the method would convert thesaurus concepts into `rdfs:Class`es with `rdfs:subClassOf` and other relations between them, or rather as instances of a class `Concept` as is in SKOS.

An elaborate 7-step method is defined by Hyvönen [4][5] with the goal of creating a true ontology consisting of an RDFS or OWL class hierarchy. Thesaurus concepts are converted into instances of a metaclass (a subclass of `rdfs:Class`) so that they are simultaneously instances and classes. A main requirement of the method is that conversion refines the traditional BT/NT relationships into `rdf:type`, `rdfs:subClassOf` or `partOf`. Another requirement is to rearrange the class hierarchy to better represent an ontological structure, e.g. to ensure only the real root concepts do not have a parent. Besides refining the relations it retains the original structure by also converting the BT/NT/RT relations into equivalent RDFS properties. It does not currently use SKOS.

A third stream of research presents methods to convert thesauri into RDF/OWL without creating an ontology. Earlier work by Van Assem et al. [12] describes a method to convert thesauri in four steps: (1) preparation; (2) syntactic conversion; (3) semantic conversion; and (4) standardization. In the first step, an analysis is made of the thesaurus and its digital format. This is used in step two to convert to very basic RDF, after which it is converted to more common modeling used in RDF and OWL in step three. In the last step the RDF/OWL metamodel developed for the specific thesaurus is mapped to SKOS. This method is based on two requirements: (a) preservation of the thesaurus’

[5] In Finnish, our understanding is based on correspondence with the author.

original semantics; and (b) step-wise refinement of the thesaurus' RDF/OWL metamodel.

Work by Miles et al. [8] defines a method to convert thesauri to an earlier version of SKOS in three steps: (1) generate RDF encoding; (2) error checking and validation; and (3) publishing encoding on the web. Three case studies illustrate the method. It is based on two requirements: (a) conversion of a thesaurus to the SKOS model with the goal of supporting thesaurus interoperability (b) preserve all information encoded in the thesaurus. The first step is separated into conversion of thesauri with a "non-standard structure" or "standard structure". Thesauri with "standard structure" are based on the ISO 2788 standard. Such thesauri can be converted into instances of the SKOS schema without loss of information. Thesauri with "non-standard structure" are those who have "structural features that are not described by the standard ISO 2788". The recommendation is to develop an extension of the SKOS schema using `rdfs:subClassOf` and `rdfs:subPropertyOf` to support non-standard features as this solution ensures that both method requirements are met. The method and described cases does not admit of a third category of thesauri, namely those with non-standard structure which cannot be defined as a strict specialization of the SKOS schema (this paper shows examples of these). The second step comprises error checking and validation using the W3C's RDF validator, while the third step is not discussed further.

3 Development of Conversion Method

The development of our method is based on a tentative process with the following components: (a) defining requirements on the method; (b) comparing to existing methods and choosing an applicable one; (c) developing the steps of our method; (d) applying the method; and (e) evaluating the method. This section presents the first three components. We apply the method in Sects. 4 through 6 and evaluate in the discussion. We restrict the scope of our method to monolingual thesauri and do not discuss thesaurus metadata. We also ignore some practical issues such as defining an appropriate namespace for the converted thesaurus.

3.1 Method goal and requirements

The general goal of the method is to support interoperability of thesauri encoded in RDF/OWL. The first requirement of the method is to produce conversion programs that convert the digital representations of a specific thesaurus to SKOS. The underlying assumption is that converting to SKOS provides interoperability. A sub-requirement that follows is that the resulting conversion program should produce correct SKOS RDF. The second requirement of the method is that the converted thesaurus is complete (i.e. has all information that is present in the original) as long as this does not violate the previous requirement. For this method we value the goal of interoperability higher than the requirement of being complete.

3.2 Comparison with existing methods

Here we compare the goals and requirements to those of existing methods to choose a suitable one to use as a basis for our own. The method by Soergel et al. does not have interoperability of thesauri as a goal. For each thesaurus a new metamodel is developed. Its main requirement is to produce a more refined version of the thesaurus. This is not in opposition to our requirement of completeness, but does introduce more work than necessary to achieve our main goal and may also introduce incorrect interpretations of the thesaurus' relations.

