A practical approach to model classification schemes with OWL ontologies

Ma. Auxilio Medina[1], J. Alfredo Sánchez[2], J. de la Calleja[1], Antonio Benitez[1]

[1]Universidad Politécnica de Puebla
Tercer Carril del Ejido Serrano S/N
Juan C. Bonilla, Puebla, México
[2]Universidad de las Américas Puebla
Ex-Hacienda Santa Catarina Mártir S/N
San Andrés Cholula, Puebla, México
{mmedina,abenitez,jdelacalleja}@uppuebla.edu.mx
j.alfredo.sanchez@gmail.com

Abstract. Classification schemes are used in libraries to organize digital resources or physical materials. They are reference systems formed by hierarchies of topics. Topics define structured vocabularies that are often used as indexes to explore collections. This paper reports on the potential of ontologies to model classification schemes. The links between topics and the topics themselves are stored in a hierarchy of classes that has a human comprehensible and machine-processable representation. Ontologies support consistency maintenance, definition of new classes, representation of relationships between individuals from different classes and offer the possibility to integrate semantic search mechanisms. An excerpt of the 1998 ACM Computing Classification System is used as a test bed for the proposed model. The results outline the expressiveness of the ontology against the traditional classification schema. The ontology is built in Protégé editor and this is validated with Pellet reasoner. A web accessible application allows users to make queries about ontology classes, properties, individuals and metadata.

Keywords: content classification, classification schemes, ontologies, semantic web technologies

1 Introduction

"Content classification is the process of analyzing a document and adding metadata that describe that document which are sourced from a taxonomy or other form of controlled vocabulary [15]". A classification system is a method used by librarians to organize multimedia collections.

The Dewey Decimal Classification (DDC) [6], the Library of Congress Classification (LCC) [5] and the Universal Decimal Classification (UDC) [3] are common examples of general classification schemes, that is, these schemes try to cover all the knowledge fields. DDC is widely used in primary, secondary and public libraries while university and research libraries often work with LCC [17].

UDC is a multilingual classification scheme script codes independent. Collections indexed by UDC can be found in databases and online public access catalogs (OPACs).

At present, classification schemes are distributed in multiple formats, from print to web browsable and machine readable [13]. This paper reports on the potential of ontologies to model classification schemes. Classes, properties, restrictions and individuals form ontologies that can be exploited by syntactic and semantic search engines. An excerpt of the 1998 ACM Computing Classification System (CSS) is used to show our approach[1] . An OWL 2 encoding is explored.

The paper is organized as follows: Section 2 describes the basic features of classification schemes. A quick view on ontologies is presented in Section 3. Section 4 includes related work. Section 5 presents our approach to model classification schemes with ontologies. Section 6 shows the results. Finally, Section 7 includes conclusions and suggests future directions of our work.

2 Features of classification schemes

Classification schemes are used in libraries to organize digital resources or physical materials. They are reference systems formed by hierarchies of topics. Topics define structured and controlled vocabularies to explore collections [18].

Classification schemes are a kind of simple knowledge organization systems (SKOS) [10]. It is assumed that these schemes are static systems, they do not often change. The definition of new subject categories originates new versions of a classification scheme. As a way of illustration, consider the 1998 ACM Computing Classification System (CSS), this has been used during fourteen years; previous versions were published in 1964, 1982, 1983, 1987 and 1991 [4].

Some features of classification schemes are the following [8]:

- Classification schemes provide a conceptual base for knowledge-based systems
- Collections are organized by subject categories identified by codes based on numbers, letters or a combination of both
- There is a unique identifier for each category
- The main categories are divided into subcategories, which in turn are divided into further subcategories, and so on. Categories form a hierarchical structure. The more specific a category is, the deeper level assigned to it
- Classification schemes can be *specific* if they represent a knowledge field or *general* if diverse knowledge fields are taken into account
- Information retrieval tasks make use of the information of the classification scheme

The domain of interest in CSS is Computer Science. There are 11 main categories and 72 subcategories organized in four levels [4]. Categories at level 1 are identified with a capital letter [A-K], the identifiers of subcategories at level

[1] ACM refers to the Association for Computing Machinery

2 use a letter, a period and an arabic number or the *m* letter; two numbers are used at level 3. Subcategories at level 4 do not have identifiers, instead, they are presented as an ordered list divided into two sections: *subjects* and *proper nouns*. As a way of illustration, Table 1 shows the main categories of CSS and some subcategories.

