MAKING USE OF UPPER ONTOLOGIES TO FOSTER INTEROPERABILITY BETWEEN SKOS CONCEPT SCHEMES

SALVADOR SANCHEZ-ALONSO
ELENA GARCIA-BARRIOCANAL

University of Alcalá
Ctra. de Barcelona km. 33,600 – Alcalá de Henares 28871 (SPAIN)
{salvador.sanchez, elena.garciab}@uah.es

The SKOS (Simple Knowledge Organization System) Core is a model for representing thesauri and similar types of knowledge organization systems as RDF graphs. Although it provides a basic framework for building concept schemes, SKOS does not carry the strictly defined semantics of formal ontology languages and thus has a number of shortcomings to fully port existing schemes to the Semantic Web. This paper introduces a mapping of SKOS metadata to an ontology-based intermediate model, whose main aim is to foster the semantic interoperability of different concept schemes. It has been achieved through the introduction of a common ground for the definition of concepts, based on the use of shared definitions already included in widely-used upper ontologies. This effort makes use of an upper ontology in particular: *OpenCyc*, the open source version of *Cyc*, which is currently one of the most complete general knowledge bases.

1. Introduction

The SKOS Core (Miles and Brickley, 2005a) is an application of the Resource Description Framework (RDF) that allows expressing a concept scheme as an RDF graph by using a number of terms. These terms are known as the *SKOS Core Vocabulary* (Miles and Brickley, 2005c). Concept schemes, as defined by SKOS, are "thesauri, classification schemes, subject heading lists, taxonomies, terminologies, glossaries and other types of controlled vocabularies". Thus, the metadata elements in the SKOS vocabulary allow to represent the content and structure of concept schemes (particularly those that have a specific structure described by the SKOS Guide) with the aim of promoting their use by Semantic Web applications.

One of the main advantage of SKOS is that it lifts any kind of organised description into an easily usable set of classes (Euzenat, Scharffe and Serafini, 2006). An example of the representation of a fragment of a concept scheme in SKOS is the extract of the *Art & Architecture Thesaurus* (http://www.getty.edu/research/conducting_research/vocabularies/aat/) shown in Table 1. Making use of SKOS, the information and structure of this extract can be represented as an RDF graph as shown in Figure 1. This example serves as an introductory illustration on how SKOS can be used to map to RDF a given thesaurus.

Table 1. An extract of the Arts & Architecture Thesaurus concept scheme: the term *castles*.

Term	castles
Used for	buildings or groups of buildings intended primarily to serve as a fortified residence of a prince or nobleman
Broader terms	fortifications
Narrower terms	châtelets moated castles qasrs
Related terms	fortification elements

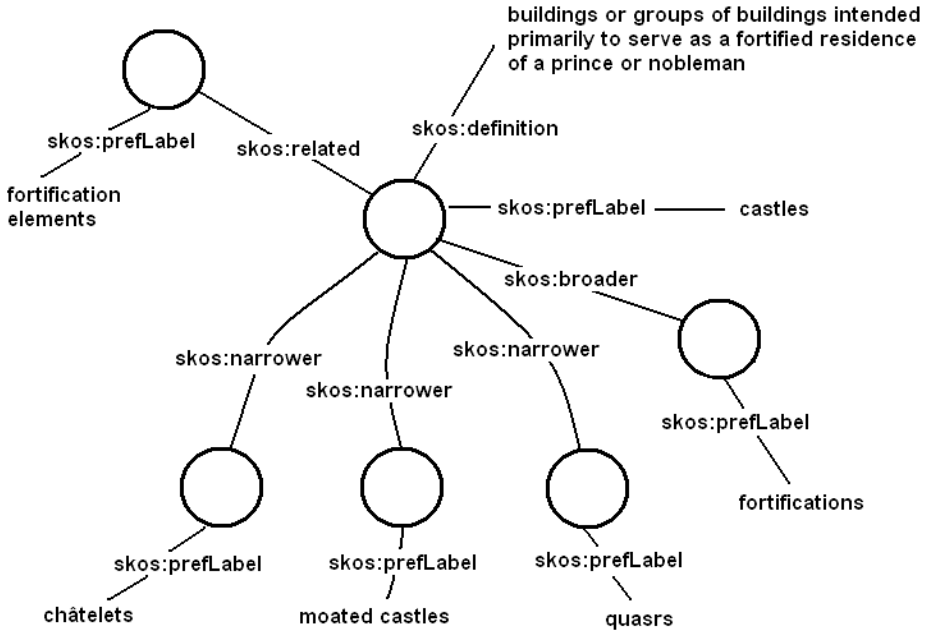

Figure 1. The same extract of the Arts & Architecture Thesaurus expressed as a SKOS RDF graph.

The use of RDF graphs as a representation mechanism has a number of benefits, such as allowing data to be linked to other RDF data by Semantic Web applications, or providing serialization capabilities for concept schemes to be encoded as a series of characters according to a number of RDF syntaxes (RDF/XML, N3/Turtle or N-Triple). However, although this is an important step towards the use of a particular vocabulary by applications, two important shortcomings can be identified:

- First, the representation of a concept scheme as an RDF graph is not, by itself, enough to make it interoperable because the meaning of the terms in the SKOS vocabulary is not formally defined. Providing formal definitions for SKOS terms would both prevent the inherent ambiguities in the interpretation of some terms, and ease the shared use of SKOS-based schemes by including more precise definitions.
- Second, the existence of similar concepts in different schemes suggests the possibility of establishing mapping criteria to foster the interoperability between them. Regarding this, the *SKOS Mapping Vocabulary Specification draft* (Miles and Brickley, 2005b) is oriented to give support to mappings between concepts from different schemes, but is currently in a very early stage. Another promising work in a similar direction is the *Inter-Thesaurus Mapping*, currently in draft version 0.1. (Miles and Matthews, n.d.). On the other hand, recent studies demonstrate that thesaurus mapping is not free of problems (Doerr, 2001).

