Results of Taxonomic Evaluation of RDF(S) and DAML+OIL Ontologies using RDF(S) and DAML+OIL Validation Tools and Ontology Platforms Import Services

Asunción Gómez-Pérez, M. Carmen Suárez-Figueroa

Laboratorio de Inteligencia Artificial
Facultad de Informática
Universidad Politécnica de Madrid
Campus de Montegancedo sn.
Boadilla del Monte, 28660. Madrid, Spain
asun@fi.upm.es
mcsuarez@delicias.dia.fi.upm.es

Abstract. Before using RDF(S) and DAML+OIL ontologies in Semantic Web applications, its content should be evaluated from a knowledge representation point of view. In recent years, some RDF(S) and DAML+OIL 'checkers', 'validators', and 'parsers' have been created and several ontology platforms are able to import RDF(S) and DAML+OIL ontologies. Two are the experiments presented in this paper. The first one reveals that the majority of RDF(S) and DAML+OIL parsers (Validating RDF Parser, RDF Validation Service, DAML Validator, and DAML+OIL Ontology Checker) do not detect taxonomic mistakes in ontologies implemented in such languages. So, if such ontologies are imported by ontology platforms, are they able to detect such problems? The second experiment presented in this paper reveals that the majority of the ontology platforms (OilEd, OntoEdit, Protégé-2000, and WebODE) only detect a few of mistakes in concept taxonomies before importing them.

1 Introduction

In recent years, considerable progress has been made in developing the conceptual bases for building technology that allows reusing and sharing ontologies for the Semantic Web. As any other resource used in software applications, ontology content should be evaluated before (re)using it in other ontologies or applications. In that sense, we could say that it is unwise to publish an ontology or to implement software that relies on ontologies written by others (even by yourself) without first evaluating its content, that is, its concept definitions, its taxonomy and its formal axioms.

Ontology evaluation is an important activity to be carried out during the whole ontology life-cycle. Up to now, few domain-independent methodological approaches [6, 11, 15, 17] include an evaluation activity.

The first works on ontology content evaluation started in 1994 [9, 10], and in the last three years the interest of the Ontological Engineering community in this issue has grown. The main efforts were made by Gómez-Pérez [7, 8] and by Guarino and

colleagues with the OntoClean method [12]. ODEClean [5] is a tool integrated into the WebODE environment that gives support to the OntoClean method.

With the increasing number of ontologies implemented in the ontology markup languages RDF(S) [3, 13] and DAML+OIL [18], many specialized ontology validation tools for these languages have been built: Validating RDF Parser[1], RDF Validation Service[2], DAML Validator[3], DAML+OIL Ontology Checker[4], etc. These tools are mainly focused on evaluating ontologies from a syntactic point of view, that is, checking whether the ontologies are compliant with the languages specification. However, they are not focused on detecting mistakes from a knowledge representation point of view, that is, if the ontologies have inconsistencies and redundancies.

We have performed experiments with 24 ontologies (7 on RDF(S) and 17 on DAML+OIL), which are well built from a syntactic point of view, according to the languages specifications, but have inconsistencies and redundancies. We have parsed them with the previous four tools and we have discovered that on the majority of the experiments, they do not detect the taxonomic mistakes identified in [7].

The key point is that RDF(S) and DAML+OIL ontologies are imported by ontology platforms. In fact, OilEd [2], OntoEdit [16], Protégé-2000 [14], and WebODE [4, 1] are able to import ontologies implemented in both languages, but there are not previous works analysing whether such platforms are able to detect wrong RDF(S) and DAML+OIL ontologies. In order to carry out this analysis, we have used the same 24 ontologies (7 on RDF(S) and 17 on DAML+OIL) and we have imported them within the previous ontology platforms. We have found out that on the majority of the experiments, these ontology platforms do not detect mistakes in concept taxonomies represented in RDF(S) and DAML+OIL.

This paper is organized as follows, section two presents briefly the method for evaluating taxonomic knowledge in ontologies. Section three presents a description of some ontology 'checkers', 'validators', and 'parsers'. Section four includes our first comparative study, including examples of the RDF(S) and DAML+OIL ontologies used on the testbed. Section five presents an overview of some ontology platforms. Section six presents the results of importing RDF(S) and DAML+OIL ontologies with taxonomic mistakes in the ontology platforms. Finally, we conclude with further work on evaluation.

2 Method for Evaluating Taxonomic Knowledge in Ontologies

Figure 1 presents a set of possible mistakes that can be made by ontologists when modeling taxonomic knowledge in an ontology under a frame-based approach [7]. In this paper we only focus on inconsistency mistakes (circularity and partition) and redundancy mistakes (grammatical), and we postpone the analysis of the others for further works. Below we explain briefly the studied mistakes.