In Hyvönen's method the thesaurus is converted into a rearranged class hierarchy. It does not use a standard metamodel such as SKOS to promote interoperability and it rearranges the thesaurus' original structure. The method by Van Assem et al. also does not have interoperability of thesauri as a goal. The metamodels of different thesauri converted using this method may have structural differences. The method by Miles et al. has the same goal as ours: interoperability of thesauri in RDF/OWL. The stated requirements of using SKOS and of completeness also match. A difference is that it does not acknowledge possible conflicts between these requirements.

3.3 Developing steps of the method

The method by Miles et al. has a comparable goal and requirements and therefore we take their method as a starting point and adapt it. We focus here on working out the first step of the method, namely producing a conversion ("encoding") of the thesaurus in correct SKOS RDF. We do not adapt and discuss steps two and three.

The first step in the method by Miles et al. is split in two different processes depending on whether the thesaurus is "standard" or "non-standard". This requires an analysis of the thesaurus, so we include this as a separate activity in our method. Furthermore, the two processes only differ on whether they convert directly to instances of the SKOS schema or into extensions of the SKOS schema (defined with `rdfs:subPropertyOf` and `rdfs:subClassOf`). We decide to merge the two processes, and for each thesaurus feature in the analysis we determine whether to use a class/property from the SKOS schema or define a new subclass/subproperty.

Substep	Activity	Output
(A) thesaurus analysis	analyze digital format, analyze documentation	catalogue of data items and constraints, list of thesaurus features
(B) mapping to SKOS	define data item to SKOS schema mapping	tables mapping data items to schema items
(C) conversion program	develop algorithm	conversion program

Table 1. Substeps and activities of step 1.

We analyzed which activities need to be performed in the step, starting with its inputs and outputs. The input of the step is the thesaurus digital format, and its documentation (including interviews with experts and applications that use the thesaurus such as websites). The output of the step should be a program that transforms the data from the original digital format to SKOS RDF. In some cases the output of the step will also include an extension of the SKOS schema. There are three activities to be performed that link output to input: creating an (algorithm for the) transformation program, defining a mapping between input data items and output SKOS RDF as a basis for the algorithm, analyzing the thesaurus. We split the last activity into two parallel analyses: an analysis of the digital format and of the documentation. Both are helpful to understand which features the thesaurus has and how they are encoded. This results in the substeps and activities summarized in Table 1.

For the thesaurus analysis, we have listed the set of features that appear in common thesauri. We derived this set from studying thesaurus standards [2, 1] and the SKOS documentation listed earlier. There are three sets: one specific to term-based thesauri, one specific to concept-based thesauri and one set that is used in both. *Term-based features* are: term, compound term (combination of two or more terms), "use" relation, "use for" relation, broader term relation between preferred terms, narrower relation between preferred terms, scope note attached to preferred term (indicates scope for which term can be used in indexing), documentation attached to terms such as definitions and historical notes. *Concept-based features* are: concept, compound concept, preferred labels, non-preferred labels, broader concept relation, narrower concept relation, documentation attached to concepts such as definitions and historical notes. General features are: node labels (explained later), facets (a top-level named group of terms or concepts that is not meant for use in indexing itself). SKOS is a concept-based model. Therefore, any feature that cannot be converted into a concept-based or generic feature falls outside the scope of the SKOS schema and thus of SKOS interoperability. Although most term-based features in their most basic form can be converted into concept-based features, there are exceptions.

A sub-activity we would like to highlight here is the identification of unique identifiers in the source to generate the `rdf:ID`s of `skos:Concepts`. Some thesauri like MeSH already provide unique identifiers, but others like GTAA do not provide one. A number of options exists: (a) generate completely new identifiers which have no relation to the terms or concepts themselves; or (b) use the name of the preferred term if it is unique (replacing illegal URI characters). The first option has the disadvantage of additional management (a mapping between source terms and identifiers needs to be maintained). The second option has the disadvantage that a concept is not independent of its name. Additional programming is required to ensure that when a term changes name, the corresponding `skos:Concept`'s label is changed, instead of its URI. Currently we have not found a particular reason to prefer one option over the other.