These two shortcomings can be summed up in one sentence: SKOS does not provide strict computational semantics. That is to say, the representation of a concept scheme as an RDF graph can not be used as the basis for performing automated tasks associated to the knowledge represented in the scheme. In fact, when some degree of automation is desired, the provision of a specific ground for the delegation of tasks to automated or semiautomated systems is necessary. Formalisms such as description logics (Baader et. al, 2003) provide support for the explicit definition of terms and properties oriented to shared management, automated processing and reasoning based on specific inference mechanisms. The following sections will show how the use of ontologies can significantly improve the interoperability of concept schemes, as the inner description logics they

provide introduce the necessary degree of formalization.

The rest of this paper is structured as follows. Section 2 introduces the term *semantic interoperability* (Doan, Noy and Haleevy, 2004) and provides a specific definition for the purpose of this work. After that, a brief introduction to the field of ontologies and the benefits of their use for the knowledge representation of concept schemes is provided. Section 3 presents two ontology-based proposals aimed at fostering semantic interoperability between SKOS concept schemes. Section 4 completes one of the two the proposals introduced, presenting an in depth study about the most relevant categories in SKOS and suggesting mappings to terms in upper ontologies. Finally, section 5 provides conclusions and some directions for further research.

2. The role of ontologies

As it has been mentioned in the previous section, RDF representations are a big step towards permitting Semantic Web applications to use and manage concept schemes. However, a number of shortcomings were also pointed out. This section deals with the problem of semantic interoperability between thesauri and introduces ontologies as a useful tool towards attaining this goal.

Most thesauri include terms which are also included in other thesauri, sometimes with exactly the same meaning, sometimes with just a similar meaning and sometimes with a meaning significantly different. For example, the term *back* in a anatomy thesaurus would be defined as “the rear part of the human body, especially from the neck to the end of the spine”, while a sports glossary could perhaps include the same term with a slightly different meaning: “a position behind the front line of players”. It is also feasible to think that this latter glossary might include a second definition of the term, to designate “a player in the back position”. This example, that is illustrated in Figure 2, points out the fact that any term in a SKOS scheme can have different meanings and consequently refer to different terms in a knowledge base. The notation used for concept representation in Figure 2 separates the glossary (e.g. *Sports*) to which a concept belongs from the concept itself (e.g. *back*) by means of a colon.

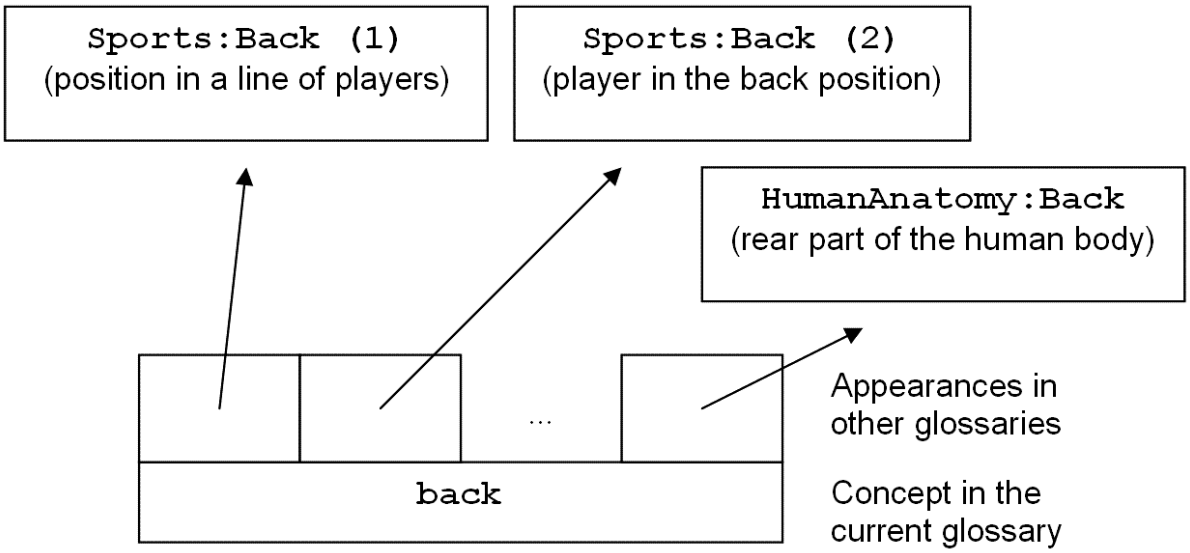

Figure 2. Different meanings of the term “back” according to different glossaries.

One of the main difficulties in attaining interoperability (as a general feature) is the lack of explicit, shared definitions that allow to unambiguously refer to a term. To overcome this problem, thesauri should include formal definitions of all their terms and relations, which should in turn be achieved by

making use of a specific formal language (i.e. mathematical or logical). Definitions like these would be then referred to as “semantic definitions”. Unfortunately, this is not the case of neither the terms in the SKOS vocabulary nor the terms in most thesauri, as they do not provide support for the so-called *semantic interoperability*. For the purpose of this work, semantic interoperability will be defined as “the use of explicit semantic descriptions to facilitate concept scheme integration with the main objective of fostering the automated or semiautomated use of the information”. In this work, ontologies are introduced as a remarkable tool to attain semantic interoperability in SKOS concept schemes.

