Defining a Benchmark Suite for Evaluating the Import of OWL Lite Ontologies

Stefano David, Raúl García-Castro, Asunción Gómez-Pérez

Ontology Engineering Group, Departamento de Inteligencia Artificial.
Facultad de Informática, Universidad Politécnica de Madrid, Spain
sdavid@delicias.dia.fi.upm.es,{rgarcia,asun}@fi.upm.es

Abstract. Semantic Web tools should be able to correctly interchange ontologies and, therefore, to interoperate. This interchange is not always a straightforward task if tools have different underlying knowledge representation paradigms. This paper describes the process followed to define a benchmark suite for evaluating the OWL import capabilities of ontology development tools in a benchmarking activity in progress in the Knowledge Web[1] European Network of Excellence.

1 Introduction

The spreading interest in Semantic Web technologies is leading to the development of an increasing number of tools, each providing a different set of functionalities: ontology development tools, ontology repositories, ontology alignment tools, ontology-based annotators, etc. As ontologies are widely used in the Semantic Web for representing knowledge, these tools should be able to correctly interchange ontologies between themselves and, therefore, to interoperate.

OWL is the language recommended by the World Wide Web Consortium for defining and instantiating ontologies [1] and it currently seems the right choice to use as an language for interchanging them.

Nevertheless, this interchange may not be as straightforward as it seems. Interoperability between tools with different underlying knowledge models using an interchange language requires that these tools are able of translating ontologies from their own knowledge model to the interchange language and vice versa.

This kind of interoperability is being assessed in the Knowledge Web European Network of Excellence by benchmarking the interoperability of ontology development tools using OWL as interchange language; as these tools are a clear example of the use of different underlying knowledge models and require frequent ontology interchanges when developing ontologies collaboratively.

This paper describes the process followed for the definition of one of the benchmark suites that is being used in this benchmarking activity for evaluating the OWL import capabilities of ontology development tools.

This benchmark suite considers the different combinations of ontology components that can be found in ontologies as well as other factors that may influence the interoperability such as the different syntactic variants that an OWL serialization allows.

[1] http://www.knowledgeweb.semanticweb.org/

This paper is structured as follows: Section 2 presents other interoperability evaluation initiatives and benchmark suites. Section 3 explains the experimentation-related tasks of the methodology that are being carried out. Section 4 introduces the benchmark suite for evaluating the import of OWL ontologies and how it is defined. Section 5 submits a proposal for extending this benchmark suite to cover the OWL DL and OWL Full sublanguages of OWL. Finally, Section 6 draws the conclusions from this work and proposes future lines of work.

2 Related Work

Interoperability is the ability of two or more systems or components to exchange information and to use this information exchanged [2]; in our case, it is the ability that ontology development tools have to interchange ontologies and use them.

As presented in [3], Semantic Web tools can interoperate in four different ways: by mapping ontologies in the source tool to ontologies in the target tool, by translating ontologies into a single pivot language, by translating ontologies into one language in a layered architecture of languages, or by generalising the pivot and layered approaches, not requiring either a fixed pivot language or a fixed layering of languages. This paper only deals with interoperability when translating ontologies to a single pivot language and does not cover other interoperability approaches.

The three following initiatives have also dealt with the interoperability problem:

OWL Test Cases. The OWL Test Cases[2] were developed by the W3C Web Ontology Working Group. Although these test cases might be used for evaluating the OWL importers of the ontology development tools, there are several differences between them and the benchmark suite presented in this paper:
  - The OWL Test Cases check if a tool deals correctly with the OWL language, clarify the formal meaning of the constructors, and show examples of their use. By constrast, our approach aims to evaluate exhaustively OWL importers.
  - We distinguish between the benchmarks that depend on the OWL knowledge model and those that depend on the RDF/XML syntax of the OWL files, all of which contain valid ontologies. On the other hand, the OWL Test Cases distinguish between tests that check the incorrect use of the OWL namespace, tests that check the importing of ontologies, entailment and non-entailment tests, and consistency and inconsistency tests.
  - The OWL Test Cases were defined for any tool that implements OWL knowledge bases. Our benchmark suite focuses on ontology development tools, although it can be used in any tool capable of importing OWL files.
  - The OWL Test Cases have tests for the three sublanguages of OWL (Lite, DL and Full) whereas our benchmark suite only deals with OWL Lite.