All the CSS subcategories have two common features:

1. A subcategory called *General* whose identifier finishes with the number zero. This is used when the name of the category is enough to classify a bibliographic resource.
2. A subcategory called *Miscellaneus* denoted with the *m* letter. This is used to express that a resource can belong to more than one of the subcategories.

This paper reports on the potential of ontologies to model classification schemes. A quick view on ontologies is presented in Section 3. There are some advantages of using ontologies, such as the links between topics and the topics themselves are stored in a hierarchy of classes that has a human comprehensible and machine-processable representation.

3 A quick view on ontologies

An ontology is a set of descriptive statements about some part of the world. [12]. In order to formulate knowledge explicitly, it is assumed that it consists of elementary pieces often referred as *statements* or *propositions*.

The basic elements of ontologies are the following ones [9]:

- *Axioms*.- The statements that an OWL ontology expresses. The assertion term is also used to refer to axioms. In Descriptive Logics, axioms are called facts.
- *Entities*.- The elements used to refer to real-world objects, they are the result of the abstraction task that developers do.
- *Expressions*.- The combinations of entities that form complex descriptions from basic ones. Names of entities can be combined into expressions using constructors.

Ontologies can be represented in different languages such as XML[2], XML-Schema, RDF[3], RDF-Schema, SWRL[4] or OWL[5]. In OWL, (specifically the second version of this language abbreviated as OWL 2), objects are denoted as *individuals*, categories as *classes* and relationships as *properties* [2].

Besides the machine-readable encoded hierarchical relationships of ontologies, they maintain information about properties and value restrictions. In a semantic web application, ontologies support the following tasks [1]:

[2] XML refers to eXtensible Markup Language

[3] RDF is an abbreviation of Resource Description Framework

[4] SWRL comes from semantic web rule-based language

[5] OWL refers to Ontology Web Language

- Consistency checking
- Completion (automatic inclusion/exclusion of properties)
- Interoperability support
- Support structured and customized searching
- Explote generalization/specialization relationships

There are two types of tools addressing the main stages of the ontologies lifecycle: *ontology editors* to create and edit ontologies and *reasoners* to query ontologies for implicit knowledge, i.e. to determine whether a statement is a logical consequence in the ontology.

A widely used OWL editor is Protégé, a free open-source editing framework developed at Stanford University [11]. For reasoning within OWL DL some systems are Fact++ by the University of Manchester, Hermit by Oxford University Computing Laboratory, Pellet by Clark & Parsia, LLC, and RacerPro by Racer Systems [9].

4 Related work

This section describes some related works that use semantic web technologies to represent classification schemes.

In the European Educational Research Quality Indicators project (EERQI), multilingual semantic techniques are used to locate specific information from unstructured documents. A crawler searches and processes educational research publications on the web based on link analysis. As in our approach, Dublin Core based XML format is used for metadata [16].

The Semaphore classification server has mechanisms to analyze and classify text by adding tags to documents, web pages or reports. This is considered a semantically enhanced system due to it performs entity extraction to support faceted search. Metadata are extracted automatically. An approach based on rules defines the criteria of content classification; rules can be applied to metadata. An ontology-driven navigation is supported [15]. Because Semaphore is a product of the Smartlogic company, there is no information about the ontology representation neither of the metadata.

An issue about UDC and its modeling using relational databases is presented in [1]. Since a technical point of view, the expressiveness of this classification scheme is analyzed to support interoperability. The use of CSS to construct OWL ontologies is presented in [14]. These ontologies support the visualization of classified documents from the ACM Digital Library. Authors use relationships between classes and keywords to build a spherical 3D information surface. The details of how to model the CSS categories into ontologies are not included.