In the field of philosophy, the term *ontology* is defined as the theory of objects and their ties. Therefore, the definition of a shared ontology for a given domain provides criteria for distinguishing different types of objects in the domain as well as their relations (Corazzon, n.d.). Outside philosophy, ontologies can be understood as conceptualizations that provide an appropriate context for the interpretation of concepts in a given domain. An often-cited definition by Gruber (1993) states that an ontology is “an explicit specification of a conceptualization”. In this sense, ontology engineering becomes of particular interest when applied to conceptual modeling.

The existence of ontology-based schemes in a domain of discourse is essential when some degree of automation is desired. The inner logics in the ontology allows automated systems to perform tasks according to the elements defined, which is the basis for applying the principles of Semantic Web in the domain of the ontology. However, creating a new ontology from scratch is a huge effort that might imply to define the (probably hundreds of) terms and relations that are needed before the elements in the current concept scheme can be explicitly defined and situated in the right place in the full hierarchy of concepts. To avoid defining time and time again all the concepts from which others derive, *upper ontologie*s can be used. Upper ontologies are large general knowledge bases that include definitions of concepts, relations, properties, constraints, and instances, as well as reasoning capabilities on these elements. They are limited to generic, high-level, abstract concepts, general enough to address a broad range of domains, not including concepts specific to given domains, or do not focusing on them. One of the major efforts in the field is *OpenCy*c (http://www.opencyc.org), an upper ontology “for all of human consensus reality” which includes more than 47,000 concepts, 300,000 assertions about them, an inference engine, a browser for the knowledge base and other useful tools. It is the open source version of the larger *Cy*c knowledge base (Lenat, 1995), a huge representation of the fundamentals of human knowledge.

An in-depth study of the SKOS vocabulary suggests its extension with the aim of correcting the shortcomings identified in section 1. However, although such an extension would help both to avoid ambiguities and to enable inter-thesaurus semantic interoperability, the solution to the problems pointed out in the preceding sections should be better focused as a non-invasive contribution. Non-invasive in the sense that the SKOS Core should not be modified as a result of this activity, but also, non-invasive in the sense that current SKOS schemes should not require modifications. To achieve the goals expressed before, we propose the use of formal representations to provide the SKOS terms with computational semantics, as well as the introduction of an intermediate ontology-based model built on top of the SKOS information. Both proposals stand on one upper ontology in particular, *OpenCyc*, but they could be easily adapted to others. It is worth mentioning that some work on the integration of conceptual structures into *upper ontologies* has been already carried out, like the research described in Sicilia et al. (2006), where Knowledge Management conceptual structures are integrated into the *OpenCyc* ontological base for the practical purpose of providing the required support for the development of intelligent applications.

3. Attaining semantic interoperability

In section 1, two disadvantages of the current state of SKOS were pointed out:

a) The lack of formalization of the terms in the SKOS Core. For example, the SKOS property *narrower* has been created to represent the fact that a concept “is more specific in meaning than other” but, what does *narrower* exactly means in computational terms? What are the implications of its formal definition?

b) The need for mapping criteria to foster the semantic interoperability between thesauri. If the

same concept can have different meanings across different vocabularies, is there a way to map a concept in a SKOS scheme to a term in an upper ontology that provides a formal definition for it? The existence of such mapping criteria would allow, e.g. to specify the exact meaning of the concept *back* in one glossary (where it refers to one and only one particular meaning) and thus would foster the reuse of this concept in related glossaries which would not need to redefine *back* again, but instead to link to it.

The first problem is addressed by proposing a set of precise definitions for the terms (classes and properties) in the SKOS vocabulary through mapping them to terms in an upper ontology (*OpenCy*c is used here as a case study). This approach is described in section 3.1 and applied to the modeling of the *semanti*c *relationships* properties. The solution described in section 3.1 represents a general mapping, although particular mappings stating something like "this term has this specific meaning in this context" can also be carried out. As part of this proposal, Section 4 provides an in depth analysis of the most relevant categories of elements in SKOS from the point of view of the semantic interoperability.

The second problem is addressed by defining an intermediate model to map the concepts in a SKOS scheme to terms in an upper ontology (specifically *OpenCyc*). Explicitly defining the terms in the SKOS vocabulary by using an ontology language both improves the effective integration of semantically heterogeneous thesauri and fosters the automated or semiautomated processing of SKOS schemes by Semantic Web applications in specific contexts of use, preserving, at the same time, the original SKOS information. This approach is described in more detail in sections 3.2 and 3.3.

3.1. Providing formal definitions for SKOS metadata elements

Metadata elements in the SKOS vocabulary are divided into six categories: *conceptual elements*, *labelling properties*, *documentation properties*, *semantic relationships*, *meaningful collections of concepts* and *subject indexing properties*. The properties in the *semantic relationships* category, in particular, are metadata elements aimed at "asserting semantic (paradigmatic) relationships between concepts". However, SKOS does not provide a clear, formal definition of what a semantic relationship is. In addition, some of the relationships are described in a purposefully vague language. Both factors somewhat hamper the use of the information on relationships by semantic Web applications. Table 2 shows the relationship elements in SKOS.

Table 2. SKOS relations.