[1] http://139.91.183.30:9090/RDF/VRP/
[2] http://www.w3.org/RDF/Validator/
[3] http://www.daml.org/validator/
[4] http://potato.cs.man.ac.uk/oil/Checker

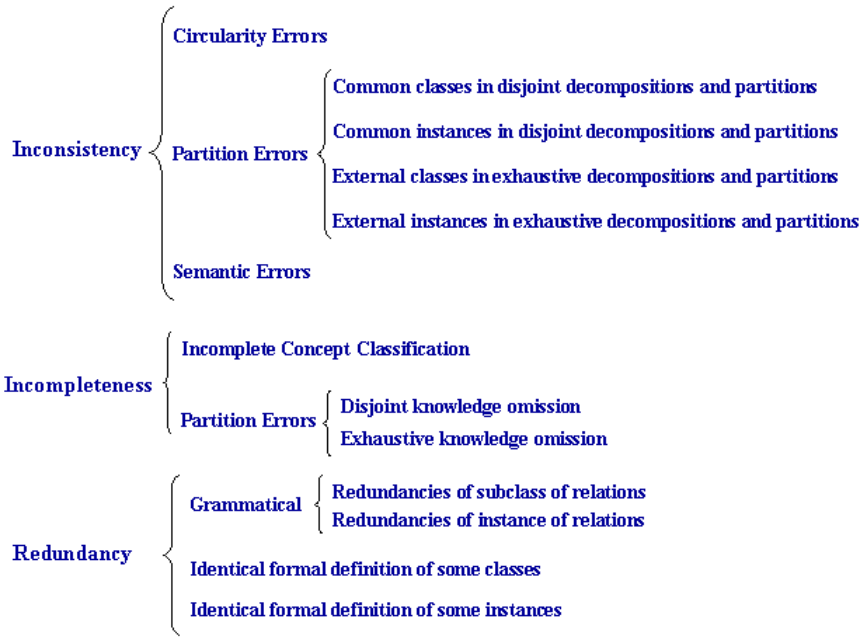


Figure 1. Types of mistakes that might be made when developing taxonomies with frames

Inconsistency: Circularity Errors occur when a class is defined as a specialization or generalization of itself. Depending on the number of relations involved, circularity errors can be classified as circularity errors at distance zero (a class with itself), circularity errors at distance 1, and circularity errors at distance n.

Inconsistency: Partition errors. Concept classifications can be defined in a disjoint (disjoint decompositions), a complete (exhaustive decompositions), and a disjoint and complete manner (partitions). The following types of partition errors are identified:

- *Common classes in disjoint decompositions and partitions*. These occur when there is a disjoint decomposition or a partition *class-*p_1,…, *class-*p_n defined in a class *class-A*, and one or more classes *class-*B_1,..., *class-*B_k are subclasses of more than one *class-*p_i.
- *Common instances in disjoint decompositions and partitions*. These errors happen when one or several instances belong to more than one class of a disjoint decomposition or partition.
- *External classes in exhaustive decompositions and partitions*. They occur when having defined an exhaustive decomposition or a partition of the base class (*class-A*) into the set of classes *class-*p_1,..., *class-*p_n, and there are one or more classes that are subclasses of the *class-A*, instead of being subclasses of a class the set of classes *class-*p_1,..., *class-*p_n.
- *External instances in exhaustive decompositions and partitions*. These errors occur when we have defined an exhaustive decomposition or a partition of the base class (*class-A*) into the set of classes *class-*p_1,..., *class-*p_n, and there are one

or more instances of the *class-A* that do not belong to any class *class-*p_i of the exhaustive decomposition or partition.

Redundancy: Grammatical Errors.

- *Redundancies of 'subclass-of' relations* occur between classes they have more than one 'subclass-of' relation. We can distinguish direct and indirect repetition.
- *Redundancies of 'instance-of' relations*. As in the above case, we can distinguish between direct and indirect repetition.

3 Ontology 'Checkers', 'Validators' and 'Parsers'

At the moment, there exist various ontology 'checkers', 'validators', and 'parsers' which are intended to carry out some kind of validation and/or checking of ontologies on diverse web-based languages. In this paper, we focus on the most frequently used parsers that validate and/or check ontologies on RDF(S) and DAML+OIL: Validating RDF Parser and RDF Validation Service for RDF(S), and DAML Validator and DAML+OIL Ontology Checker for DAML+OIL. Other parsers not included in this paper are: Rapier RDF Parser[5], Thea RDF Parser[6], Chimaera[7], ConsVISor[8], etc.