5 Modeling classification schemes with ontologies

Ontologies support consistency maintenance, flexibility to define new classes, representation of relationships between individuals and offer the possibility to integrate semantic search mechanisms. An excerpt of the 1998 ACM Computing Classification System is used as a test bed for our approach.

Ontology classes are sets of individuals. In this work, they represent the categories of classification schemes. The steps to model a classification scheme with ontologies are presented in the following sections. An OWL 2 encoding with OWL/XML syntax is used [9]. OWL 2 is an ontology language for the semantic web with formally defined meaning. OWL 2 follows the open-world assumption, that is if some fact is not present in the knowledge database, it may simply be missing (but possibly true).

5.1 Construction of the hierarchy of classes

Classes are organized in a hierarchy by means of subclass relationships. Ontology elements are identified by IRIs[6]. The main categories of the classification schemes are defined as direct descendants of the `Thing` class, this is the main class in Protégé OWL Plugin [11]. Then, it is necessary to add the classes at level 2 with subclass relationships with the classes at level 1 and go on. Subclass relationship is *transitive* and *reflexive*.

Example 1 shows that `Software engineering` is a subclass of `Software`.

Example 1.

```
<owl:Class> <Class IRI=‘‘Software engineering’’
<Class IRI=‘‘Software’’>< /owl:Class>
```

The classes that belongs to a hierarchy are called *named classes*. Thus, an ontology that represents a classification scheme of *n* categories would have at least *n+1* named classes.

Classes are not considered disjoint unless there is other evidence. The aggregation of `AllDisjointClasses` predicates is necessary to express explicitly that all the classes in each level are disjoint. Example 2 express that the `Programming techniques`, the `Software engineering` and the `Programming language` classes are disjoint. However, if the remaining subclasses of are not included, a reasoner would infer that there are only three subclasses of the `Software` class, which is incorrect.

Example 2.

```
<DisjointClasses>
<Class IRI=‘‘Programming techniques’’/ >
<Class IRI=‘‘Software engineering’’/ >
<Class IRI=‘‘Programming languages’’/ >
</DisjointClasses>
```

A hierarchy of classes is an ontology by its own. However, to construct an environment for a semantic application, another ontology elements such as entities and properties can also be modeled.

[6] IRI is an abbreviation of International Resource Identifier, a kind of URI (Unified Resource Identifier)

5.2 Integration of properties

In this section it is assumed that the Book, Author and Editorial classes are also defined at level 1 of the ontology. These disjoint classes are added to show the potential of the incorporation of properties. Properties are binary relationships between ontology elements. There are `object` and `datatype` properties.

Object properties express the relationships between two individuals. The order in which the individuals are written is important. Example 3 shows the definition of the `hasAutor` property, that establishes a relationship between an individual of the `Author` class (`Asunción Gómez Pérez`) and an individual of the `Book` class (`Ontological engineering`). An `ObjectPropertyAssertion` need to be added for each author.

Example 3.

```
<ObjectPropertyAssertion>
<ObjectProperty IRI=‘‘hasAuthor’’/ >
<NamedIndividual IRI=‘‘Ontological engineering’’/ >
<NamedIndividual IRI=‘‘Asunción Gómez Pérez’’/ >
</ObjectPropertyAssertion>
```

Domain and range classes can be associated to properties. The domain and the range of the `hasAuthor` property are the `Author` and `Book` classes, respectively. A domain (or range) statement allows a reasoner to infer further knowledge.

New classes are defined by using properties with the individuals that share those properties. Property hierarchies can also be constructed. Example 4 shows a sub property of the `hasAuthor` property.

Example 4.

```
<SubObjectPropertyOf>
<ObjectProperty IRI=‘‘hasMainAuthor’’/ >
<ObjectProperty IRI=‘‘hasAuthor’’/ > </SubObjectPropertyOf>
```

Two properties are disjoint if there are no two individuals that are interlinked by both properties. The fact that two individuals are related via a property carries implicit additional information about these individuals, in particular, a reasoner might infer class membership. Example 5 expresses that every book has only one edition by characterizing the `hasEdition` property as *functional*.