In the next three sections we apply the method to three thesauri. We have chosen IPSV, GTAA and MeSH because they (a) are used in practice; and (b) represent progressively complex thesauri (i.e. non-standard features). The progressive complexity allows us to explore the limitations of our method and of SKOS.

4 Case Study: IPSV

The Integrated Public Sector Vocabulary (IPSV) is a thesaurus developed in the UK for indexing government documents[6]. It is modeled with the ISO2788/BS5723 standards in mind and contains 2732 preferred terms and 4230 non-preferred terms. The IPSV is a result of the merger of three thesauri. The sources and results of the conversion are available on-line[7].

Step A: analyze thesaurus. We used the XML version[8] in our analysis as it is the most complete. IPSV-XML has a DTD which provides the catalogue of data items and their constraints. IPSV-XML is a reasonably standard term-based thesaurus with preferred and non-preferred terms both called <Item>s in the XML data. Columns one and three of Table 2 list the data items and the features (for non-standard features we describe the function instead). IPSV provides unique identifiers for its terms and has a polyhierarchy.

Step B: map data items to SKOS. We have analyzed which data items correspond to which SKOS features or specializations of them (column three of Table 2). Although polyhierarchies are not allowed in ISO 2788, this is allowed in SKOS so this does not hinder a correct conversion. We were not able to find appropriate (specializations of) SKOS properties for the last four data items in the table. The two data items that indicate version information for terms cannot be made subproperties of `skos:altLabel` or `skos:prefLabel` as done for the `AToZ` attribute, because there is no place to store the version number (only literals are allowed for the label properties). A solution would be to attach two new properties to `skos:Concept` that have instances of a class `Term` as range. To these instances we can then attach a property that repeats the term name and then another property with the version number. Although this solution represents the information correctly, it introduces redundancy into the conversion (it repeats the term name with non-SKOS classes and properties). If this is not an issue this solution can be used to remain complete. However, it is a structural work-around because SKOS does not have the ability to attach information on specific `skos:prefLabel`s and `skos:altLabel`s directly.

Items that are `Obsolete` are removed from the actual thesaurus but are retained to be able to retrieve documents that were indexed with older versions of the thesaurus. The `skos:hiddenLabel` is intended to contain labels that should not be displayed to users but should be available for retrieval purposes, so we create an `ipsv:obsoleteTerm` that is a subproperty of `skos:hiddenLabel`. Shortcuts are attached to terms in the XML, but are actually meant to be able to

[6] http://www.esd.org.uk/standards/ipsv/index.html

[7] http://thesauri.cs.vu.nl/eswc06/

[8] Also available in other formats, see http://www.esd.org.uk/documents/IPSVVersionsAndFormats.pdf

Data Item	Feature/function	Property/class
<Item Id=“*A*” ConceptId=“*B*” Preferred=”True”> <Name>*X*	Preferred Term	`skos:Concept` with `rdf:ID`=A, `skos:prefLabel`=X attached to it
<Item Id=“*A*” ConceptId=“*B*” Type=”Synonym”> <Name>*X*	Non-Preferred Term	`skos:prefLabel`=X attached to concept with rdf:ID=B
<Item Type=“misspelling”> <Name>*X*	common misspelling of a (non)preferred term	`skos:hiddenLabel`=X
<UseItem>	USE relation	none required
<ScopeNote>*X*	ScopeNote	`skos:scopeNote`=X attached to concept created for surrounding <Item>
<BroaderItem Id=“*X*”>	Broader Term	`skos:broader` to Concept with rdf:ID=X
<RelatedItem ConceptId=“*X*”>Y	Related Term	`skos:related` to Concept with rdf:ID=X
<BroaderItem Id=“*X*” Default=“true”>	default broader term	`ipsv:broaderDefault` (subproperty of `skos:broader`) to Concept with `rdf:ID`=X
<Item AToZ=“true” Preferred=“*Y*”>Z	term should be displayed on websites	`ipsv:displayableAltLabel`=Z (subproperty of `skos:altLabel`) when Y=false, `ipsv:displayablePrefLabel`=Z (subproperty of `skos:prefLabel`) when Y=true
<Item Obsolete=“true”>	obsolete term	`ipsv:obsoleteTerm`=X (subproperty of `skos:hiddenLabel`
<Item AddedInVersion=“*X*”>	X is a real indicating in which IPSV version the term was added	
<Item LastUpdatedInVersion=“*X*”>	X is a real indicating in which IPSV version the term was last changed	
<Shortcut>*X*	X is a letter; keyboard shortcut for an application	`ipsv:shortcut` attached to concept created for surrounding <Item>

