Benchmarking the RDF(S) Interoperability of Ontology Tools

Raúl García-Castro
Asunción Gómez-Pérez
Ontology Engineering Group, Departamento de Inteligencia Artificial.
Facultad de Informática, Universidad Politécnica de Madrid, Spain
York Sure
Institut AIFB, Universität Karlsruhe (TH), 76128 Karlsruhe, Germany

***Abstract*—The number of ontology tools, such as ontology editors and repositories, is constantly rising. Ideally, one could use them all seamlessly and thus benefit from all the functionalities they offer. As shown in previous EON workshops, interoperability among different development tools is not straightforward since ontology editors rely on specific internal knowledge models which are translated into common formats such as RDF(S). This paper addresses the urgent need for interoperability by providing an exhaustive set of RDF(S) benchmarks and demonstrating in an extensive field study the state-of-the-art of interoperability among six ontology tools. From the field study we have compiled a comprehensive set of best practices which may serve as guidelines. *Tool developers* benefit from having guidelines to design their import and export functionalities and a concrete set of benchmarks against which they can evaluate their import and export functionalities. *Ontology engineers* benefit from our work by having an overview to which extend interoperability is ensured for combinations of specific tools.**

I. Introduction

Ontologies enable interoperability among heterogeneous applications. The development and deployment of ontologies and ontology-based applications follows methodological guidelines and is supported by ontology tools such as ontology editors and repositories. Ideally one could use all existing ontology tools seamlessly and thus benefit from all the functionalities they offer. As shown in previous workshops on Evaluation of Ontology-based Tools (EON), interoperability among different ontology tools is not straightforward. For instance, ontology editors usually rely on specific internal knowledge models which are translated *more or less* into common formats such as RDF(S)[1]. Finding out why interoperability fails is cumbersome and non-trivial as any assumption made for the translation within one tool may easily prevent successful interoperability with other tools.

This paper addresses the urgent need for interoperability. We provide an exhaustive set of RDF(S) benchmarks which have been developed as part of the EU IST Knowledge Web Network of Excellence[2]. The benchmarks were designed to support evaluation and improvement of ontology tool interoperability. In an extensive field study we explored the state-of-the-art in interoperability among six ontology tools. Three of the participating tools are ontology editors (KAON, Protégé, WebODE) and three are repositories (Corese, Jena, Sesame), thus tool support for ontology development as well as deployment is covered. The field study helped us gain a deep understanding of the import and export functionalities of ontology tools. Our findings may serve as guidelines for developing tools and are summarized in comprehensive best practices on interoperability.

Tool developers benefit by having guidelines to design their import and export functionalities and by having a concrete set of benchmarks against which they can (automatically) evaluate their import and export functionalities. *Ontology engineers* benefit from our work by having an overview to which extend interoperability is ensured for combining specific tools. We hope that future generations of ontology tools will offer smooth interoperability and thus fulfil the key promise of ontologies.

This paper is structured as follows: Section II presents the motivation behind *benchmarking* software rather than *evaluating* it and other evaluation initiatives that have taken place. Section III examines how the RDF(S) interoperability benchmarking was conducted and how the RDF(S) benchmarks were designed. Sections IV and V summarize the results of executing the export, import and interoperability benchmarks. Section VI provides the recommendations extracted from benchmarking for ontology and software developers, and for anybody interested in carrying out a benchmarking activity. Finally, Section VII presents the conclusions derived from this work and future lines of work.

II. Related Work

A. Evaluation vs. Benchmarking

Software *evaluation*, according to the ISO 14598 standard [1], is defined as *the systematic examination of the extent to which an entity is capable of fulfilling specified requirements*; considering software not just as a set of computer programs but as the procedures, documentation and data produced.

Software *benchmarking* is a continuous process for improving software products, services and processes by systematically evaluating and comparing them to those considered to be the best. This definition, adapted from the definition given by the business management community [2], is followed by

[1] http://www.w3.org/RDF/
[2] http://knowledgeweb.semanticweb.org/

some authors in the Software Engineering community while others consider benchmarking as a software evaluation method [3].