Example 5.

```
<FunctionalObjectProperty>
<ObjectProperty IRI=‘‘hasEdition’’/ >
</FunctionalObjectProperty>
```

A subtype of data properties are *qualification properties*, they express existential or universal qualifications. Qualifications are written with `someValuesFrom` and `allValuesFrom` properties. The example 6 indicates that a book must have at least one author.

Example 6.

```
<EquivalentClasses> <Class IRI="Book"/ >  <ObjectSomeValuesFrom>
<ObjectProperty IRI=‘‘hasAuthor’’/ > <Class IRI=‘‘Author’’/ >
</ObjectSomeValuesFrom> </EquivalentClasses>
```

Data properties relate individuals to data values. The statements of example 7 shows a data property. This assigns a data type and a value to the `hasEdition` data property. Data types are often taken from XML Schema.

Example 7.
```
<DataPropertyAssertion>
<DataProperty IRI=‘‘hasEdition’’/ > <NamedIndividual IRI=‘‘Ontological
engineering’’/ >
<Literal datatypeIRI=‘‘http://www.w3.org/2001/ XMLSchema#integer’’>
1 </Literal> </DataPropertyAssertion>
```

New classes can be defined by restrictions on datatype properties. Example 8 shows domain and range restrictions on the `hasEdition` property.

Example 8.
```
<DataPropertyDomain> <DataProperty IRI=‘‘hasEdition’’/ >
<Class IRI=‘‘Book’’/ >
</DataPropertyDomain> <DataPropertyRange>
<DataProperty IRI="hasEdition"/ >
<Datatype IRI=‘‘http://www.w3.org/2001/
XMLSchema#nonNegativeInteger’’/ > </DataPropertyRange>
```

It is possible to express and define new datatypes by constraining or combining existing ones. Example 9 shows the use of cardinality restrictions on the `hasEdition` property. These kind of restrictions can be also used to construct facets.

Example 9.
```
<DatatypeDefinition> <Datatype IRI=‘‘bookEdition’’/ >
<DatatypeRestriction>
<Datatype IRI=‘‘&xsd;integer’’/ >
<FacetRestriction facet=‘‘&xsd;minInclusive’’>
<Literal datatypeIRI=‘‘&xsd;integer’’> 1 </Literal>
</FacetRestriction> <FacetRestriction facet=‘‘&xsd;maxInclusive’’>
<Literal datatypeIRI=‘‘&xsd;integer"> 9 </Literal>
</FacetRestriction> </DatatypeRestriction> </DatatypeDefinition>
```

Datatypes can be combined just like classes by complement, intersection and union by using the following OWL constructors: `ObjectComplementOf`, `ObjectIntersectionOf` and `ObjectUnionOf`.

5.3 Adding expressiveness to the ontology

Metadata properties are used to add document information and annotations. These properties do not actually contribute to the logical knowledge specified in the ontology, but provide additional information about ontology elements. Example 10 shows an annotation property that is formed by literals, this property has a brief description for the `Book` class.

Example 10. <AnnotationAssertion>
<AnnotationProperty IRI=‘‘&rdfs;comment’’/> <IRI> Book </IRI>
<Literal> Represents a resource identified by the Latex entry @book
</Literal> </AnnotationAssertion>

Table 2 shows other OWL properties to add expressiveness to the constructed ontology.

5.4 Populating the ontology

The ClassAssertion property is used to associate individuals to classes. Example 11 assigns the ‘‘Ontological engineering’’ individual to the Book class.

Example 11. <ClassAssertion> <Class IRI=‘‘Book’’/ > <NamedIndividual IRI=‘‘Ontological engineering’’/ > </ClassAssertion>

OWL does not follow the assumption that different names refer to different individuals (UNA) [9]. Thus, the DifferentIndividuals and SameIndividual predicates are needed to express that two individuals are different ones or the same ones.

