A Method for Performing an Exhaustive Evaluation of RDF(S) Importers

Raúl García-Castro and Asunción Gómez-Pérez

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

Abstract. Interoperability is one of the main quality criteria required for Semantic Web technology. In this paper we propose a method for defining benchmark suites for evaluating the RDF(S) importers of Semantic Web technology. We also show how this method was used for developing a benchmark suite that is being used for benchmarking the interoperability of ontology development tools.

1 Introduction

Interoperability is one of the main quality criteria required for Semantic Web technology. Users need to know which tools allow them to interchange their ontologies (or part of them) with other users or with other tools, contributing to the scalability of the Semantic Web. Therefore, the need of an objective evaluation of these tools according to their interoperability, that can be adapted to each case, is of high relevance.

As most of the Semantic Web information is available as RDF(S) ontologies in the web, in this paper we propose a method for defining benchmark suites for evaluating the RDF(S) import capabilities of Semantic Web technology. This method allows to create tailored benchmark suites, focusing on the components of interest of the RDF(S) knowledge model.

We also show how this method was used to develop a benchmark suite that is being used for benchmarking the interoperability of ontology development tools, in the context of the Knowledge Web Network of Excellence.

This paper starts presenting in Section 2 an overview of the benchmarking methodology for ontology tools and showing in Section 3 how this methodology is being applied for benchmarking the interoperability of ontology development tools. Section 4 describes the method used for defining the benchmark suites in the benchmarking and Section 5 presents the resulting benchmark suite for RDF(S) importers. Finally, Section 6 discusses our results and points out ongoing and future work.

2 Benchmarking Methodology for Ontology Tools

In the last decades, benchmarking has become relevant within the business management community as a continuous process for comparing the products,

services, and work processes of an organisation with those of the organisations that are recognised as representing best practices [1].

The Software Engineering community does not have a common benchmarking definition. Some authors consider benchmarking as a software evaluation method [2] while others adopt the business benchmarking definition, defining benchmarking as a continuous improvement process that strives to be the best of the best through the comparison of similar processes in different contexts [3].

A benchmark, in contrast, is a test that measures the performance of a system or subsystem on a well-defined task or set of tasks [4]. However, Sim et al. [5] propose to also measure tools and techniques to compare their performance.

This section summarises the benchmarking methodology developed by the authors in the Knowledge Web Network of Excellence [6]. The benchmarking methodology provides a set of guidelines to follow in benchmarking activities over ontology tools. This methodology adopts and extends methodologies of different areas such as business community benchmarking, experimental software engineering and software measurement as described in [6].

The benchmarking methodology for ontology tools is composed of a benchmarking iteration that is repeated forever. Each iteration is composed of three phases (*Plan*, *Experiment* and *Improve*) and ends with a *Recalibration* task:

- **Plan phase.** Its main goals are: to produce a document with a detailed proposal for benchmarking, including all the relevant information about it; to search for other organisations that want to participate in the benchmarking; and to agree on the benchmarking proposal and on the benchmarking planning with all the participants.
- **Experiment phase.** In this phase, the organisations must define and execute the evaluation experiments for each of the tools that participate on the benchmarking. The evaluation results must be compiled and analysed, determining the practices that lead to these results and identifying which of them can be considered as best practices.
- **Improve phase.** This phase comprises the writing of the benchmarking report, the communication of the benchmarking results to the participant organisations and finally, in several improvement cycles, the improvement of the tools and the monitorisation of this improvement.

The goal of the *Recalibration* task is not to improve the tools, but to improve the benchmarking process itself using the lessons learnt while performing it.

At the time of writing this paper, this methodology is being used in Knowledge Web for benchmarking the interoperability of ontology development tools[1]. A proposal for using this methodology for benchmarking the performance and the scalability of ontology development tools can be found in [7].

---

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

3   Interoperability Benchmarking

In the benchmarking activity that is taking place in Knowledge Web, the interoperability between two ontology development tools is assessed using RDF(S) files to exchange ontologies. To exchange ontologies from one ontology development tool into another, they must first be exported from the origin tool to a RDF(S) file and then this file must be imported into the destination tool.