The Validating RDF Parser. The ICS-FORTH RDFSuite[9] is a suite of tools for RDF metadata management. This RDFSuite consists of tools for parsing, validating, storing and querying RDF descriptions, namely the Validating RDF Parser (VRP), the RDF Schema Specific DataBase (RSSDB) and the RDF Query Language (RQL). The ICS-FORTH Validating RDF Parser (VRP v2.5)[10] analyzes, validates and processes RDF schemas and resource descriptions. This parser offers the following functions:

- *Syntactic Validation* for checking if the RDF/XML syntax of the input namespace conforms to the updated RDF/XML syntax proposed by W3C.
- *Semantic Validation* for verifying the selected constraints derived from RDF Schema Specification (RDFS). VRP allows to choose several semantic validation constraints: class hierarchy loops, property hierarchy loops, domain and range of subproperties, source and target resources of properties, and types of resources.

RDF Validation Service. The W3C RDF Validation Service[11] is based on HP-Labs Another RDF Parser (ARP[12]), which currenlty uses the version 2-alpha-1. This online service supports the Last Call Working Draft specifications issued by the RDF Core Working Group, including datatypes. This online service offers the following functions:

- *Syntactic Validation* for checking if the input namespace conforms to the updated RDF/XML Syntax Specification proposed by W3C.

---

[5] http://www.redland.opensource.ac.uk/raptor/
[6] http://www.semanticweb.gr/
[7] http://www.ksl.stanford.edu/software/chimaera/
[8] http://vis.home.mindspring.com/index.html
[9] Partially supported by EU projects C-Web (IST-1999-13479), MesMuses (IST-2001- 26074), and QUESTION-HOW (IST-2000-28767)
[10] http://139.91.183.30:9090/RDF/VRP/index.html
[11] http://www.w3.org/RDF/Validator/
[12] ARP was created and is maintained by Jeremy Carroll at HP-Labs in Bristol

- *Semantic Validation*. The service does not do any RDF Schema Specification validation.

DAML Validator. The DAML Validator[13] is available via either a WWW interface or download. The Validator uses the ARP parser from the Jena (1.6.1) toolkit to create an RDF triple model from the input code being validated. The DAML Validator checks DAML+OIL markup for problems beyond simple syntax errors. The Validator reads in a DAML file and examines it for a variety of potential errors. The output is a list of indications (errors, warnings, or information), a pointer to the errors in the file, and some guidance on the nature of the problems. It offers the following functions:

- *Syntactic Validation* for checking for namespace problems (outdated URIs, file extensions in URIs) during model creation. The validator tests RDF resources for existence: any subject, or object resource that is referenced must have a defined type.
- *Semantic Validation* for verifying the global domain and range constraints of the predicate. The subject and object of a statement should be instances of the predicate's domain and range classes. Each node (RDF Resource and it's accompanying statements) is validated based on the following types: Class, Property, Restriction, ObjectRestriction, DatatypeRestriction, or an Instance of one or more classes.

DAML+OIL Ontology Checker. The DAML+OIL Checker[14] was developed by University of Manchester (UK). The DAML+OIL Checker is a servlet that uses the OilEd codebase to check the syntax of DAML+OIL ontologies and returns a report on the classes and properties in the model. This checker is a web interface to check DAML+OIL ontologies and content using Jena. It offers the following functions:

- *Syntactic Validation* for checking missing definitions. The checker is fairly strict about the format of the input: in particular "rdf:ID attributes" must be conforming XML names, and unqualified attributes should not be used.
- *Semantic Validation* for verifying class hierarchy loops.

4 Comparative Study of RDF(S) and DAML+OIL 'Checkers', 'Validators' and 'Parsers'

As we said before, the first goal of this paper is to analyse whether RDF(S) and DAML+OIL parsers presented in section 3 detect the concept taxonomy mistakes presented in section 2. In order to achieve this goal, we have built a testbed of 24 ontologies (7 in RDF(S) and 17 in DAML+OIL), each of which implements one of the errors presented in section 2. And we have parsed them with the previous parsers. In the case of RDF(S) we have only 7 ontologies because partitions cannot be defined in this language.

These ontologies and the results of their evaluation can be found at http://minsky.dia.fi.upm.es/odeval/index.html.

[13] http://www.daml.org/validator/
[14] http://potato.cs.man.ac.uk/oil/Checker

In figure 2 we show the RDF(S) code and graphical notation of two of these ontologies: the one that implements the circularity error at distance 2, and the one that implements the mistake of indirect redundancy of 'instance-of' relation. Figure 3 shows the DAML+OIL code and graphical notation of three of these ontologies: the one that implements the circularity error at distance 1, the one that implements the mistake of common class in disjoint decomposition, and the last one that implements the mistake of external instance in partition.