EON 2003 Workshop. The central topic of the Second International Workshop on Evaluation of Ontology-based Tools was the evaluation of ontology development tools interoperability [4]. In this workshop, the participants were asked to model ontologies with their ontology development tools and to perform different tests for evaluating the import, export and interoperability of the tools. In these experiments:

[2] `http://www.w3.org/TR/owl-test/`

- There was no constraint regarding the use of the interchange language; of the experiments carried out only two used OWL as interchange language.
- Each experiment was performed with a different procedure; hence the results obtained in that workshop did not provide general recommendations, only specific ones for each ontology development tool participating.

RDF(S) Interoperability Benchmarking. Knowledge Web had organised a benchmarking of the interoperability of ontology development tools using RDF(S) as the interchange language [3] before starting the benchmarking presented in this paper. These two benchmarking activities are quite similar as they follow the same methodology and their goals are almost identical. They differ in the language selected for interchanging ontologies: RDF(S) and OWL and this affects mainly to the ontology development tools that can participate in the benchmarking and to the benchmark suites used in the experimentation.

3 OWL Interoperability Benchmarking

In this benchmarking activity we have followed the Knowledge Web benchmarking methodology [5] for ontology tools, which has been used before for benchmarking the interoperability of ontology development tools using RDF(S) as the interchange language [6], and for benchmarking the performance and the scalability of ontology development tools [7].

The two main goals that we want to achieve with the benchmarking are:

- To **assess and improve the interoperability of ontology development tools** using OWL as the interchange language. To reach this aim would permit knowing the current interoperability of the tools and maximizing the knowledge that these tools can interchange while minimizing the information addition or loss.
- To **identify the fragment of knowledge that ontology development tools can share** using OWL as the interchange language. As this fragment becomes larger, more expressive ontologies can be interchanged among ontology development tools.

Since interoperability using an interchange language depends on the importers and exporters of the tool to that language, we have decided to evaluate them before evaluating the interoperability. Hence, the evaluation is performed in two consecutive stages:

Evaluation Stage 1. The OWL importers and the exporters of ontology development tools are evaluated and their results collected and analysed.

Evaluation Stage 2. The interoperability of ontology development tools is evaluated. The results, which are based on the results of the previous stage, will be collected and analysed, and we will obtain information about the interoperability of the tools and about the fragment of knowledge that they can share.

We have defined three different benchmark suites for evaluating the import, export, and interoperability of ontology development tools using OWL; then, the Knowledge Web members have assessed and agreed on them to ensure its quality.

[3] http://knowledgeweb.semanticweb.org/benchmarking_interoperability/

As we evaluate the interoperability of ontology development tools by means of their importers and exporters, any tool capable of importing and exporting OWL ontologies (for example, an ontology repository) can participate in the benchmarking.

The next section describes in detail the import benchmark suite while a complete description of the export and interoperability benchmark suites can be found in [6]. They are identical to the ones used in the RDF(S) interoperability benchmarking but differ in the procedure to run the benchmarks.

4 OWL Lite Import Benchmark Suite

The OWL Lite Import Benchmark Suite is intended to evaluate the OWL import capabilities of the ontology development tools by checking the import of the different combinations of the OWL Lite knowledge model.

The syntax chosen for serializing the OWL Lite ontologies to be imported is the RDF/XML syntax[4] because this is the syntax most used by ontology development tools when importing and exporting from/to OWL. As the RDF/XML syntax allows serializing ontology components in different ways while maintaining the same semantics, we decided to define separately the benchmarks that depend on the OWL knowledge model and those that depend on the syntax.

The OWL Lite Import Benchmark Suite is available in a public web page[5] and is composed of 82 benchmarks, which are divided in 12 groups (named from A to L).