SKOS relation	Definition	Comment
`semanticRelation`	A relation of meaning	Not to be used directly, but as a super-property for all properties denoting a relationship of meaning
`narrower`	The scope (meaning) of one concept falls completely within the scope of another	Narrower concepts are typically rendered as children in a concept hierarchy tree
`broader`	A concept that is more general in meaning than another	Broader concepts are typically rendered as parents in a concept hierarchy tree
`related`	(weak semantics) A concept with which there is an associative semantic relationship	Expresses the fact that two concepts are in some way related, and that the relationship should not be used to create a hierarchy

The class `skos:CollectableProperty`[1] supports a generic mechanism by which collections can be involved in semantic relationships (and other sorts of statement). However, the semantics of

[1] Both SKOS elements and OpenCyc terms and relations are shown in `courier` font.

this mechanism are not explicit in the SKOS Guide. In fact, even though each and every relationship in the SKOS Guide is informally defined, the lack of explicit formal descriptions can be considered a problem for some applications. To perform reasoning tasks on the knowledge defined, thus avoiding misinterpretation of terms across different thesauri, computational agents require machine-readable descriptions (in addition to the human-readable versions of the information). These can be provided in the form of explicit definitions in an ontology language such as OWL. Table 3 is a partial example on the effort to map SKOS relations to formally defined predicates in the *OpenCyc* knowledge base. Doing this, we are providing SKOS elements with a machine-consumption semantics that will disambiguate any possible interpretation.

Table 3. A mapping of SKOS relations to *OpenCyc* predicates.

SKOS relation	*OpenCyc* term	Comments
`semanticRelation`	`Predicate`	Either a property of things (unary) or a relationship holding between two or more things (n-ary).
`narrower`	`genls`	Relates a given collection to those collections that subsume it.
	`subSet`	Relates a set or collection SUB to a set or collection SUPER whenever the extent (the set consisting of all of its elements) of SUB is a subset of the extent of SUPER.
	`TaxonomicPredicate`	Used to help specify the position of a thing within one of the major taxonomies or hierarchies in the *OpenCyc* ontology.
`broader`	inverse of `genls`	Relates the subsumed collection to the subsumer collection.
`related`	not `TaxonomicPredicate`	Any predicate not used to help specify the position of a thing within one of the taxonomies in *OpenCyc*.

3.2. An intermediate model to map SKOS terms to upper ontologies

The same concept can have different meanings across different thesauri. To both avoid misinterpretations and foster automated reasoning on the terms of the thesauri, terms can be linked to a general knowledge ontology such as *OpenCyc*. This should be made without the need of modifying existing SKOS records for existing SKOS concept schemes. Figure 3 depicts how this can be done through a non-invasive intermediate model. In the example, inspired by an interesting study on the existence of different learning object conceptualizations by McGreal (2004), one concept in a particular thesaurus, *learning object* (Polsani, 2002), is linked to specific meanings depending on its different characterizations. The example assumes that a SKOS scheme has been created from the original concepts in the thesaurus. On top of this information, and probably performed by other persons (experts either in upper ontologies or in learning ontologies), a number of intermediate records can be built to link the concept *learning object* to terms formally defined in an ontology.

In this particular case, and following the mentioned discussion by McGreal, one organization could define learning object as “anything and everything” and thus link this concept to the term `oc:Thing` in *OpenCyc* (the prefix “oc” indicates that it is an *OpenCyc* term, so its meaning is unambiguously stated as a formal definition). In *OpenCyc*, `oc:Thing` is the top concept from which all the others derive. On the other hand, if we consider learning objects to be digital entities, they could be considered to be instances of `oc:ComputerFileCopy`, i.e. “information bearing things that contain digitally coded information readable by a computer”. Although this definition is controversial due to the dynamic nature of many learning objects, it serves the purpose of abstracting them as elements available at a given URI. Finally, an organization maintaining a domain ontology on learning

terms (such as the Learning Technology group at the IE Research Unit, http://www.cc.uah.es/ie/) could find useful to link the concept *learning object* to, for example, the term RLO in its ontology, RLO standing for reusable learning object. The prefix *IELearning* in Figure 3 indicates the origin of the ontology term.

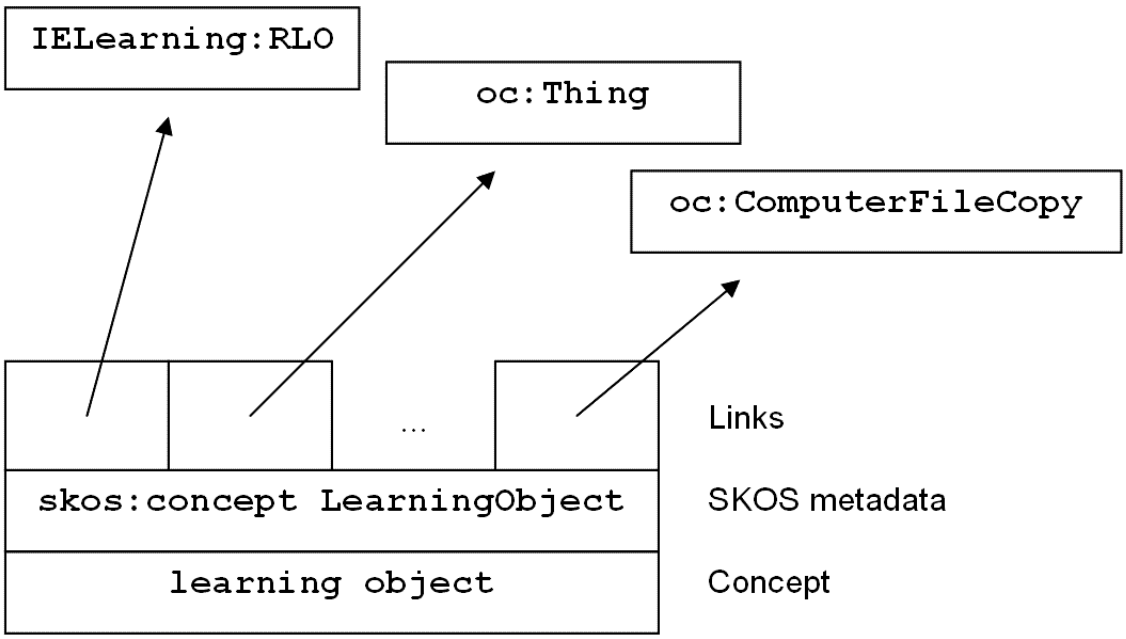

Figure 3. An intermediate model allows mapping a SKOS concept to terms in different ontologies.