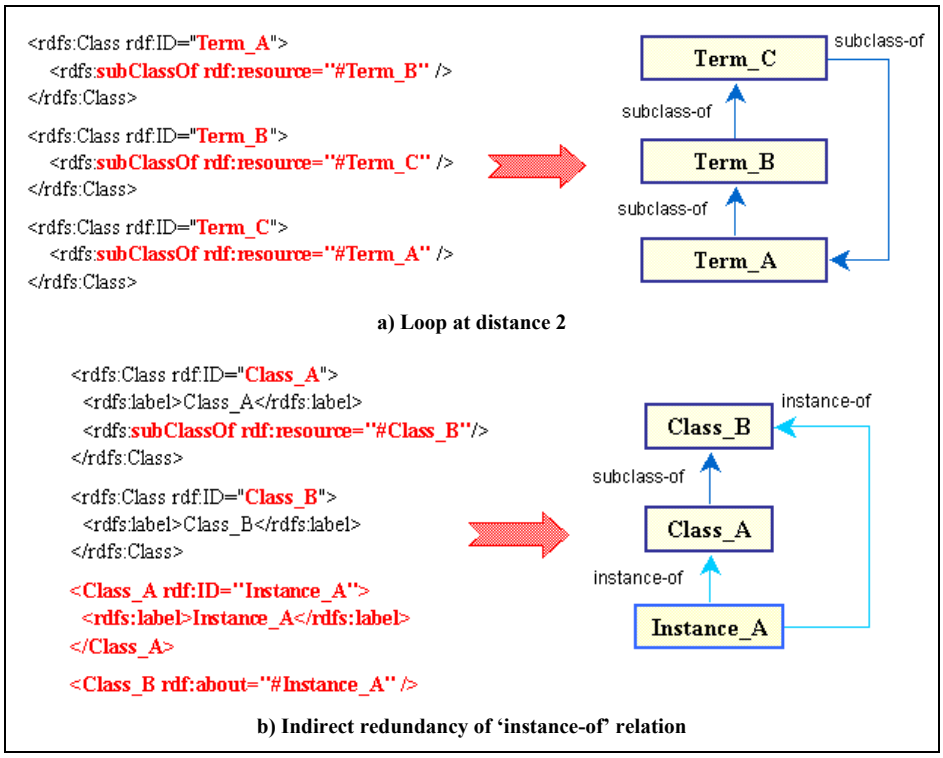


Figure 2. Examples of RDF(S) ontologies

After parsing the ontologies on the testbed with the parsers, we found that all these parsers recognised the code as well formed code, but the majority had problems detecting most of the knowledge representation mistakes that these ontologies contained.

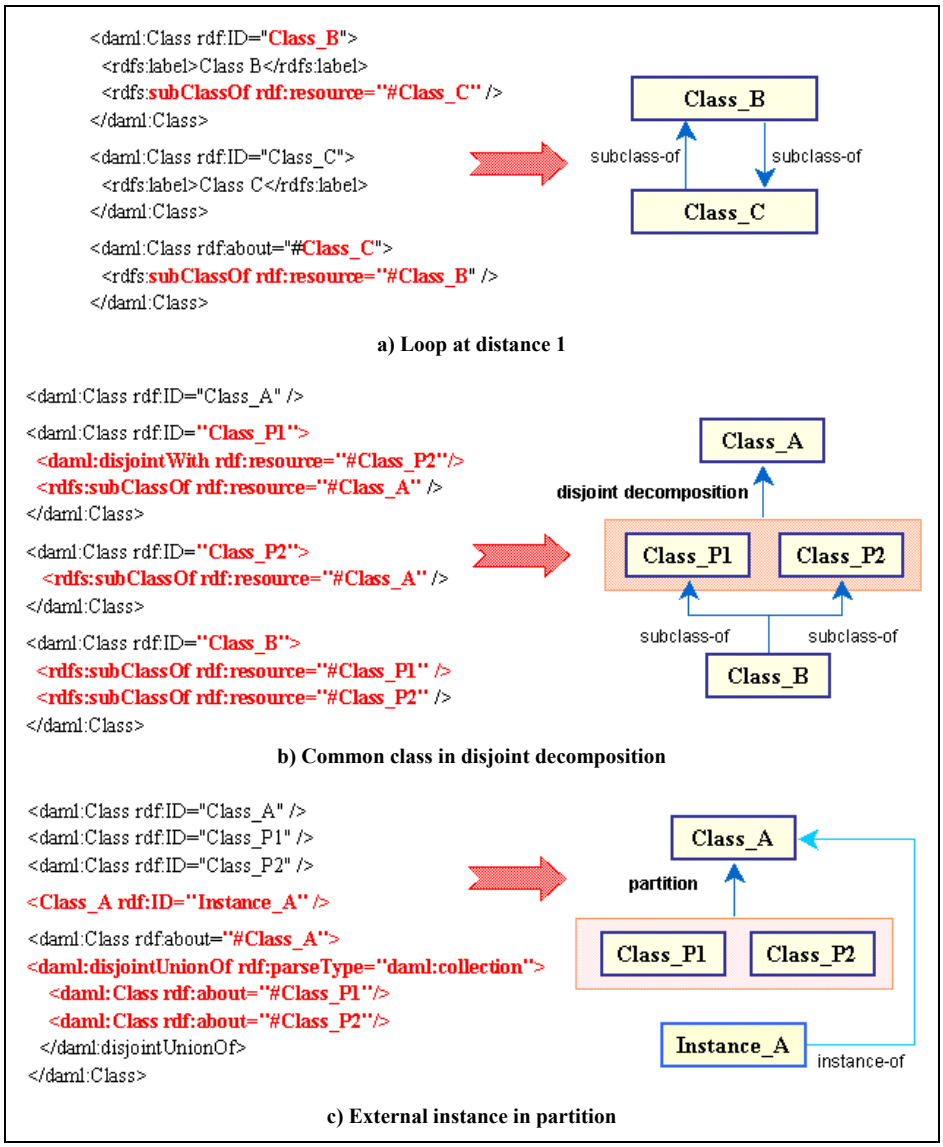
The results of analysing and comparing these parsers are shown in table 1. The symbols used in this table are the following:

⦸: The parser does not accept files written in this language

✓: The parser detects the mistake in this language

×: The parser does not detect the mistake in this language

--: The mistake cannot be represented in this language




```
<daml:Class rdf:ID="Class_B">
  <rdfs:label>Class B</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Class_C" />
</daml:Class>
<daml:Class rdf:ID="Class_C">
  <rdfs:label>Class C</rdfs:label>
</daml:Class>
<daml:Class rdf:about="#Class_C">
  <rdfs:subClassOf rdf:resource="#Class_B" />
</daml:Class>
```

```
<daml:Class rdf:ID="Class_A" />
<daml:Class rdf:ID="Class_P1">
  <daml:disjointWith rdf:resource="#Class_P2"/>
  <rdfs:subClassOf rdf:resource="#Class_A" />
</daml:Class>
<daml:Class rdf:ID="Class_P2">
  <rdfs:subClassOf rdf:resource="#Class_A" />
</daml:Class>
<daml:Class rdf:ID="Class_B">
  <rdfs:subClassOf rdf:resource="#Class_P1" />
  <rdfs:subClassOf rdf:resource="#Class_P2" />
</daml:Class>
```

```
<daml:Class rdf:ID="Class_A" />
<daml:Class rdf:ID="Class_P1" />
<daml:Class rdf:ID="Class_P2" />
<Class_A rdf:ID="Instance_A" />
<daml:Class rdf:about="#Class_A">
<daml:disjointUnionOf rdf:parseType="daml:collection">
    <daml:Class rdf:about="#Class_P1"/>
    <daml:Class rdf:about="#Class_P2"/>
  </daml:disjointUnionOf>
</daml:Class>
```

Figure 3. Examples of DAML+OIL ontologies

As we can see in table 1, we have checked whether RDF(S) tools (VRP and RDF Validation Service) were able to evaluate DAML+OIL files, and whether DAML+OIL tools (DAML Validator and DAML+OIL Ontology Checker) were able to evaluate RDF(S) files. In the case of RDF(S) tools, the experiments showed that RDF Validation Service can read DAML+OIL ontologies, although it does not detect the mistakes, but VRP cannot read them. In the case of DAML+OIL tools, the experiments showed that both of them are able to recognize RDF(S) files. Although

the DAML+OIL Ontology Checker is not a RDF(S) validation tool, it was able to detect circularity errors in that language.

Before going in detail with circularity errors, we have an important comment to make. The RDF(S) and DAML+OIL specifications allow cycles in concept taxonomies. However, we consider that this is a mistake from the knowledge representation point of view, that is, we would not recommend designing ontologies with cycles in their concept taxonomies. So here we want to stress the distinction between checking an ontology from a syntactic point of view (checking whether the ontology is compliant with the language specification) and checking an ontology from a knowledge representation point of view (checking whether the ontology does not have the mistakes presented in section 2).

Circularity errors are the only ones detected by some of the parsers studied in this experiment. VRP is able to detect circularity errors at any distance in RDF(S) ontologies, indicating that there is a semantic error ("loop detected"). The DAML+OIL Ontology Checker detects circularity errors at any distance in RDF(S) and DAML+OIL ontologies, throwing a warning about it ("cycles in class hierarchy").