Table 2. Mapping of IPSV Data Items to features and RDFS property/classes. The upper part lists standard features, the middle part specializations and the lower part non-standard features. Omitted closing tags in Data Item column.

insert a whole concept within an application, so it is attached to `skos:Concept` as a non-standard feature without a SKOS superproperty.

Step C: create conversion program. We created a SWI-Prolog program that parses the IPSV-XML file and converts it to SKOS RDF using the mappings from step 1b. The program takes an <Item> and applies the matching mappings between data items and SKOS RDF. There is no need for any other information external to the <Item> to generate the triples for that Item. For example, because non-preferred Items also contain the identifier of their preferred Item (in the ConceptId attribute), we can generate the `skos:altLabel` triple even if the preferred Item that is used to generate the `skos:Concept` is not yet processed.

Case study summary. The case study took one analyst approximately two weeks to perform and was not very complex as the thesaurus is not complicated and is clearly documented. For a few issues we contacted one of the original developers. We learned that it is not always possible to perform a complete information-preserving conversion as some information on terms was lost.

5 Case Study: GTAA

The GTAA thesaurus is the controlled vocabulary used at The Netherlands Institute for Sound and Vision[9], which archives and indexes most of the public broadcasted TV and radio programs of the Netherlands [10]. GTAA stands for *the Common Thesaurus for Audiovisual Archives*; it is the result of the collaborative work of different institutions concerned with audiovisual documents indexing, including the FilmMuseum of Amsterdam. It contains 159 831 preferred terms, 1900 non-preferred terms, and 88 categories. A sample of the source file, the conversion program and the resulting RDF are available on-line[11].

Step A: analyze thesaurus. We had access to GTAA documentation and data as text files with an ISO-style formatting. This thesaurus is a faceted term-based thesaurus, where only one facet (the Subject facet, used to describe the content of a program) is organized with the ISO 2788 broader term/narrower term hierarchy. The other facets are alphabetical controlled lists, with some scope notes (lists of people's names, geographical location, etc.). The Subject facet contains one non-standard feature called Category. Each term is supplied with at least one Category, providing an alternative way to the normal NT/BT hierarchy for indexers to find them. We list GTAA data items in column one of the upper part of Table 3 and the features they represent in column two.

Step B: map data items to SKOS. Two issues arose in this step. The first one concerns the GTAA BT relationship. In the documentation of the thesaurus, the BT and NT relationships are stated to be each other's inverse. In the data itself, two or more preferred terms can have a NT link with the same narrower term. However, this narrower term has only one BT link to one of the broader terms (instead of multiple BT links). There are two options: either the missing BT links are intended but omitted in the data, or the BT link has a special status, e.g. it is a `defaultBroader` such as in IPSV. After discussion with GTAA experts, and according to the fact that this defaultBroader relationship does not appear in the documentation, we mapped the GTAA BT to `skos:broader` (see column three of Table 3).

[9] http://www.beeldengeluid.nl/index.jsp

[10] Of the estimated 850,000 hours of audio-visual material that is preserved in the Netherlands, around 700,000 hours is archived by Sound and Vision.