Benchmarks that depend on the knowledge model. These benchmarks check the import of ontologies with different combinations of the OWL Lite vocabulary terms. Information about heading, versioning and annotation are not considered since these vocabulary terms are seldom used in most of the ontology development tools.

To identify all the combinations of components, we did as follows:

– **Classes.** We started by listing the different ways of describing classes that can be used in OWL Lite: with a class identifier, with a value or cardinality restriction on a property, or with the intersection operator. From these building blocks, we used the OWL class axioms (bearing into account the OWL Lite use restrictions) and defined the different ways of describing a class in OWL Lite with these class descriptions and axioms. The benchmark groups defined for classes are:
  - **A.** Class and class hierarchy benchmarks, including classes that are subclasses of value and cardinality restrictions on properties, and classes that are subclasses of class intersections.
  - **B.** Class equivalence benchmarks, including classes that are equivalent to value and cardinality restrictions on properties, and classes that are equivalent to class intersections.
  - **C.** Class defined with set operator bechmarks, including classes defined with the intersection operator.

[4] `http://www.w3.org/TR/rdf-syntax-grammar/`

[5] `http://knowledgeweb.semanticweb.org/benchmarking_interoperability/owl/import.html`

– **Properties.** We have combined object and datatype properties with the different RDF(S) constructs that can be used with properties and with the OWL constructs for defining relations between properties, global cardinality constraints, and logical property characteristics. The benchmark groups defined for properties are:
  D. Property and property hierarchy benchmarks.
  E. Property with domain and range benchmarks.
  F. Relation between properties benchmarks, including property equivalence and inverse properties.
  G. Global cardinality constraints and logical properties characteristics benchmarks, including symmetric, transitive, functional, and inverse functional properties.
– **Instances.** We have combined named and anonymous individuals with user defined properties and with individual equivalence and difference properties. The benchmark groups defined for instances are:
  H. Individual benchmarks, including instances of classes.
  I. Named individual and property benchmarks, including instances of classes with properties among them.
  J. Anonymous individual and property benchmarks, including anonymous individuals related to other (named) individuals.
  K. Individual identity benchmarks, including instance equivalences and differences.

Although the resulting benchmark suite is neither complete nor sufficient, we have decided not to define benchmarks for importing real complex ontologies because it would significantly increase the number of benchmarks to be executed and analysed and because it is much more difficult to diagnose the problem cause in complex cases than in simple ones. We have defined combinations of components with zero, one and two cardinalities (i.e. we have considered a property without domain, with one domain and with two domains). We have supposed that the results of cardinalities greater than two are the same as those of a cardinality of two.

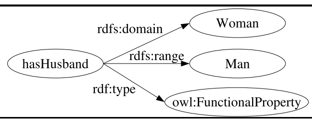
Benchmarks that depend on the syntax. These benchmarks check the correct import of OWL ontologies with different variants of the RDF/XML syntax, as stated in the RDF/XML specification. These benchmarks compose the group **L** of the benchmark suite and check the different ways of writing URI references (absolute URI references, URI references relative to a base URI, URI references transformed from *rdf:ID* attribute values, and URI references relative to an *ENTITY* declaration), empty nodes, multiple properties, typed nodes, string literals, blank nodes, and language identification attributes (*xml:lang*) in tags.

These syntactic variants are the same as those considered in the RDF(S) Import Benchmark Suite. However, the ontologies defined in each benchmark suite are different because in one case they are written in RDF(S) and in the other in OWL.

Each benchmark of the benchmark suite, as Table 1 shows, is described by a unique **identifier**, a **description** in natural language of the benchmark, a **formal description** in Description Logics notation of the ontology, a **graphical representation** of the ontology, and a **file** with the ontology in the RDF/XML syntax[6].

[6] All the files have been syntactically validated against the WonderWeb OWL Ontology Validator (`http://phoebus.cs.man.ac.uk:9999/OWL/Validator`)

Table 1. The description of a benchmark of the OWL Lite Import Benchmark Suite.