5.5 Building a multilingual ontology

Equivalent classes are useful to construct multilingual classification schemes. In OWL 2, two classes are *equivalent* if they contain exactly the same individuals. Example 12 defines an equivalence relationship between the Software engineering and the ‘‘Ingeniería de Software’’ class.

Example 12. <EquivalentClasses> <Class IRI=‘‘Software engineering’’>
<Class IRI=‘‘Ingeniería de Software’’> </EquivalentClasses>

6 Results

Unlike librarians, non specialized users of digital libraries do not need to know how the classification schemes work, although they are often interested in the relationships between subject headings to explore collections.

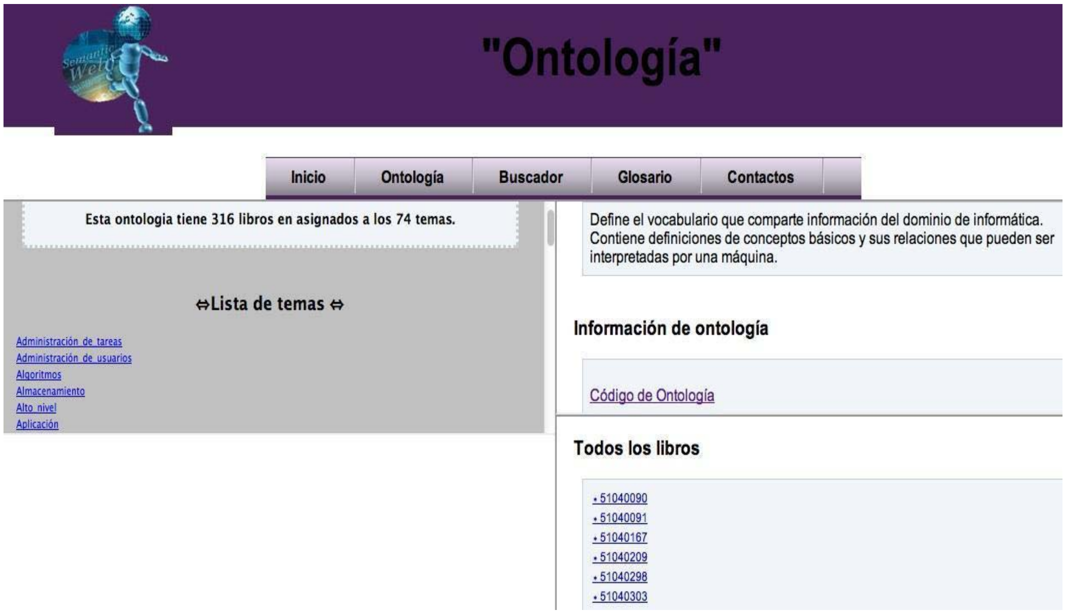
As a test bed for our approach, an excerpt of the 1998 ACM Computing Classification System is modeled as an OWL ontology. We use Protégé editor to create, browse and populate this ontology [11]. Protégé implements the OWL 2 QL profile[7]. OWL ontologies are placed into OWL documents, which are then placed into local file systems or on the World Wide Web. Protégé ontologies can be exported to different formats, such as OWL/XML or RDF/XML.

[7] The QL acronym reflects the fact that query answering in this profile can be implemented by rewriting queries into a standard relational query language

The constructed ontology is used to classify a book collection of Computer Science that belongs to the library of the Universidad Politécnica de Puebla (UPPuebla)[8]. At present, the ontology is formed by 14 classes at the first level, (11 of these classes correspond to the main categories of CSS, besides the sibling classes `Author`, `Editorial` and `Book`).

Each class has a set of individuals. Object properties, data type properties, qualification, cardinality, domain and range restrictions are also included. A set of 160 individuals of the Book class approximately have been classified. Only the CSS subcategories for each one of these individuals have been modeled. Fact++ and Pellet reasoners have been used to verify ontology consistency (a set of statements is consistent if there is a possible state in which all the statements in the set are jointly true [9]).