[11] http://thesauri.cs.vu.nl/eswc06/

Data Item	Feature/function	Property/class
Term A	Preferred Term	`skos:Concept` with `rdf:ID`=A, `skos:prefLabel`=A attached to it
US Term B	Non-Preferred Term	`skos:altLabel`=B attached to concept
CC Category C	Grouping of Preferred Terms by Categories	`skos:member` between a `skos:Collection` (with rdf:ID=C) and a `skos:Concept`
BT Term A	Broader Term	`skos:broader`
NT Term A	Narrower Term	`skos:narrower`
RT Term A *or* See also	Related Term	`skos:related`
SN X *or* (X)	ScopeNote	`skos:scopeNote`=X attached to concept created for surrounding Preferred Term
LT	relationship between terms from different facets	`gtaa:hasLinkedTerm` (subproperty of `skos:related`)
DL	relationship between terms within a certain time period	`gtaa:hasDebateLine` (subproperty of `skos:related`)

Table 3. Mapping of GTAA Data Items to features and RDFS property/classes. Upper part lists standard features, the lower part specializations. "Term A" is an actual term in the thesaurus such as "Boat"

Secondly, there are two ways to interpret the CC relationship. Either it is meant to disambiguate different aspects of a term (as in "Chruch-institution" *vs* "Church-building"), or it is a way of grouping terms sharing a specific aspect (as with "Milk_by_animal" and "Cow-milk", "Buffalo-milk", etc.). In the second case, "Milk_by_animal" is called a node label: it is a way of grouping terms, but it should not be used for indexing. These node labels are ususally part of the term hierarchy. The experts indicated that this option was the intended usage of Categories: to provide a grouping of terms under a label that is not used in the indexing process. Nevertheless, they are meant to provide an *alternative* grouping of the GTAA terms, and thus are not part of the BT/NT hierarchy. Although we mapped the Categories to an existing SKOS construct, namely the `skos:Collection` (see column three of Table 3), this modelling remains a non-standard feature that cannot be processed by SKOS software. The Categories have explicit identifiers, from which we could infer their hierarchy (01 stands for Philosophy, and 01.01 is one of its subdivisions, for instance).

GTAA does not include identifiers for its terms, so we used the preferred term's name as the `rdf:ID` of concepts.

Step C: create conversion program. As our source for the GTAA data was plain text, we created a Perl program to convert it according to the mappings in Table 3. We also had to make some manual corrections for reference errors introduced by thesaurus maintenance. Some relationships were referring to terms of the thesaurus that became obsolete, to terms which changed spelling, or to terms that became non-preferred terms. We corrected the references, or suppressed the relationships when no reference could be found; as these are relatively straightforward decisions no expert involvement was necessary.

Case study summary. The conversion could be made by direct mapping to or by extension of the SKOS schema, except for the Categories. In the conversion process, understanding the GTAA model from textual resources and experts interview, and converting the Categories into a SKOS construct took the longest time. Including programming, the process took about two weeks for one person and a half full time.

6 Case Study: MeSH

The Medical Subject Headings (MeSH) is a large thesaurus-like vocabulary developed by the U.S. National Library of Medicine and used to index millions of biomedical article citations[12]. It contains 22,997 "descriptors", most of which are used to index the subject of articles (two of the trees do not contain subjects but publication types and geographical regions). MeSH is the result of a merger of many different sources. The input data files and results of the conversion are available on-line[13].

Step A: analyze thesaurus. MeSH is available in different formats which contain the same information. We chose the XML version[14] because it is easier to analyze and convert. MeSH-XML has a DTD which provides us with the data catalogue and constraints. MeSH is a concept-based thesaurus without facets. Concepts are called "Descriptors" in MeSH terminology. The MeSH structure is complicated: "Descriptors" contain "Concepts", "Concepts"contain "Terms". Each has a name and a unique identifier, and to each entity documentation is attached such as its date of introduction and historical notes. Descriptors are hierarchically related: each MeSH Descriptor has one or more "TreeNumbers", which implicitly encode its position in a polyhierarchy (e.g. `A01.456` is a child of `A01`). Each Descriptor has a preferred Concept, and each Concept has a preferred Term. MeSH Concepts that appear within one Descriptor can be related to each other with relations "brd", "nrw" and "rel". MeSH has fifteen trees with top-concepts named e.g. "organisms" or "diseases". These appear to be facets, but they are used in idexing articles so we interpret them as normal thesaurus Concepts.