The reason for benchmarking software products instead of just evaluating them is to obtain several benefits that cannot be obtained from evaluations. A software evaluation shows the weaknesses of the software or its compliance to quality requirements. If several software products are involved in the evaluation, we also obtain a comparative analysis of these products and recommendations for users. When benchmarking several software products, besides all the benefits commented, we gain continuous improvement of the products, recommendations for developers on the practices used when developing these products and, from these practices, those that can be considered best practices.

B. Related Evaluations

We now briefly present two evaluation initiatives closely related to our work. The first is a benchmark suite for evaluating RDF(S) usage, and the second is a previous evaluation of the interoperability of ontology development tools.

The *RDF Test Cases* [4] were created by the W3C RDF Core Working Group. These tests check the correct usage of the tools that implement RDF knowledge bases and illustrate the resolution of different issues considered by the Working Group. The RDF Test Cases could also be used for evaluating RDF(S) importers but, while they provide examples for, and clarification of, the normative definition of the language, our approach aims for an exhaustive evaluation of RDF(S) importers. Another difference is that we distinguish between the benchmarks that depend on the RDF(S) knowledge model and those that depend on the RDF syntax used. Moreover, we take into consideration valid input ontologies only, whereas the RDF Test Cases take erroneous input ontologies and entailment benchmarks.

The *Second International Workshop on Evaluation of Ontology-based Tools (EON 2003)* had as main topic the evaluation of the interoperability of ontology-based tools [5]. The results of the workshop led to significant improvements in various well-known ontology editors. The main reasons for benchmarking the interoperability of ontology tools again are:

- Interoperability is a great problem in the Semantic Web which is still unsolved.
- The workshop experiments (and those of its successors) involved only few tools and focused on editors.
- Some experiments evaluated export functionalities, others import functionalities and only a few evaluated interoperability.
- No systematic evaluation on a technical level was performed because ontology tool developers were just asked to model and interchange a domain ontology and report about findings. Each experiment used different test strategies and interchange languages, and also different principles for modelling ontologies. Therefore, only specific comments and recommendations were made.

We learnt many lessons from the results of the initial EON experiments which enabled us to do a systematic evaluation on a technical level such as the one presented in this paper.

III. RDF(S) Interoperability Benchmarking

The RDF(S) interoperability benchmarking started in Knowledge Web as an effort to improve the interoperability of ontology tools and to provide comprehensive recommendations for industry on how to use these tools. The benchmarking was organized and carried out following a generic software benchmarking methodology developed in Knowledge Web [6]. We now present the main decisions and outcomes of instantiating such methodology.

The goals for benchmarking the interoperability of ontology tools are related to the benefits pursued through it, and these are: to *evaluate and improve* their interoperability, to acquire a *deep understanding* of the practices used to develop the importers and exporters of these tools and to extract from these practices those that can be considered the *best practices*, to produce *recommendations* for users, and to create *consensual processes* for evaluating their interoperability.

These goals involve different communities that are related to the ontology development tools, namely, the research and industrial communities, and tool developers.

Participation in the benchmarking was open to any organisation irrespective of being a Knowledge Web partner or not. To involve other organisations in the process, with the goal of having the best-in-class tools participating, several actions were taken:

- The benchmarking proposal, a document being used as a reference along the benchmarking, was published as a public web page[3], which includes all the relevant information about the benchmarking: motivation, goals, benefits and costs, tools and people involved, planning, related events, and a complete description of the experimentation and the benchmark suites.
- Research was performed on the existing ontology development tools, both freely available and commercial ones, which could export and import to and from RDF(S), and their developers were contacted. In the future, any tool capable of importing and exporting RDF may participate in the benchmarking or benefit from the benchmarks.
- The interoperability benchmarking was announced with a call for participation through the main mailing lists of the Semantic Web area and through lists specific to ontology development tools.