Regarding *partition errors*, they have only been studied for DAML+OIL, since they cannot be represented in RDF(S). None of the DAML+OIL validators, neither the RDF Validation Service, have detected partition errors with the 10 ontologies from the testbed.

The same occurs with the *grammatical redundancy errors*, which are not detected by any of the RDF(S) and DAML+OIL parsers studied.

5 Ontology Platforms

In this paper we focus on the most representative ontology platforms that can be used for importing ontologies: OilEd, OntoEdit, Protégé-2000, and WebODE. In this section, we provide a broad overview of these ontology platforms.

OilEd[15] [2] was initially developed as an ontology editor for OIL ontologies, in the context of the European IST OntoKnowledge project. However, OilEd has evolved and now is an editor of DAML+OIL and OWL ontologies. OilEd can import ontologies implemented in RDF(S), OIL, DAML+OIL, and the SHIQ XML format. Besides exporting ontologies to DAML+OIL, OilEd ontologies can be exported to the RDF(S) and OWL ontology languages and to the XML formats SHIQ and DIG.

OntoEdit[16] [16] has been developed by AIFB in Karlsruhe University. It is an extensible and flexible environment, based on a plugin architecture, which provides functionality to browse and edit ontologies. It includes plugins for reasoning using Ontobroker, plugins for exporting and importing ontologies in different formats (FLogic, OXML, RDF(S), DAML+OIL), etc. Two versions of OntoEdit are available: OntoEdit Free and OntoEdit Professional.

[15] http://oiled.man.ac.uk
[16] http://www.ontoprise.de/com/start_downlo.htm

			ICS-FORTH Validating RDF Parser		RDF Validation Service		DAML Validator		DAML+OIL Ontology Checker	
			RDF(S)	DAML+OIL	RDF(S)	DAML+OIL	RDF(S)	DAML+OIL	RDF(S)	DAML+OIL
Inconsistency: Circularity Errors	At distance zero		✓	⊘	×	×	×	×	✓	✓
	At distance one		✓	⊘	×	×	×	×	✓	✓
	At distance n		✓	⊘	×	×	×	×	✓	✓
Inconsistency: Partition Errors	Common classes in disjoint decompositions	Direct	--	⊘	--	×	--	×	--	×
		Indirect	--	⊘	--	×	--	×	--	×
	Common classes in partitions		--	⊘	--	×	--	×	--	×
	Common instances in disjoint decompositions	Direct	--	⊘	--	×	--	×	--	×
		Indirect	--	⊘		×	--	×	--	×
	Common instances in partitions		--	⊘	--	×	--	×	--	×
	External classes in exhaustive decompositions		--	⊘	--	×	--	×	--	×
	External classes in partitions		--	⊘	--	×	--	×	--	×
	External instances in exhaustive decompositions		--	⊘	--	×	--	×	--	×
	External instances in partitions		--	⊘	--	×	--	×	--	×
Redundancy: Grammatical Errros	Redundancies of 'subclass-of' relations	Direct	×	⊘	×	×	×	×	×	×
		Indirect	×	⊘	×	×	×	×	×	×
	Redundancies of 'instance-of' relations	Direct	×	⊘	×	×	×	×	×	×
		Indirect	×	⊘	×	×	×	×	×	×

Table 1. Results of the analysis of the RDF(S) and DAML+OIL parsers

Protégé-2000[17] [14] has been developed by the Stanford Medical Informatics (SMI) at Stanford University, and is the latest version of the Protégé line of tools. It is an open source, standalone application with an extensible architecture. The core of this environment is the ontology editor, and it holds a library of plugins that add more functionality to the environment (ontology language importation and exportation, OKBC access, constraints creation and execution, etc.). Protégé-2000 ontologies can be exported and imported with some of the backends provided in the standard release or as plugins: RDF(S), DAML+OIL, OWL, XML, XML Schema, and XMI.

WebODE[18] [4, 1] has been developed by the Ontology Engineering Group at Universidad Politécnica de Madrid (UPM). It is an ontology-engineering suite created with an extensible architecture. WebODE is not used as a standalone application, but as a Web application. There are several services for ontology language import and export (XML, RDF(S), DAML+OIL, OIL, OWL, CARIN, FLogic, Jess, Prolog), axiom edition with WAB (WebODE Axiom Builder), ontology documentation, ontology evaluation, and ontology merge.

6 Comparative Study of Ontology Platforms Import Services

As we said before, the second main goal of this paper is to analyse whether ontology platforms presented in section 5, are able to detect taxonomic mistakes in RDF(S) and DAML+OIL ontologies before importing them.