As the MeSH DTD defines almost 90 tags[15] and for each tag different attributes, we only list the exemplary and special data items in column one of Table 4 (the corresponding feature, or function if it is a non-standard feature, is in column two). MeSH Descriptors have a redundant <DescriptorName> and <ConceptName> as these strings are the same as the name of the preferred Concept and Term, respectively.

MeSH has two non-standard features that require special attention. Firstly, so-called Qualifiers are used to indicate specific aspects of Descriptors, such

[12] http://www.ncbi.nlm.nih.gov/entrez/

[13] http://thesauri.cs.vu.nl/eswc06/

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

[15] An overview of their meaning is given in: http://www.nlm.nih.gov/mesh/xml_data_elements.html

Data Item	Feature/function	Property/class
<DescriptorRecord> <DescriptorName> <String>*X* <DescriptorUI>*Y*	Concept	`skos:Concept` with `rdf:ID`=Y and `skos:prefLabel`=X
<Concept PreferredConceptYN=”Y”> <ScopeNote>*X*	Scope Note	`skos:scopeNote`=X attached to concept created for surrounding <DescriptorRecord>
<TreeNumber>*X*	implicitly indicates Broader Concept	`skos:broader` to concept with `rdf:ID`=X
<Term RecordPreferredTerm=”N”> <String>*B*	Non-preferred Label	`skos:altLabel`=B attached to concept with `rdf:ID` found in surrounding Descriptor
<SeeRelatedDescriptor> <DescriptorReferredTo> <DescriptorUI>*X* <DescriptorName>*Y*	Related Concept	`skos:related` to Concept with `rdf:ID`=X
<HistoryNote>X	Historical Note	`mesh:historyNote`=X (subproperty of `skos:historyNote`)

Data Item	Feature/function	Property/Class
<EntryCombination> <ECIN> *X* <ECOUT> *Y*	Compound Concept and special relation (see text). X and Y contain tags with the identifiers of one Descriptor/Qualifier pair in them	`mesh:CompoundConcept` `mesh:Qualifier` `mesh:main` `mesh:qualifier` (subclasses and subproperties of `skos:Concept` and `skos:broader`) `mesh:preferredCombination` (no parent)
<PublicMeSHNote>X	Note mixing historical and see also information	`mesh:publicMeSHNote`=X (subproperty of `skos:note`)
<PreviousIndexing>X	Historical Note	`skos:historyNote`
<ConsiderAlso>X	textual reference to other possible records	`mesh:considerAlso`=X (subproperty of `skos:note`)
<ActiveMeSHYear>X	Year in which the Descriptor was part of MeSH	`mesh:activeMeSHYear`=X (subproperty of `skos:editorialNote`)
<RecordOriginator>X	Thesaurus where the Descriptor comes from	`mesh:recordOriginator` (suproperty of `skos:note`)
<DateCreated>X	Date Descriptor was first created	`mesh:dateCreated`=X (subproperty of `skos:editorialNote`

Data Item	Feature/function	Property/class
<ActiveMeSHYear>	Year in which Descriptor was present in MeSH	`mesh:activeMeSHYear`
<DescriptorClass>	Classifies Descriptor into one of four numbered categories, including “topical descriptor” and “publication type”	`mesh:descriptorClass`
<RunningHead>	page header used in printed MeSH versions	`mesh:runningHead`
<LexicalTag>	lexical category of a <Term>	
<Abbreviation>	abbreviation of a <Term>	

Table 4. Mapping of representative MeSH Data Items to features and RDFS property/classes. Upper part lists standard features, the middle part specializations and lower part non-standard features. Omitted closing tags in Data Item column.

as “pathology” or “abnormalities”. They are combined with Descriptors to enable more specific article indexing (e.g. “Abdomen/abnormalities”). Secondly, so-called EntryCombinations relate a non-preferred Descriptor/Qualifier pair to a preferred Descriptor/Qualifier pair (or preferred Descriptor without Qualifier). This is comparable to but slightly different from the ISO 2788 “USE” relation, which can be used to point from a non-preferred non-compound term to a preferred compound term. The difference is that in MeSH the preferred concept is a compound.