Identifier	**ISG03**
Description	Import a single functional object property whose domain is a class and whose range is another class
Formal description	$\top \sqsubseteq \leq 1\ hasHusband$ $\top \sqsubseteq \forall hasHusband^{-}.\text{Woman}$ $\top \sqsubseteq \forall hasHusband.\text{Man}$
Graphical representation	
RDF/XML file	... <owl:Class rdf:about="&ex;Woman"/> <owl:Class rdf:about="&ex;Man"/> <owl:ObjectProperty rdf:about="&ex;hasHusband"> <rdf:type rdf:resource="&owl;FunctionalProperty"/> <rdfs:domain rdf:resource="&ex;Woman"/> <rdfs:range rdf:resource="&ex;Man"/> </owl:ObjectProperty> ...

4.1 Evaluation criteria

The evaluation criteria of the OWL Lite Import Benchmark Suite are defined as follows:

Modelling (YES/NO). The ontology development tool can model the ontology components described in the benchmark.

Execution (OK/FAIL). The execution of the benchmark is normally carried out without any execution problem, and the tool always produces the expected result. When a failed execution occurred, the benchmarking participants are asked to provide information about the practices used when developing the OWL importers:
- The reasons for failing the benchmark execution.
- If the tool was corrected to pass a benchmark, which changes were performed.

Information added or lost. The information added or lost in the ontology interchange when executing the benchmark.

Table 2 shows an example of the execution of the benchmark ISG03 (i.e., *Import a single functional object property whose domain is a class and whose range is another class*) in five fictitious ontology development tools, identified by A, B, C, D, and E; with the intention of showing the possible variety of results in a benchmark.

In our example, tools A and B can model functional object properties and, therefore, their *Modelling* result is *YES*; tools C, D and E cannot model functional object properties and, therefore, their *Modelling* result is *NO*.

The expected result of tools A and B is a functional object property whose domain is a class and whose range is another class. Tool A imports all these components and adds a label with the name of the component to all the components; therefore, its *Execution* result is *OK* and it inserts new information into the ontology. Tool B imports the functional object property whose domain is a class, but it does not import the range class. As it does not produce the expected result, its *Execution* result is *FAIL*, and it loses information when importing the ontology.

Table 2. Ficticious results of executing benchmark ISG03

Tool	ID	Modelling	Execution	Information added	Information lost
A	ISG03	YES	OK	A label in all the components	-
B	ISG03	YES	FAIL	-	The property's domain
C	ISG03	NO	OK	-	The property as functional
D	ISG03	NO	OK	A cardinality of 1 in the property	The property as functional
E	ISG03	NO	FAIL	-	The property

Since tools C, D and E cannot model functional object properties though they can model object properties, the expected result of these tools is to obtain an object property whose domain is a class and whose range is another class. Tools C and D produce this expected result and their *Execution* result is *OK*; both lose the information about the object property being functional, though tool D also creates the property with a cardinality of 1 and therefore inserts new information in the ontology. Tool E does not import the functional object property at all, although its expected result is to import it as an object property; its *Execution* result is *FAIL*, and it loses all the information about the object property when it imports the ontology.

4.2 Procedure for Executing the Benchmark Suite

If a tool developer wants to evaluate the OWL importer of his tool, he has to execute each of the 82 benchmarks of the OWL Lite Import Benchmark Suite as follows:

1. To define the expected result, either by modelling the ontology expected from importing the file with the OWL ontology in the ontology development tool or by defining it informally (i.e., in natural language).
2. To import, from the ontology development tool, the OWL file which contains the OWL ontology defined in the benchmark.
3. To check whether the ontology modelled in the tool equals the imported one, examining also whether there is some information insertion or loss.

It is possible to automate this procedure, or part of it, thus saving time in the benchmark suite execution. For example, we can rely on the API of the ontology development tool for importing or comparing the ontologies.

5 Towards import benchmark suites for OWL DL and OWL Full

The OWL Web Ontology Language is composed of three layered sublanguages that increase expressiveness, and these are: OWL Lite, OWL DL and OWL Full. In a first approach, presented in the previous section, we have considered the import of OWL Lite ontologies to obtain a low number of benchmarks.