This approach, that consists of building an extra layer of links to concepts, will not be considered complete until the reason why the `skos:concept` is being linked to a concept can be established. That is to say: the conditions under which a particular link is considered operational (or the context that defines those conditions) must be stated as part of the task of construction of the layer of links. Returning to our previous example, the term *back* can be linked to different concepts in different sport glossaries. Focusing on the different meanings that the term *back* has in two sports, football and soccer, it becomes clear that at least two links can be created to connect *back* to terms belonging to glossaries specific to each sport in particular. In soccer, for example, *back* is a synonym of *defende*r and is accordingly defined as “a player of the team that does not have possession of the ball”. On the other hand, the meaning of *back* in a football glossary is “a running back, either the halfback or the fullback”. Besides, the term *back* has a completely different meaning in human anatomy: “the rear part of the human body, etc.”. Using an intermediate layer of links, as proposed, can help to link the term *back* in a SKOS scheme to terms in different glossaries as shown in Figure 3, but the specific context for which one meaning in particular (of the three available in the example) is operational is not yet formally stated as part of the layer of links.

The knowledge base of *Cyc*, and consequently that of its open source version *OpenCyc*, is divided into locally-consistent contexts called *microtheories*. The concept `oc:Microtheory` serves to group a set of assertions (about time, topic, space, granularity, etc.) together that share some common assumptions, defining “an atemporal abstract informational thing that represents a context”. Consequently, the assertions in a `oc:Microtheory` constitute the content of that `oc:Microtheory`. It is important to indicate that “all the assertions stated to be true in one microtheory will also be true (by inference) in more specialized microtheories that depend on the content of that microtheory”.

This notion of context can be considered to be the “missing link” between a SKOS concept and the concepts to which it is linked to, and provides full sense (in semantic interoperability terms) to the

modeling of a layer of links. The information about the context allows to write assertions in the following manner: “in this particular context, this concept in a SKOS scheme can be linked to a concept in other glossaries or ontologies”, which is formally implemented by means of a ternary predicate to be used in quadruplets including information about the context, the SKOS concept and the linked concept in the following way:

```
(linkedTo oc:microtheory skos:concept oc:Thing)
```

The predicate `linkedTo`, specifically created for the purpose of this work, might be somewhat related or derived from `oc:ist`, a predicate used to relate an assertion to the microtheories in which that assertion is true. Figure 4 shows an updated version of the *back* modeling example where microtheories have been incorporated.

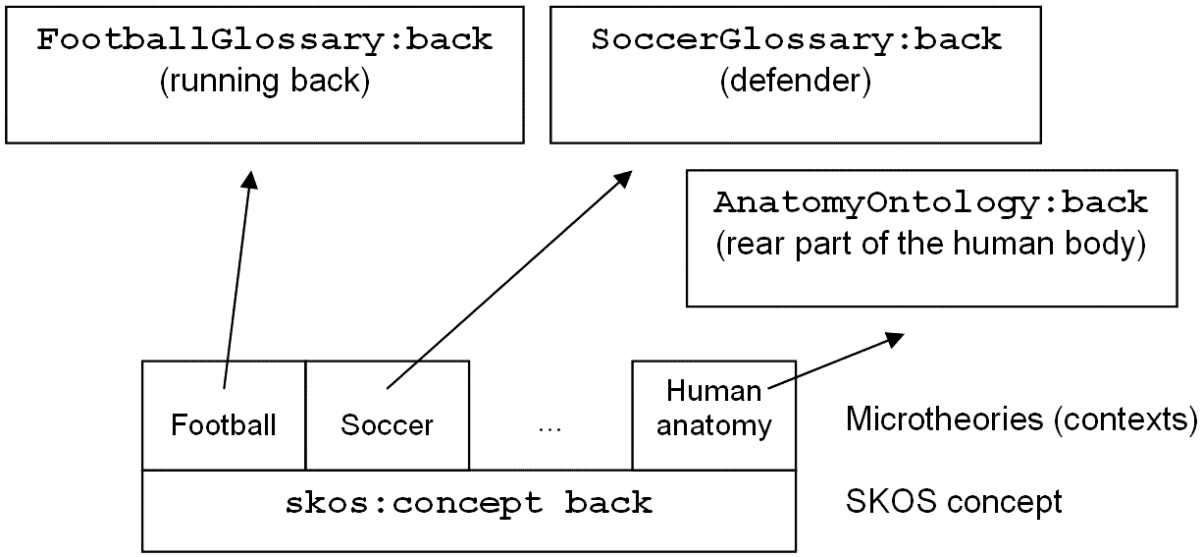

Figure 4. Using microtheories to map SKOS concepts to specific definitions.