A web page has been designed to explore the constructed ontology. This page allows users to access the information of classes and make queries about properties, individuals and metadata. Book metadata are described with the unqualified Dublin Core (DC) elements [7]. Spanish values are used for DC elements. The use of standard metadata formats supports information sharing and improves interoperability between collections. The values of metadata can be included in different languages in order to improve collection accessibility.



"Ontología"

Esta ontologia tiene 316 libros en asignados a los 74 temas.

⇔Lista de temas ⇔

Administración de tareas
Administración de usuarios
Algoritmos
Almacenamiento
Alto nivel
Aplicación

Define el vocabulario que comparte información del dominio de informática. Contiene definiciones de conceptos básicos y sus relaciones que pueden ser interpretadas por una máquina.

Información de ontología

Código de Ontología

Todos los libros

- 51040090
- 51040091
- 51040167
- 51040209
- 51040298
- 51040303

Fig. 1. A web page to query the constructed ontology (Spanish version)

[8] An Spanish version of the ontology is available at http://informatica.uppuebla.edu.mx/ mmedina/ontologiaInformatica/Ontologia.php

From an information retrieval point of view, the main advantage of the modeling of classification schemes using ontologies is that users and applications can exploit the class-subclass relationships and the keywords simultaneously. Other advantages are the followings:

- Preservation of basic features of traditional classification schemes
- Expressiveness improvements due to properties, restrictions and annotations
- Consistency maintenance supported by reasoning services
- Importation of metadata standards and well defined name spaces
- Possibility to enhance search capabilities of applications
- Non exclusive membership of individuals
- Flexibility to define unnamed classes based on object and data properties
- Automatic validation and maintenance of consistency

7 Conclusions

This paper has presented the main steps to model a classification scheme like an ontology. The results outline the expressiveness of the ontology against the traditional classification schemes. Ontologies maintain consistently other relationships between classified documents besides topic relationships of the original classification schemes.

Ontologies enables the possibility to extend the search capabilities for applications by means of properties and restrictions. An excerpt of CSS has been used to construct an OWL 2 ontology. This ontology is constructed in Protégé editor and this is validated with Fact ++ and Pellet reasoner. A web page allow users to query this ontology.

As future work, we plan to estimate the improvements of our approach by means of information retrieval measures such as recall and precision in reference collections.

References

1. Aida, S.O.: Classification management and use in a networked environment: the case of the universal decimal classification. Ph.D. thesis, University of London (April 2005)
2. Allemang, D., Hendler, J.: Semantic web for the working ontologist. Morgan Kauffman Publishers (2011)
3. C., M.I.: The Universal Decimal Classification - a guide to its use. UDC Consortium (2007)
4. for Computing Machinery., A.: ACM Computing Classification System toc. http://www.acm.org/about/class/ (2012)
5. of congress., L.: Library of congress classification outline. http://www.loc.gov/catdir/cpso/lcco/ (2012)
6. Dewey, M., Mitchell, J.S., Beall, J., Green, R., Martin, G.: Dewey decimal classification and relative index. Forest Press. Dublin, Ohio. OCLC Online Computer Library Center, Inc. (2011)