This scenario requires that the importers and exporters from/to RDF(S) of the ontology development tools work accurately to be able to exchange ontologies correctly. Therefore, the benchmarking is composed of the following phases:

Agreement phase. The quality of the benchmark suites is essential for the results of the benchmarking. Therefore, the first step is that a group of experts reach an agreement on the definition of these benchmark suites.

Evaluation phase 1. In this phase, the RDF(S) importers and exporters of the ontology development tools are evaluated.

Evaluation phase 2. In this phase, the ontology exchange between ontology development tools is evaluated.

The method described in the next section was used for defining the benchmark suites that are being used for benchmarking the interoperability of ontology development tools. Section 5 shows how this method was applied for defining a benchmark suite for RDF(S) importers, taking as an input the knowledge model of RDF(S).

Similarly, for defining a benchmark suite for RDF(S) exporters, a common core of the knowledge model of the ontology development tools was taken as an input. This second benchmark suite is not described in this paper because of space constraints.

4   Definition of the Benchmark Suite

The benchmark suite for evaluating RDF(S) importers is composed of benchmarks that import an ontology with a simple combination of components of the RDF(S) knowledge model (classes, properties, etc.) [8]. Assessing the import of real, large or complex ontologies can be useless if we don't know if the importer can deal correctly with simple ones. Besides, it is easier to find problems in simple cases than in complex ones.

We have considered the import of all the possible combinations of the components of the RDF(S) knowledge model to make the benchmark suite exhaustive. There are four different types of benchmarks:

– **Benchmarks that import single components.** For each component of the knowledge model of RDF(S), we defined: a benchmark to import a single component and another to import several components. For example, for *rdfs:Class*, we defined two benchmarks to import:
  - One class.
  - Several classes.

– **Benchmarks that import all the possible combinations of two components with a property.** We defined all the combinations of two components related by a property, assigning cardinalities to the relations. These cardinalities define the different number of benchmarks that will be performed. For example, for *rdfs:Class* and the property *rdfs:subClassOf*, we defined five benchmarks to import:

- One class that is subclass of another class, being this last class subclass of a third one.
- One class that is subclass of several classes.
- Several classes that are subclass of the same class.
- One class that is subclass of another class and viceversa, forming a cycle.
- One class that is subclass of himself, forming a cycle.

– **Benchmarks that import combinations of more than two components that usually appear together in RDF(S) graphs**, such as properties that have both domain and range (*rdf:Property* with *rdfs:domain* and *rdfs:range*); statements that have subject, predicate and object (*rdf:Statement* with *rdf:subject*, *rdf:predicate* and *rdf:object*); and definitions of lists (*rdf:List* with *rdf:first*, *rdf:rest* and *rdf:nil*). For example, for a property with a domain and a range, we defined five benchmarks to import:

- One property that has as domain a class and as range another class.
- One property that has as domain a class and as range several classes.
- One property that has as domain several classes and as range another class.
- One property that has as domain several classes and as range another several classes.
- One property that has as domain and range the same class.

– **Benchmarks that import RDF(S) graphs with the different variants of the RDF/XML syntax**[2]:

- Different syntax of 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.
- Language identification attributes (*xml:lang*) in tags.
- Abbreviations of: empty nodes, multiple properties, typed nodes, string literals, blank nodes, containers, collections, and statements.

5   Resulting Benchmark Suite

The resulting benchmark suite obtained using the previous method is composed of a huge number of benchmarks, because of the large number of combinations that can exist between RDF(S) components. To make the benchmark suite more usable, we propose to prune it according to its intended use and to the kind of tools that it is expected to evaluate.

---

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

For example, for evaluating the RDF(S) importers of ontology development tools, we only considered the components of the knowledge model of RDF(S) that are most frequently used for modelling ontologies in these tools: *rdfs:Class*, *rdf:Property*, *rdfs:Literal*, *rdf:type*, *rdfs:subClassOf*, *rdfs:subPropertyOf*, *rdfs:range*, and *rdfs:domain*; not dealing with the rest of the RDF(S) components.

The definition of each benchmark in the benchmark suite, as Table 1 shows, includes the following fields:

– An identifier, for tracking the different benchmarks.
– A description of the benchmark in natural language.
– A graphical representation of the ontology to be imported in the benchmark.
– A file containing the ontology in the RDF/XML syntax.