Another informative example is that of the term “abdomen”. This term has different meanings depending on the animal which it is being applied to. This way, a human anatomy glossary might contain the following definition: “the belly, the part of the trunk between thorax and the perineum”, while for animals the definition of abdomen could be quite different. In hermit crabs, for example, abdomen is the “region of the body furthest from the mouth”, although in ants, abdomen is “the third section of the insect body (head, thorax, abdomen) which consists of the propodeum and metasoma”. In this example, the information that was, for the previous example, represented in Figure 4, can be formally expressed in the form of assertions like this:

```
(linkedTo Human-Microtheory abdomen HumanAnatomyGlossary:abdomen)

(linkedTo Animal-Crustacean-Crab-Microtheory abdomen CrabGlossary:abdomen)

(linkedTo Animal-Insect-Ant-Microtheory abdomen AntGlossary:abdomen)
```

To conclude, it is significant to mention that in a generic microtheory of animals (e.g. `Animal-Microtheory`), that is to say, a context where we were dealing exclusively with concepts applicable to animals, the two last assertions might be valid but the first one would never be. This is because `Human-Microtheory` does not derive from the more general `Animal-Microtheory`, but `Animal-`

`Crustacean-Crab-Microtheory` `Animal-Insect-Ant-Microtheory` yet do derive: they are both subsumed by `Animal-Microtheory`.

3.3. Linking terms in SKOS schemes to upper ontologies

To link terms in a SKOS scheme to terms in an upper ontology such as *OpenCyc*, a method described elsewhere (Abran et al., n.d.) can be used. This process can be roughly described in four steps:

1. Find one or several terms that subsume the category under consideration.
2. Check carefully that the mapping is consistent with the rest of the subsumers inside the upper ontology.
3. Provide the appropriate predicates to characterize the new category.
4. Edit it in an ontology editor to come up with the final formal version.

This process has the advantage of allowing the individual work of an expert, whose outcomes can then be contrasted with the work of others. The results of the process are much more efficient and structured than engineering a new ontology, since the argumentation against or in favor of a given concept or predicate is put in the formal context of an upper ontology.

An innovative approach, aimed at the automated population of knowledge bases from information in the Web, is being explored by Shah et al. (2006). This approach, which continues the research work by Matuszek et al. (2005) about the population of *Cyc* from the Web through a method to assist in entering the knowledge, should be kept in mind as a reference to explore assisted methods to create links from SKOS concepts to terms in upper ontologies.

4. Other SKOS categories

Apart from the *semantic relationships* category, the metadata elements in the SKOS vocabulary are classified into other six categories: *Conceptual elements*, *Labelling properties*, *Documentation properties*, *Concept schemes*, *Meaningful collections of concepts* and *Subject indexing elements*. This section studies the semantic implications of the most relevant terms in those categories and discusses their implications in the light of the definitions in the *OpenCyc* knowledge base.

4.1. Conceptual elements

Only one element forms the category here referred to as *conceptual elements*: the `skos:Concept` class. Concept, defined in SKOS as “an abstract idea or notion; a unit of thought in SKOS”, allows to assert that a particular resource is itself a concept. This `skos:Concept` compares to the universal collection `oc:Thing` in *Opencyc*. `oc:Thing` is the collection which contains everything there is: every thing in the *Cyc* ontology –every `oc:Individual` (of any kind) and every `oc:Collection`– is an instance of `oc:Thing`. Similarly, every `oc:Collection` is a subcollection of `oc:Thing`.

4.2. Labelling properties

SKOS Core *labellling properties* (`skos:prefLabel`, `skos:altLabel`, `skos:hiddenLabel`, `skos:prefSymbol` and `skos:altSymbol`) are aimed at helping to assign some sort of token to a resource. As the tokens are “intended to be used to denote the resource in natural language discourse and/or in representations intended for human consumption”, the semantic implications of the elements in this category are few (if they can be considered useful at all). It becomes difficult to imagine inferences based on the information provided by elements explicitly defined as *intended for human consumption*.

Nevertheless, at least a weak inference can be deduced from the existence of labelling information. When e.g. displaying information about a resource that includes more than one labelling property, an inference mechanism can be programmed to always choose a preferred name over any other kind of name. In this manner, the preferred label will be always chosen to refer to the concept in

detriment of other labels that could exist. This behaviour can be modeled by making use of the predicate `oc:preferredNameString`. The existence of a triplet (`oc:preferredNameString`, STRING, THING) states that the STRING is a preferred name to use when referring to THING. Similar inferences can be programmed for the other labelling properties (e.g. to perform multilingual or symbolic labelling) even though their usefulness is arguable.

4.3. Documentation properties

The hierarchy of *documentation properties* in SKOS (`skos:note`, `skos:definition`, `skos:scopeNote`, `skos:example`, `skos:historyNote`, `skos:editorialNote`, `skos:changeNote`) is composed of seven properties that can be used "to add human-readable documentation to the description of a concept". Used only to label a concept, the usefulness of these properties can be as arguable as the information about *labelling properties*. However, the so-called 'recommended usage patterns' for the SKOS Core documentation provide room for more complex modeling. There are three recommended usage patterns:

- Documentation as an RDF Literal: the simplest pattern for using the SKOS Core documentation properties. In this pattern, the property value is represented as an RDF literal.
- Documentation as a Related Resource Description: the documentation is structured as a related resource description, what allows to represent complex documentation structures.
- Documentation as a Document Reference: allows to refer to documentation that is itself a document, via the URI of that document.

From the point of view of introducing some kind of inference mechanisms, the more interesting pattern is the *documentation as a related resource description* one. This pattern allows to include information about the properties of the documentation itself, such as the creator(s) of the documentation, dates related to it, or its intended audience, among others. In an upper ontology, several predicates can be found to represent the same or very similar knowledge. For example, the predicate `oc:createdBy` relates something to its creator(s). In this way, any triplet (`oc:createdBy` THING AGENT) means that AGENT is "one of the people, corporations, publishers, etc., responsible for the invention or bringing into being of THING". Similarly, date information can be modeled through the use of many different predicates that provide support for information related to a date, as for example:

- `oc:myCreationDate`, a binary predicate to be used in triplets (`oc:myCreationDate` CONSTANT DATE), informs that the atomic term that is CONSTANT was created at DATE. Depending on the precision of the bookkeeping information stored for CONSTANT, DATE may have varying degrees of accuracy.
- `oc:dateOfPublication-CW`, a binary predicate whose meaning for a triplet (`oc:dateOfPublication-CW` CW DATE) is that the CW (which stands for Conceptual Work) was published on DATE.