Nevertherless, every valid OWL Lite Ontology is a valid OWL DL ontology, and every valid OWL DL ontology is a valid OWL Full ontology [1]. Hence, the OWL Lite Import Benchmark Suite described in this paper might also be used for evaluating the importers from OWL DL and from OWL Full of ontology development tools.

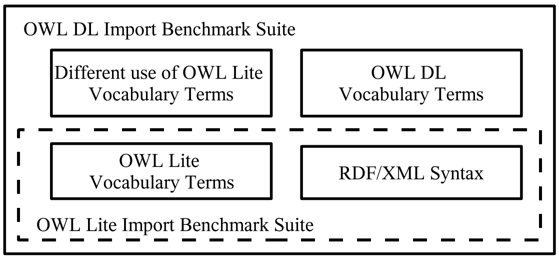


Fig. 1. The OWL DL Import Benchmark Suite

However, the definition of the OWL Lite Import Benchmark Suite does not take into account the OWL vocabulary terms whose use is not allowed in OWL Lite. Moreover, the use of the OWL vocabulary terms is restricted both in OWL Lite and in OWL DL.

5.1 OWL DL

As mentioned above the OWL Lite Import Benchmark Suite can be extended by implementing an OWL DL Import Benchmark Suite on top of it and, therefore, any ontology development tool that had already carried out the experiments of the OWL Lite Import Benchmark Suite would not need to repeat them.

The OWL Lite Import Benchmark Suite already provides us with benchmarks for evaluating the different combinations of the OWL Lite vocabulary terms and the different variants of the RDF/XML syntax. As Figure 1 shows, to cover the OWL DL sublanguage of OWL, we should also need to consider:

- The different combinations of the OWL Lite vocabulary terms according to their use in OWL DL because OWL DL imposes less restrictions to their use. Table 3 shows these differences in the use restrictions of the vocabulary terms[7].
- The different combinations of the OWL DL vocabulary terms not allowed in OWL Lite, and these with the OWL Lite vocabulary terms. The vocabulary terms allowed in OWL DL and not allowed in OWL Lite are: *owl:oneOf*, *owl:disjointWith*, *owl:unionOf*, *owl:complementOf*, *owl:hasValue*, and *owl:DataRange*.

However, when relaxing the restrictions in the use of OWL vocabulary terms from OWL Lite to OWL DL, a quite larger number of new benchmarks should be defined which affects the usability of the whole benchmark suite.

5.2 OWL Full

OWL Full has the same vocabulary terms than OWL DL has, but it places no restrictions in their use. In fact, OWL Full is a superset of RDF(S) that gives the user the freedom to extend the RDF(S) vocabulary with the OWL constructors and also to augment the meaning of both vocabularies [1].

The main characteristics in the use of OWL Full that are relevant to our case are:

[7] `http://www.w3.org/TR/owl-semantics/`

Table 3. Use restrictions in OWL Lite and DL

Vocabulary Terms	OWL Lite restrictions	OWL DL restrictions
owl:cardinality *owl:minCardinality* *owl:maxCardinality*	Object must be 0 or 1	Object must be any integer ≥ 0
owl:equivalentClass *rdfs:subClassOf*	Subject must be class names Object must be class names or restrictions	Subject must be OWL DL descriptions[8] Object must be OWL DL descriptions
rdf:type	Object must be class names or restrictions	Object must be OWL DL descriptions
rdfs:domain	Object must be class names	Object must be OWL DL descriptions
rdfs:range	Object must be class names or datatype names	Object must be OWL DL descriptions, datatype names or sets of data values
owl:allValuesFrom *owl:someValuesFrom*	Object must be class names or datatype names	Object must be OWL DL descriptions, datatype names or sets of data values
owl:intersectionOf	Used with lists of length greater than one of class names or restrictions	Used with lists of OWL DL descriptions

– The whole RDF(S) vocabulary can be used within OWL Full.
– OWL Full has no separation between classes, datatypes, datatype and object properties, annotation properties, individuals, data values, and the built-in vocabulary.
– Axioms in OWL Full do not need to be well formed.