Six tools took part in the benchmarking, three of which are ontology development tools: KAON, Protégé (using its RDF backend), and WebODE; the other three are RDF repositories: Corese, Jena and Sesame. These six tools do not share a common knowledge model and benchmarking was not always performed by tool developers.

The experimentation over the tools aimed to obtain results for interoperability improvement. Therefore, other quality at-

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

tributes such as performance, scalability, robustness, etc. were not considered. However, an approach for benchmarking the performance and scalability of ontology development tools can be found in [7].

The experimentation took into account the most common way of interchanging ontologies that ontology tools provide, that is, by exporting ontologies from a tool into an interchange language and then importing ontologies into the other tool, using RDF(S) as interchange language and serializing the ontologies into RDF/XML syntax. A future benchmarking activity inside Knowledge Web will cover the case of using OWL[4] as interchange language.

Interoperability of ontology tools using an interchange language depends on the capabilities of the tools to import and export ontologies from/to this language. Therefore, the experimentation included not only the evaluation of interoperability but also that of the RDF(S) import and export functionalities of the tools.

The evaluation criteria must describe in depth the import, export and interoperability capabilities of the tools, whereas the experiments to be performed in the benchmarking must provide data informing how the tools comply with these criteria. Therefore, to obtain detailed information about these capabilities, we need to know: the elements of the *internal knowledge model* of an ontology development tool that can be imported from RDF(S), exported to RDF(S), and interchanged with another tool using RDF(S) as interchange language; the *secondary effects* of importing, exporting and interchanging these components, such as insertion or loss of information; and the *subset of elements* of the internal knowledge models that these tools may use to interoperate correctly.

To obtain these experimentation data, we defined three benchmark suites for evaluating the import, export and interoperability capabilities of the tools [8], which are common for all the tools. As the quality of the benchmark suites to be used is essential for the results of the benchmarking, the first step was to agree on the definition of these suites. Then, we decided to perform the import and export experiments before the interoperability ones, as the results of the first influence those of the second.

The steps to follow for executing the three benchmark suites are similar, and these are: the definition of the expected ontology resulting from importing, exporting or interchanging the ontology described in the benchmark; the import, export, or interchange of the ontology defined in the benchmark; and the comparison of the expected ontology with the ontology imported, exported or interchanged, checking whether there is some addition or loss of information.

The benchmark suites were intended to be executed manually but, as they contain many benchmarks, it is highly recommended to execute them (or part of them) automatically. In the cases of Corese, Jena, Sesame, and WebODE, most of the experimentation was automated. In the other cases, it was performed manually.

[4]http://www.w3.org/TR/owl-features/

The benchmarking web page[3] contains a detailed description of the benchmark suites, all the files to be used in the experiments, templates for collecting the results, and the experimentation results obtained.

A. Benchmark Suites

The benchmark suites check the correct import, export and interchange of ontologies that model a simple combination of ontology components (classes, properties, instances, etc.). Because one of the goals of the benchmarking is to improve the tools, the benchmark ontologies are kept simple on purpose so as to isolate problem causes and to identify problems.

As the ontology tools participating in the benchmarking have different internal knowledge models, both the experimentation and the analysis of the results are based on a common group of ontology modelling primitives, available in RDF(S) and in these tools. However, since tackling this common group exhaustively would yield a huge number of benchmarks, we have only considered the components most used for modelling ontologies in ontology development tools: classes, instances, properties with domain and range, literals, and class and property hierarchies. The rest of the components have not been dealt with so far.

1) The RDF(S) Import Benchmark Suite: contains 82 benchmarks, which define a simple RDF(S) ontology serialized in a RDF/XML file that must be loaded into the ontology development tool.

To isolate the factors influencing the correct import of an ontology, we have defined two types of import benchmarks: those that evaluate the import of the different combinations of components of the RDF(S) knowledge model, and those that evaluate the import of the different variants of the RDF/XML syntax, as stated in the RDF/XML specification.