But in general, the highest level predicate of the hierarchy of properties that might help to model this information is `oc:ComplexTemporalPredicate`, a specialization of `oc:BinaryTemporalRelationPredicate`, whose instances relate temporal things (instances of `oc:TemporalThing`). A complex temporal predicate might be used to relate "complicated temporal objects such as events, tangible objects, and proper time intervals".

4.4. Concept schemes

After what is stated in the SKOS Core Guide, a concept can be defined either in relation to other concepts (as part of an internally coherent concept scheme), or as a stand-alone resource. The SKOS Guide defines a *concept scheme* as "a set of concepts, optionally including statements about semantic relationships between those concepts", and provides the class `skos:ConceptScheme` for authors to assert "that a resource is a concept". However, this definition of concept scheme is very similar to the earlier mentioned definition by Gruber (1993): "an explicit specification of a

conceptualization". Consequently, providing information for the `skos:ConceptScheme` metadata element may be used to assert that a particular resource is itself an ontology. Using the `skos:hasTopConcept`, ontologies can be linked to their top level concepts in the following way:

```
(skos:hasTopConcept OpenCyc oc:Thing)
```

It is also possible to assert that a concept belongs to a particular concept scheme, by using the `skos:inScheme` property. For example:

```
(skos:inScheme BillGates OpenCyc)
```

This kind of assertion can be useful to situate terms when several ontologies are being used, for example whenever there is a need for working with multiple domains and concepts (and/or relationships) cross-cut domains.

*4.5. Meaningful collection*s *of concept*s

This category of metadata elements allows to define meaningful groupings of concepts which are called *collections*. To assert labelled collections, the vaguest form of meaningful collection of concepts in SKOS, it is just necessary to use the `skos:Collection` class to set the collection name and the `skos:member` property to relate that collection to the concepts which belong to it. No other meaning, apart from membership, can be inferred from such declarations. The following RDF code is an example of the definition of a labelled collection of Italian opera composers:

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:skos="http://www.w3.org/2004/02/skos/core#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

  <skos:Collection>
    <rdfs:label>italian opera composers</rdfs:label>
    <skos:member rdf:resource="http://www.example.com/concepts#verdi"/>
    <skos:member rdf:resource="http://www.example.com/concepts#mascagni"/>
    <skos:member rdf:resource="http://www.example.com/concepts#bellini"/>
    <skos:member rdf:resource="http://www.example.com/concepts#puccini"/>
  </skos:Collection>

  <skos:Concept rdf:about="http://www.example.com/concepts#Verdi">
    <skos:prefLabel>Giusseppe Verdi</skos:prefLabel>
  </skos:Concept>

  <skos:Concept rdf:about="http://www.example.com/concepts#mascagni">
    <skos:prefLabel>Pietro Mascagni</skos:prefLabel>
  </skos:Concept>

  <skos:Concept rdf:about="http://www.example.com/concepts#bellini">
    <skos:prefLabel>Vincenzo Bellini</skos:prefLabel>
  </skos:Concept>

  <skos:Concept rdf:about="http://www.example.com/concepts#puccini">
    <skos:prefLabel>Giacomo Puccini</skos:prefLabel>
  </skos:Concept>