This lack of restrictions implies that the use and possible combinations of the vocabulary terms in OWL DL and OWL Full is extremely different. Although the import benchmark suite for OWL DL could be used for evaluating the import of OWL Full ontologies, to develop a import benchmark suite for OWL Full it might not be sufficient to develop some new benchmarks on top of the import benchmark suite for OWL DL, but it must be necessary to create a whole new benchmark suite that covers all the differences between OWL DL and OWL Full.

This import benchmark suite for OWL Full should consider all the possible combinations of the OWL and RDF(s) vocabularies terms and, as the number of these combinations is high, it would be necessary to prune the generation of benchmarks as it was done for the RDF(S) Import Benchmark Suite [6].

6 Conclusions and future work

We are currently benchmarking the interoperability of ontology development tools using OWL as interchange language. In order to do so we are reusing most of the experimentation process followed in the RDF(S) interoperability benchmarking and part of its benchmark suites. The main change in the definition of the experimentation occurs in the OWL Lite Import Benchmark Suite presented in this paper.

[8] OWL DL descriptions can be class names, restrictions, boolean combinations of descriptions, or sets of individuals.

At the time of writing this paper, we do not have a definitive list of the tools that will participate in the benchmarking and the evaluation over the tools has not started. Therefore, we do not have conclusive results from any tool.

We think that benchmark suites like the one presented in this paper are necessary for tool developers because these benchmark suites offer them an easy way to evaluate their tools. We also encourage tool developers to participate in benchmarking activities like the one presented in this paper, to make the results available to the research and industrial communities, and to develop benchmark suites for evaluating their technology.

One drawback of the approach presented in this paper is that the execution of the benchmark suite is manual and, therefore, costly and hard to verify. Although automating all or part of the execution for a certain tool using its own APIs could be possible, to provide a fully-automated evaluation for every tool is not feasible. We are working on helping in this automation by providing an evaluation framework for executing the benchmark suite and making the benchmarks machine-processable.

Another way of continuing the work here presented is to define in detail the OWL Interoperability Benchmark Suite for OWL DL on top of the actual benchmark suite, as presented in Section 5.

Acknowledgments

This work is partially supported by a FPI grant from the Spanish Ministry of Education (BES-2005-8024), by the IST project Knowledge Web (IST-2004-507482) and by the CICYT project Infraestructura tecnológica de servicios semánticos para la web semántica (TIN2004-02660). Thanks to Rosario Plaza for reviewing the grammar of this paper.

References

1. Smith, M.K., Welty, C., McGuinness, D.L., (eds.): OWL Web Ontology Language Guide. W3C recommendation 10 february 2004. Technical report, W3C (2004)
2. IEEE: IEEE Std 610-1990 IEEE Standard Computer Dictionary: A compilation of IEEE Standard Computer Glossary (1990) New York, NY.
3. Euzenat, J., Stuckenschmidt, H.: the family of language approach to semantic interoperability. In Omelayenko, B., Klein, M., eds.: knowledge transformation for the semantic web, Amsterdam (NL), IOS press (2003) 49–63
4. Sure, Y., Corcho, O., eds.: 2nd International Workshop on Evaluation of Ontology-based Tools (EON2003). Volume 87 of CEUR-WS. (2003)
5. García-Castro, R., et al.: Specification of a methodology, general criteria, and benchmark suites for benchmarking ontology tools. Deliverable D2.1.4, KnowledgeWeb EU-IST Network of Excellence IST-2004-507482 (2005)
6. García-Castro, R., Gómez-Pérez, A.: Benchmark suites for improving the RDF(S) importers and exporters of ontology development tools. In Sure, Y., Domingue, J., eds.: Proceedings of the European Semantic Web Conference 2006. LNCS, Springer-Verlag (2006) 155–169
7. García-Castro, R., Gómez-Pérez, A.: Guidelines for benchmarking the performance of ontology management APIs. In: Proceedings of the 4th International Semantic Web Conference (ISWC2005). Number 3729 in LNCS, Galway, Ireland, Springer-Verlag (2005) 277–292