In order to carry out this experiment, we have reused the same 24 ontologies (7 in RDF(S) and 17 in DAML+OIL with inconsistency and redundancy mistakes) used in the previous experiment. In the case of RDF(S) we have only 7 ontologies because partitions cannot be defined in this language. We have imported these ontologies using the import facilities of the ontology platforms presented in section 5. Table 2 presents the results of the experiment using the following symbols:

- 😐 : The ontology platform does not allow representing this type of mistake
- ✓ : The ontology platform detects the mistake during ontology import
- × : The ontology platform does not detect the mistake during ontology import
- -- : The mistake cannot be represented in this language

The main conclusions of the RDF(S) and DAML+OIL ontology import are:

Circularity errors at any distance are the only ones detected by most of ontology platforms analyzed in this experiment. However, OntoEdit Free does not detect circularity errors at distance zero, but it ignores them.

Regarding *partition errors*, we have only studied DAML+OIL ontologies because this type of knowledge cannot be represented in RDF(S). Most of ontology platforms used in this study do not detect partition errors in DAML+OIL ontologies. Furthermore, some partition errors (common instance in partitions, external instance

[17] http://protege.stanford.edu/plugins.html
[18] http://webode.dia.fi.upm.es/

in exhaustive decompositions, etc.) cannot be represented in the ontology platforms studied. Only WebODE detects some partition errors using the ODEval[19] service.

Grammatical redundancy errors are not detected by most of ontology platforms used in this work. However some ontology platforms ignore direct redundancies of 'subclass-of' or 'instance-of' relations. As the previous case, only WebODE detects indirect redundancies of 'subclass-of' relations in RDF(S) and DAML+OIL ontologies using the ODEval service.

7 Conclusions and Further Work

In this paper we have shown that, in general, current RDF(S) and DAML+OIL 'checkers', 'validators', and 'parsers' are not able to detect mistakes from a knowledge representation point of view, but they mainly focus on the syntactic validation of the RDF(S) and DAML+OIL ontologies that they parser.
We have also shown that only a few taxonomic mistakes in RDF(S) and DAML+OIL ontologies are detected by ontology platforms which are able to import ontologies in such languages.

Taking into account that only a few parsers are able to detect loops in RDF(S) and DAML+OIL taxonomies, we considered that it is necessary to create more advanced evaluators than those already existing for evaluating RDF(S) and DAML+OIL from a knowledge representation point of view.

We also consider that it is necessary to create more advanced ontology import services in ontology platforms.

We think that much work must be made to integrate ontology evaluation functions in ontology development tools, and to create an integrated ontology evaluation tool suite that will permit analyzing ontologies in different languages and KR formalisms.

Acknowledgements

This work has been supported by the Esperonto project (IST-2001-34373), by the Spanish project 'Plataforma Tecnológica para la web semántica: Ontologías, análisis de lenguaje natural y comercio electrónico' (TIC-2001-2745), and by a research grant from UPM ("Beca asociada a proyectos modalidad B").

We thanks for his comments and revisions to Óscar Corcho.

[19] http://minsky.dia.fi.upm.es/odeval

			OilEd		OntoEdit Free		Protégé-2000		WebODE	
			RDF(S)	DAML+OIL	RDF(S)	DAML+OIL	RDF(S)	DAML+OIL	RDF(S)	DAML+OIL
Inconsistency: Circularity Errors	At distance zero		✓	✓	×	×	✓	✓	✓	✓
	At distance one		✓	✓	✓	✓	✓	✓	✓	✓
	At distance n		✓	✓	✓	✓	✓	✓	✓	✓
Inconsistency: Partition Errors	Common classes in disjoint decompositions	Direct	--	×	--	×	--	×	--	✓
		Indirect	--	×	--	×	--	×	--	✓
	Common classes in partitions		--	×	--	😐	--	×	--	✓
	Common instances in disjoint decompositions	Direct	--	×	--	×	--	😐	--	😐
		Indirect	--	×	--	×	--	😐	--	😐
	Common instances in partitions		--	×	--	😐	--	😐	--	😐
	External classes in exhaustive decompositions		--	×	--	😐	--	×	--	😐
	External classes in partitions		--	×	--	😐	--	×	--	✓
	External instances in exhaustive decompositions		--	×	--	😐	--	×	--	😐
	External instances in partitions		--	×	--	😐	--	×	--	✓
Redundancy: Grammatical Errors	Redundancies of subclass-of relations	Direct	×	×	×	×	×	×	×	×
		Indirect	×	×	×	×	×	×	✓	✓
	Redundancies of instance-of relations	Direct	×	×	×	×	×	😐	😐	😐
		Indirect	×	×	×	×	😐	😐	😐	😐