Table 1. An example of a benchmark definition

| Identifier	I14
Description	Import one class that has the same property with several other classes
Graphical representation	

RDF/XML file	```<rdf:RDF xmlns="http://www.w3.org/2000/01/rdf-schema#"
	  xmlns:g1="http://www.test.org/graph14#"
	  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
	  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
	<Class rdf:about="http://www.test.org/graph14#class1">
	  <g1:prop1 rdf:resource="http://www.test.org/graph14#class2"/>
	  <g1:prop1 rdf:resource="http://www.test.org/graph14#class3"/>
	</Class>
	<Class rdf:about="http://www.test.org/graph14#class2"/>
	<Class rdf:about="http://www.test.org/graph14#class3"/>
	</rdf:RDF>```

The resulting benchmark suite[3] contains 72 benchmarks grouped in the following categories: class benchmarks, metaclass benchmarks, subclass benchmarks, class and property benchmarks, property benchmarks, subproperty benchmarks, property with domain and range benchmarks, instance benchmarks, instance and property benchmarks, and syntax and abbreviation benchmarks. Table 2 shows the benchmarks corresponding to the first three categories and Table 3 shows the graphical representations of these benchmarks.

---

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

Table 2. Benchmarks that import classes, metaclasses, and subclasses

Id.	Description
Class benchmarks	I01 Import just one class
	I02 Import several classes with no properties between them
Metaclass benchmarks	I03 Import one class that is instance of another class, being this last class instance of a third one
	I04 Import one class that is instance of several classes
	I05 Import several classes that are instance of the same class
	I06 Import one class that is instance of another class and viceversa
	I07 Import just one class that is instance of himself
Subclass benchmarks	I08 Import one class that is subclass of another class, being this last class subclass of a third one
	I09 Import one class that is subclass of several classes
	I10 Import several classes that are subclass of the same class
	I11 Import one class that is subclass of another class and viceversa, forming a cycle
	I12 Import just one class that is subclass of himself, forming a cycle

Table 3. Graphical representation of the benchmarks shown in Table 2
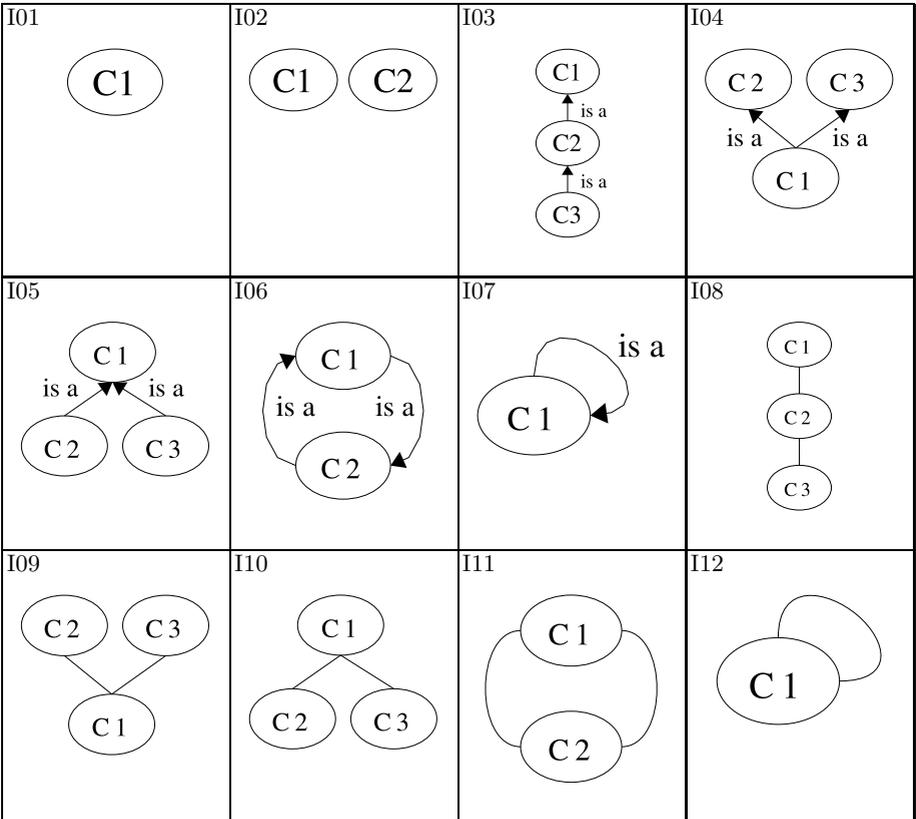
The execution of each benchmark in the benchmark suite comprises the following steps:

1. To define in the ontology development tool the expected result of importing the RDF(S) ontology.
2. To import the file with the RDF(S) ontology into the tool.
3. To compare the imported ontology with the expected ontology to check whether they are the same.