Step B: map data items to SKOS. We mapped Descriptor to `skos:Concept` instances and sub-tags to properties of `skos:Concept` (see Table 4). Each child Descriptor is linked to its parent(s) - stated implicitly in the <TreeNumber> tag(s) - with `skos:broader`. We only map Descriptor names one time, removing the redundancy.

Because the MeSH Concepts and Terms are converted into `skos:prefLabel` and `skos:altLabel`s, information about the Concepts and Terms themselves is lost. One example is the Concept's "brd", "nrw" and "rel" relations. These cannot be mapped to the broader/narrower concept feature, because the Descriptor hierarchy is already mapped to that. Two more examples are the Term's <Abbreviation> and <LexicalTag>. Only in cases where it is valid to attach information about a Concept or Term to the Descriptor can this information be preserved by attaching it to the `skos:Concept`, which is not the case for a number of Concept and Term tags. An example where this *is* possible is with a preferred Concept's <ScopeNote>.

To support the use of Descriptor/Qualifier pairs in indexing we introduced classes `mesh:Qualifier` and `mesh:CompoundConcept` as subclass of `skos:Concept`. Qualifiers are a special class of Concepts because they do not have broader/narrower relations themselves. The properties `mesh:main` and `mesh:qualifier` are used to attach a Descriptor (`skos:Concept`) and Qualifier (`mesh:Qualifier`) to the CompoundConcept. By making the properties a subproperty of `skos:broader`, the CompoundConcepts become narrower concepts of their contained concept, so that queries for documents with that concept as subject will also return documents indexed with the CompoundConcept. For the `rdf:ID` of the CompoundConcept the unique Descriptor and Qualifier identifiers are concatenated. We used the same CompoundConcept class to represent <EntryCombination>s which we link with `mesh:preferredCombination`. This last property does not have a SKOS parent. The only candidate `skos:related` has a different semantics: it links preferred concepts that are related in meaning (a symmetric relation), while `mesh:preferredCombination` links a non-preferred concept to a preferred concept (asymmetric relation).

Step C: create conversion program. We created a SWI-Prolog program that parses the MeSH-XML file and converts it to SKOS RDF using the mappings from step B. The program takes a DescriptorRecord tag and converts it into a `skos:Concept`. It also converts the non-standard features of MeSH.

Case study summary. The case study took one analyst approximately two weeks to perform and was relatively complex because of the many non-standard features and ambiguities. We have not yet been able to confirm our decisions with MeSH experts. We learned that some thesauri have complex structures for which no SKOS counterparts can be found (e.g. information on Terms) and that for some features care is required in converting them in such a way that they are still usable for their original purpose (e.g. the CompoundConcepts).

7 Discussion and Evaluation

In this section we first evaluate our method and then discuss the applicability of the SKOS metamodel for representating thesauri. The case studies showed that the method gives appropriate guidance in identifying common features of thesauri. However, we found that two of our three cases had non-standard features which our method cannot anticipate. Further case studies should increase the number of identified non-standard features to be incorporated into the method. For the analysis of the meaning of some features it is necessary to investigate how the feature is used in practice (e.g. GTAA Categories). Conversion of concept-based thesauri should be simpler than term-based thesauri as SKOS is concept-based, but we cannot confirm this as MeSH is not typical of the first category. Although MeSH was not a good choice as a case study in this respect, it did help us in identifying the boundaries of applicability of SKOS (see below). A problematic type of feature are textual notes that mix several kinds of knowledge (e.g. <PublicMeSHNote> contains historical and see also information). Our method does not investigate if it is possible to separate them. We are currently unsure whether such an investigation will result in generic rules that can be incorporated in our method.