Table 2. Results of the RDF(S) and DAML+OIL ontology import

References

1. Arpírez JC, Corcho O, Fernández-López M, Gómez-Pérez A (2003) *WebODE in a nutshell*. AI Magazine To be published in 2003
2. Bechhofer S, Horrocks I, Goble C, Stevens R (2001) *OilEd: a reason-able ontology editor for the Semantic Web*. In: Baader F, Brewka G, Eiter T (eds) Joint German/Austrian conference on Artificial Intelligence (KI'01). Vienna, Austria. (Lecture Notes in Artificial Intelligence LNAI 2174) Springer-Verlag, Berlin, Germany, pp 396–408
3. Brickley D, Guha RV (2003) *RDF Vocabulary Description Language 1.0: RDF Schema*. W3C Working Draft. http://www.w3.org/TR/PR-rdf-schema
4. Corcho O, Fernández-López M, Gómez-Pérez A, Vicente O (2002) *WebODE: an Integrated Workbench for Ontology Representation, Reasoning and Exchange*. In: Gómez-Pérez A, Benjamins VR (eds) 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW'02). Sigüenza, Spain. (Lecture Notes in Artificial Intelligence LNAI 2473) Springer-Verlag, Berlin, Germany, pp 138–153
5. Fernández-López M, Gómez-Pérez A (2002) *The Integration of OntoClean in WebODE*. In: Angele J, Sure Y (eds) EKAW02 Workshop on Evaluation of Ontology-based Tools (EON2002), Sigüenza, Spain, pp 38-52.
6. Fernández-López M, Gómez-Pérez A, Pazos-Sierra A, Pazos-Sierra J (1999) *Building a Chemical Ontology Using METHONTOLOGY and the Ontology Design Environment*. IEEE Intelligent Systems & their applications 4(1) (1999) 37-46.
7. Gómez-Pérez A (2001) *Evaluating ontologies: Cases of Study. IEEE Intelligent Systems and their Applications*. Special Issue on Verification and Validation of ontologies. Marzo 2001, Vol 16, Nº 3. Pag. 391 – 409.
8. Gómez-Pérez A (1996) *A Framework to Verify Knowledge Sharing Technology*. Expert Systems with Application. Vol. 11, N. 4. PP: 519-529.
9. Gómez-Pérez A (1994) *Some ideas and Examples to Evaluate Ontologies*. Technical Report KSL-94-65. Knowledge System Laboratory. Stanford University. Also in Proceedings of the 11th Conference on Artificial Intelligence for Applications. CAIA94.
10. Gómez-Pérez A (1994) *From Knowledge Based Systems to Knowledge Sharing Technology: Evaluation and Assessment*. Technical Report. KSL-94-73. Knowledge Systems Laboratory. Stanford University. December.
11. Grüninger M, Fox MS (1995) *Methodology for the design and evaluation of ontologies*. In Workshop on Basic Ontological Issues in Knowledge Sharing (Montreal, 1995).
12. Guarino N, Welty C (2000) *A Formal Ontology of Properties* In R. Dieng and O. Corby (eds.), Knowledge Engineering and Knowledge Management: Methods, Models and Tools. 12th International Conference, EKAW2000, LNAI 1937. Springer Verlag: 97-112. 2000.
13. Lassila O, Swick R (1999) *Resource Description Framework (RDF) Model and Syntax Specification*. W3C Recommendation. http://www.w3.org/TR/REC-rdf-syntax/
14. Noy NF, Fergerson RW, Musen MA (2000) *The knowledge model of Protege-2000: Combining interoperability and flexibility*. In: Dieng R, Corby O (eds) 12th International Conference in Knowledge Engineering and Knowledge Management (EKAW'00). Juan-Les-Pins, France. (Lecture Notes in Artificial Intelligence LNAI 1937) Springer-Verlag, Berlin, Germany, pp 17–32
15. Staab S, Schnurr HP, Studer R, Sure Y (2001) *Knowledge Processes and Ontologies*, IEEE Intelligent Systems, 16(1). 2001.
16. Sure Y, Erdmann M, Angele J, Staab S, Studer R, Wenke D (2002) *OntoEdit: Collaborative Ontology Engineering for the Semantic Web*. In: Horrocks I, Hendler JA (eds) First International Semantic Web Conference (ISWC'02). Sardinia, Italy. (Lecture Notes in Computer Science LNCS 2342) Springer-Verlag, Berlin, Germany, pp 221–235

17. Uschold M, Grüninger M (1996) *ONTOLOGIES: Principles, Methods and Applications*. Knowledge Engineering Review. Vol. 11; N. 2; June 1996.
18. van Harmelen F, Patel-Schneider PF, Horrocks I (2001) *Annotated DAML+OIL (March 2001) Markup Language*. Technical Report. http://www.daml.org/2001/03/daml+oil-walkthru.html