7. DublinCore: Dublin core. 1997. dublin core metadata element set version 1.1: Reference. copyright 1995-2008 dcmi (1997), http://www.dublincore.org/documents/dces June 16th 2008
8. Gómez, A., Fernández, M., Corcho, O.: Ontological Engineering. Springer-Verlag, London, England (2004)
9. Hitzler, P., Krtzsch, M., Parsia, B., Patel-Schneider, P.F., Rudolph, S.: Owl 2 web ontology language primer (2009), http://www.w3.org/TR/owl-primer/
10. Isaac, A., Summers, E.: Skos simple knowledge organizationsystem primer. Knowledge Organization 1(February), 1–32 (2009), http://www.w3.org/TR/skos-primer/
11. Knublauch, H., Fergerson, R.W., Noy, N.F., Musen, M.A.: The proté.g.é owl plugin: an open development environment for semantic web applications. In: Heidelberg, S.V.B. (ed.) Handbook of Stuff I Care About, pp. 229–243. Sheila A. McIlraith and Dimitris Plexousakis and Frank van Harmelen (Eds). (2004), http://protege.stanford.edu/plugins/owl/publications/ISWC2004-protege-owl.pd
12. Kruk, S.R., McDaniel, B.: Semantic Digital Libraries. Springer-Verlag (2009)
13. Marcia Lei Zeng, Michael Panzer, A.S.: Expressing classification schemes with owl 2 web ontology language. In: 11th International Conference of the International Society for Knowledge Organization (ISKO). No. February in 1 (2010)
14. Osiriska Veslaba, B.P.: New methods for visualization and improvement of classification schemes: the case of computer science. Knowledge Organization 37(3) (2010), $http : //www.ergon - verlag.de/en/start.htm?d_{K}O_{P}rint_{v}ersion_{p}lus_{P}DF_{8}958.htm$
15. Smartlogic: Semaphore overview. a smart logic white paper. (2011), http://www.smartlogic.com/home/products/products-overview
16. Sybille Peters, C.P.R., Beuermann, W.S.: A new approach towards vertical search engines - intelligent focused crawling and multilingual semantic techniques. 6th International Conference on Web Information Systems, WEBIST 2010 (2010), $http : //www.eerqi.eu/sites/default/files/paper.pdf$
17. ibiblio. The Public's Library, Archive, D.: Library of congress classification outline. University of North Carolina at Chapel Hill. Collaboration of the School of Information and Library Science, School of Journalism and Mass Communication and Information Technology Services. (2012), http://www.ibiblio.org/librariesfaq/sect5.htm
18. Wielinga, B.J., Schreiber, A.T., Wielemaker, J., Sandberg, J.A.C.: From thesaurus to ontology. In: Proceedings of the First International Conference on Knowledge Capture. pp. 194–201. K-CAP '01, ACM, New York, NY, USA (2001), http://doi.acm.org/10.1145/500737.500767

Table 1. An excerpt of the CSS classification scheme

Main categories	**Number and examples of subcategories (level 2)**	**Examples of categories (level 3)**
A. General literature	4 **A.0** *General*	Biographies / autobiographies Conference proceedings General literary works Proper nouns
B. Hardware	9 **B.0** *General* **B.2** Arithmetic and logic structures	
C. Computer Systems Organization	6 **C.3** Special-purpose and application-based systems	
D. Software	5 **D.0** *General* **D.1** Programming techniques **D.2** Software engineering **D.3** Programming languages **D.4** Operating systems **D.5** *Miscellaneous*	**D.3.0** General **D.3.1** Formal definitions theory **D.3.2** Language classifications **D.3.3** Language constructs and features **D.3.4** Processors **D.3.m** *Miscellaneous*
E. Data	6	
F. Theory of Computation	6	
G. Mathematics of Computing	5	
H. Information Systems	6	
I. Computing Methodologies	8	
J. Computer Applications	8	
K. Computing Milieux	9	

Table 2. Other OWL properties to model classification schemes with ontologies

Property	Description
`AsymmetricObjectProperty`	Asymmetry of a property
`DisjointObjectProperties`	For disjoint properties
`FunctionalObjectProperty`	Functional property
`HasKey`	Assign a key to a class expression
`InverseFunctionalObjectProperty`	The inverse of a functional property is functional
`InverseObjectProperties`	The inverse of a property
`IrreflexiveObjectProperty`	Irreflexivity of a property
`ObjectExactCardinality`	For an exact value in a cardinality restriction
`ObjectHasValue`	For classes of individuals related to one particular individual
`ObjectOneOf`	For numerated (closed) classes
`ReflexiveObjectProperty`	Reflexivity of a property
`SymmetricObjectProperty`	Symmetry of a property
`TransitiveObjectProperty`	Transitivity of a property