Although these steps can be performed manually, some automatic mean of performing them (or part of them) is highly advised when dealing with many benchmarks, specially for comparing the expected ontology with the imported one.

The expected results of a benchmark execution are:

− If the tool passes the benchmark or does not.
− If not, the reasons for not passing the benchmark. It could be because of a wrong implementation of the importer or because the tool cannot represent the RDF(S) component in its own knowledge model.
− If the tool does not pass the benchmark and is corrected to pass it, the changes performed.

The last two results have the goal of obtaining from the tool developers the practices used when developing the RDF(S) importers. This can allow to extract, if possible, the best practices performed by these developers.

6   Conclusions and Future Work

In this paper we present a method for defining benchmark suites for evaluating the RDF(S) import capabilities of Semantic Web technology. There are other benchmark suites for RDF like the RDF Test Cases [9], but they are not exhaustive enough and they are not flexible enough to be used for different evaluations. The RDF Test Cases are quite concrete, as they just deal with certain issues addressed by the RDFCore Working Group.

We also show how the method was applied for defining a benchmark suite for evaluating the RDF(S) importers of ontology development tools, and how this benchmark suite is being used for benchmarking the interoperability of ontology development tools in Knowledge Web.

This method was also used to define a benchmark suite for evaluating the RDF(S) exporters of ontology development tools[4]. Instead of taking as an input the RDF(S) knowledge model, a common core of the knowledge models of the ontology development tools was used to define the benchmark suite. The method can also be used to define benchmark suites specific to other languages such as OWL[5] or specific to the knowledge model of a certain tool.

---

[4] http://knowledgeweb.semanticweb.org/benchmarking_interoperability/
rdfs_export_benchmark_suite.html
[5] http://www.w3.org/TR/owl-features/

One important issue when defining benchmark suites with this method is to prune the resulting benchmark suite to make it more usable, as considering all the relations between RDF(S) components can result in thousands of benchmarks.

Combining the definitions of the benchmarks, the benchmark suites can be improved to include new benchmarks with more complex structures or with a higher number of components.

One future line of work would be to define different RDF(S) benchmark suites for different kinds of tools (ontology editors, ontology repositories, etc.) and for different kind of ontologies (ontologies with linear taxonomies, with graphs with cycles, with metaclasses, etc.).

Acknowledgments

References

1. Spendolini, M.: The Benchmarking Book. AMACOM, New York, NY (1992)
2. Kitchenham, B.: DESMET: A method for evaluating software engineering methods and tools. Technical Report TR96-09, Department of Computer Science, University of Keele, Staffordshire, UK (1996)
3. Wohlin, C., Aurum, A., Petersson, H., Shull, F., Ciolkowski, M.: Software inspection benchmarking - a qualitative and quantitative comparative opportunity. In: Proceedings of 8th International Software Metrics Symposium. (2002) 118–130
4. Sill, D.: comp.benchmarks frequently asked questions version 1.0 (1996)
5. Sim, S., Easterbrook, S., Holt, R.: Using benchmarking to advance research: A challenge to software engineering. In: Proceedings of the 25th International Conference on Software Engineering (ICSE'03), Portland, OR (2003) 74–83
6. García-Castro, R., Maynard, D., Wache, H., Foxvog, D., González-Cabero, R.: D2.1.4 specification of a methodology, general criteria and benchmark suites for benchmarking ontology tools. Technical report, Knowledge Web (2004)
7. García-Castro, R., Gómez-Pérez, A.: Guidelines for Benchmarking the Performance of Ontology Management APIs. In: To appear in Proceedings of the 4th International Semantic Web Conference (ISWC2005), Galway, Ireland (2005)
8. Brickley, D., Guha, R.: RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation 10 February 2004 (2004)
9. Grant, J., Beckett, D.: RDF Test Cases. W3C Recommendation 10 February 2004 (2004)