The SKOS metamodel itself seems applicable for representing resources which have considerable resemblance to the ISO 2788 standard. From the MeSH case we learned that SKOS does not have a standard class to represent compound concepts, although this is a feature that is defined in ISO 2788. A related ISO feature, the USE relation from non-preferred compound terms to preferred ones has no SKOS counterpart either. Thesauri such as IPSV and MeSH also represent management information about their terms (e.g. date of term creation) which cannot be represented within SKOS itself . One might argue that this information is not relevant to a thesaurus' content. It may represent information on a higher level of abstraction that should not be considered for conversion. However, SKOS does partly supports representing other types of management information e.g. with the `skos:changeNote` and `skos:editorialNote`. Besides management information, there is also additional content information on terms that cannot be represented in SKOS, such as the MeSH <LexicalTag>. If it is appropriate to represent additional information on terms, a solution is to introduce into SKOS a new class `skos:Term` as the range of `skos:prefLabel` and `skos:altLabel`. This would enable terms to be entities in themselves to which additional properties can be attached.

Lastly, we note that it is difficult to confirm whether or not a given RDF document is valid SKOS RDF. The draft SKOS Test Set[16] and implementation[17] can simplify this in the future.

[16] http://isegserv.itd.rl.ac.uk/cvs-public/~checkout~/skos/drafts/integrity.html

[17] http://www.w3.org/2004/02/skos/core/validation

Acknowledgements

This work was partly supported by NWO's CHIME and CHOICE projects. The authors wish to thank Stella Dextre Clarke for providing information concerning the IPSV and Eero Hyvönen for correspondence on his method. The authors also thank all participants of *public-esw-thes@w3c.org* who have contributed to the development of SKOS.

References

1. ANSI/NISO. Guidelines for the construction, format, and management of monolingual thesauri. Ansi/niso z39.19-2003, 2003.
2. International Organization for Standardization. Documentation - guidelines for the establishment and development of monolingual thesauri. Iso 2788-1986, 1986.
3. J. Goldbeck, G. Fragoso, F. Hartel, J. Hendler, B. Parsia, and J. Oberthaler. The National Cancer Institute's Thesaurus and Ontology. *Journal of Web Semantics*, 1(1), Dec 2003.
4. Eero Hyvönen. Miksi asiasanastot eivät riitä vaan tarvitaan ontologioita? why thesauri are not enough but ontologies are needed? (in finnish). *Tietolinja*, (2), 2005. ISSN 1239-9132, URL: http://www.lib.helsinki.fi/tietolinja/0205/index.html.
5. D. Johnston, S. J. Nelson, J. Schulman, A. G. Savage, and T. P. Powell. Redefining a thesaurus: Term-centric no more. In *Proc. of the 1998 AMIA Annual Symposium*.
6. A. Miles and D. Brickley (editors). SKOS Core Guide. W3C Public Working Draft, World Wide Web Consortium, November 2005. Latest version: http://www.w3.org/TR/swbp-skos-core-guide.
7. A. Miles and D. Brickley (editors). SKOS Core Vocabulary Specification. W3C Public Working Draft, World Wide Web Consortium, November 2005. Latest version: http://www.w3.org/TR/swbp-skos-core-spec/.
8. A. Miles, N. Rogers, and D. Beckett. Migrating Thesauri to the Semantic Web - Guidelines and case studies for generating RDF encodings of existing thesauri. Deliverable 8.8, SWAD-Europe, 2004. URL: http://www.w3.org/2001/sw/Europe/reports/thes/8.8/.
9. T. Peterson. *Introduction to the Art and Architecture Thesaurus*. Oxford University Press, 1994.
10. D. Soergel, B. Lauser, A. Liang, F. Fisseha, J. Keizer, and S. Katz. Reengineering thesauri for new applications: the AGROVOC example. *Journal of Digital Information*, 4(4), 2004.
11. L.F. Soualmia, C. Goldbreich, and S.J. Darmoni. Representing the mesh in owl: Towards a semi-automatic migration. In *Proc. of the 1st Int'l Workshop on Formal Biomedical Knowledge Representation (KR-MED 2004)*, pages 81–87, Whistler, Canada, 2004.
12. M. van Assem, M. R. Menken, G. Schreiber, J. Wielemaker, and B. Wielinga. A method for converting thesauri to rdf/owl. In S. A. McIlraith, D. Plexousakis, and F. van Harmelen, editors, *Proc. of the 3rd Int'l Semantic Web Conf. (ISWC'04)*, number 3298 in Lecture Notes in Computer Science, pages 17–31. Springer-Verlag, 2004.