</rdf:RDF>
```

The class `skos:Collection` can be mapped to the concept `oc:SetOrCollection`, a term in

OpenCyc describing instances “intrinsically associated with an intensional criterion for membership, which is a property or group of properties possessed by all of its elements”. At the same time, the `skos:member` property can be linked to `oc:elementOf`, a predicate that in *OpenCyc* relates a `oc:Thing` and a `oc:SetOrCollection` meaning that the `oc:Thing` is an element of the given mathematical set or collection. The class `skos:OrderedCollection`, a subclass of `skos:Collection`, defines an ordered collection of concepts whose members are linked from a `skos:memberList` property to the instance of the collection. In ordered collections a membership rule applies, implemented through a logical function named `elementOfList`. The term `oc:TotallyOrderedCollection` in *OpenCyc*, “a collection whose instances are conventionally regarded as being ordered by some relation”, provides the formal definition for the same concept and can thus be linked to it by using the method described earlier. This kind of modeling is richer as it allows to link the relationship to the kind of ordering via the `oc:orderingRelation` predicate, which permits, for example, to implement a routine that looks for all the collections that follow the same sorting criteria.

Table 4. Mapping SKOS collection concepts to terms in OpenCyc.

SKOS concept	Definition	OpenCyc concept
`Labellled collection`	A meaningful collection of concepts	`oc:SetOrCollection`
`Ordered collection`	An ordered collection of concepts, where both the grouping and the ordering are meaningful	`oc:TotallyOrderedCollection`
`member`	A member of a collection	`oc:elementOf`

In SKOS schemes collections can also be nested, but this capacity is a mere way of arranging their declarations and, consequently, has not semantic implications. Table 4 summarizes the previous discussion about collections.

4.6. Subject indexing properties

The properties `skos:subject` and `skos:primarySubject` in the SKOS Core can be both used for subject indexing of information resources on the Web. Also the inverse properties `skos:isSubjectOf` and `skos:isPrimarySubjectOf` can be used to make the same assertions but in the contrary direction. The difference between `skos:subject` and `skos:primarySubject` lies in the fact that one information resource may have more than one `skos:subject`, only one (or none) of them being the `skos:primarySubject`.

Even though the subject indexing properties can be used for formal description of terms, the scope of *subject* in SKOS is much broader, and includes any relation to very diverse resources. In ontology based representations, these properties could be mapped to (almost) any binary predicate. Obviously, such an open mapping would not provide much information, but in *OpenCyc* the predicate `oc:subjectOfInfo`, provided to link any `oc:Thing` to some `oc:InformationStores`, fits well with indexing information and can be used as concept of reference to delimit the scope of these properties.

5. Conclusions and further research directions

SKOS has proven a powerful yet simple vehicle for presenting and sharing terminology, but it does not provide any form of computational semantics. The effort described in this paper is the first step to the description of a complete non-intrusive model oriented to foster semantic interoperability among thesauri using SKOS schemas. The main purpose is to provide SKOS schemes with computational semantics, as in the current situation, the representation of a concept scheme as an RDF graph can not be used as the basis for performing automated tasks associated to the knowledge represented in

the scheme. Current efforts in the field insist in the construction of inter-thesauri mappings but do not focus on semantic interoperability. Herein, the lack of a computational semantics in the SKOS Core, has been approached from two directions: proposing a set of precise definitions for the terms in the SKOS vocabulary through mapping them to terms in an upper ontology, and defining an intermediate model to map the concepts in a SKOS scheme to terms in an upper ontology. In both cases, *OpenCyc* has been used as a case study.

Regarding the use of upper ontology terms from *OpenCyc*, the opinion of other experts will be required in order to validate the links between the SKOS elements and the corresponding *OpenCyc* classes and properties. The authors consider these opinions very valuable and thus are open to positive feedback on the precision and usefulness of the definitions included. Future work should gather feedback from ontology experts and other research units working in similar projects qith the objective of publishing a working draft describing the problems found, the lessons learned, and proposing future actions on this topic.

Acknowledgments

This work is supported by the LUISA EU project (FP6-027149) and also receives support from the University of Alcalá (project UAH-PI2005-070) and the Spanish Ministry of Education (project TSI2004-21263-E).

References

Abran, A., Cuadrado, J.J., Garcia-Barriocanal, E., Mendes, O., Sanchez-Alonso, S. and Sicilia, M.A. (n.d.) Engineering the ontology for the Software Engineering Body of Knowledge: Issues and Techniques, in Ontologies for software engineering, Springer Verlag (in press)

Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D. and Patel-Schneider, P.F. (2003). The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press.

Corazzon, R. (n.d.). Ontology. A resource guide for philosophers. Retrieved November 1, 2005, from http://www.formalontology.it

Doan, A., Noy, N. and Haleevy, A. (2004). Introduction to the special issue on semantic integration. SIGMOD Record, 33 (4).

Euzenat, J. Scharffe, F., Serafin, L. (coord.) (2006) Deliverable D2.2.6: Specification of the delivery alignment format, version 1.1. Knowledge Web: Realizing the Semantic Web. Project KWEB EU-IST-2004-507482. Retrieved March 30 from http://www.inrialpes.fr/exmo/cooperation/kweb/heterogeneity/deli/kweb-226.pdf

Doerr, M. (2001). Semantic problems of thesaurus mapping. Journal of Digital Information, 1(8).

Gruber, T. R. (1993). A translation approach to portable ontologies. Knowledge Acquisition. 5(2) 199–220.

Lenat, D. B. (1995). Cyc: A Large-Scale Investment in Knowledge Infrastructure. Communications of the ACM, 38(11) 33–38.

Matuszek, C., Witbrock, M., Kahlert, R., Cabral, J. Schneider, D., Shah, P. and Lenat, D. (2005). Searching for Common Sense: Populating Cyc from the Web. In Proceedings of the Twentieth National Conference on Artificial Intelligence, Pittsburgh, Pennsylvania.

McGreal, R. (2004). Learning Objects: A Practical definition. International Journal of Instructional Technology and Distance Learning 1(9).

Miles, A. and Brickley, D. (editors) (2005). SKOS Core Guide: W3C Working Draft 2 November 2005. Retrieved March 20, 2006, from: http://www.w3.org/TR/swbp-skos-core-guide/

Miles, A. and Brickley, D. (editors) (2005). SKOS Mapping Vocabulary Specification. Retrieved March 20, 2006, from: http://www.w3.org/2004/02/skos/mapping/spec/

Miles, A. and Brickley, D. (editors) (2005). SKOS Core Vocabulary Specification: W3C Working Draft 2 November 2005. Retrieved March 20, 2006, from: http://www.w3.org/TR/swbp-skos-core-spec/

Miles, A. and Matthews, B. (n.d.) Inter-Thesaurus Mapping draft version 0.1., deliverable number 8.4. of EU Project number: IST-2001-34732, “Semantic Web Advanced Development for Europe” (SWAD-Europe). Retrieved March 20, 2006, from: http://www.w3c.rl.ac.uk/SWAD/deliverables/8.4.html

Polsani, P.R. (2002). The Use and Abuse of Reusable Learning Objects. Journal of Digital Information, 3 (4).

Shah, P., Schneider, D., Matuszek, C., Kahlert, R.C., Aldag, B., Baxter, D., Cabral, J., Witbrock, M. and Curtis J. (2006). Automated Population of Cyc: Extracting Information about Named-entities from the Web. In Proceedings of the 2006 FLAIRS Conference, Melbourne, FL.

Sicilia, M.A., Lytras, M., Rodríguez, E. and García-Barriocanal, E. (2006). Integrating descriptions of knowledge management learning activities into large ontological structures: A case study. Data & Knowledge Engineering, 57(2) 111-121.