2) The RDF(S) Export Benchmark Suite: comprises 66 benchmarks, which define an ontology that must be modelled in the tool and saved to a RDF(S) file.

We have defined two types of benchmarks for isolating the two factors influencing the correct export of an ontology. One group of benchmarks evaluates the correct export of the combinations of components of the ontology development tool knowledge model and the other group evaluates the export of ontologies with concepts and properties whose names include characters restricted by RDF(S), such as those not allowed for representing RDF(S) or XML URIs.

3) The RDF(S) Interoperability Benchmark Suite: evaluates the interchange of ontologies from one source tool to a destination one and vice versa. Each benchmark defines an ontology that must be modelled in the origin tool, saved to a RDF(S) file, and loaded into the destination tool.

Since the factors influencing the correct interchange of an ontology (besides the correct functioning of the importers and exporters) as well as the knowledge model used for defining the ontologies are the same as those in the RDF(S) Export Benchmark Suite, the ontologies defined in the RDF(S) Interoperability Benchmark Suite are identical to those of the RDF(S) Export Benchmark Suite.

The evaluation criteria are common for the three benchmark suites and are defined as follows:

- **Modelling** (*YES*/*NO*). The ontology tool can model the ontology components described in the benchmark.
- **Execution** (*OK*/*FAIL*). The execution of the benchmark is carried out without any problem, and the tool always produces the expected result. In the case of a failed execution, the following information is required: reasons for the failure of the benchmark execution and, if the tool had been corrected to pass a benchmark, the corrections performed.
- **Information added or lost.** The information added or lost in the ontology interchange.

In the export and interoperability benchmark suites, if a benchmark defines an ontology that cannot be modelled in a certain tool, this benchmark cannot be executed in the tool, being the *Execution* result *N.E.* (Non Executed). In the import benchmark suite, even if a tool cannot model some components of the ontology, it should be able to import correctly the rest of the components.

IV. Import and Export Results

The results obtained when importing from and exporting to RDF(S) depend mainly on the knowledge model of the tool that executed the benchmark suite. The tools that natively support the RDF(S) knowledge model (Corese, Jena and Sesame, essentially the RDF repositories) do not need to perform any translation in the ontologies when importing/exporting them from/to RDF(S). The RDF repositories import and export correctly from/to RDF(S) all the combinations of components, as these operations do not require any translation.

In the case of tools with non-RDF knowledge models (KAON, Protégé and WebODE, the ontology development tools), some of their knowledge model components can also be represented in RDF(S) whereas some others cannot; on the other hand, tools do need to translate ontologies between their knowledge models and RDF(S). Besides, not all the combinations of components of the RDF(S) knowledge model that have been taken into account in the benchmarking can be modelled into all the tools.

We present an analysis of the import and export results of the participating ontology development tools.

A. Import Results

In general, ontology development tools import correctly from RDF(S) all or most of the combinations of components that they model, rarely adding or losing information. The only exceptions are: Protégé, which poses problems only when importing classes or instances that are instances of multiple classes, and WebODE, which causes problems only when importing properties with a XML Schema datatype as range.

When the ontology development tools import ontologies with combinations of components that they cannot model, they lose the information about these components. Nevertheless, they usually try to represent partially these components using other components from their knowledge models. In most cases, the import is performed correctly. The only exceptions are: KAON, which poses problems when it imports class hierarchies with cycles, Protégé poses problems when it imports class and property hierarchies with cycles and properties with multiple domains, and WebODE, which causes problems when it imports properties with multiple domains or ranges.

When dealing with the different variants of the RDF/XML syntax, ontology development tools import correctly resources with different URI reference syntaxes and with different syntaxes (shortened and unshortened) of empty nodes, of multiple properties, of typed nodes, of string literals, and of blank nodes. The only exceptions are: KAON when imports resources with multiple properties in the unshortened syntax; and Protégé when imports resources with empty and blank nodes in the unshortened syntax. The tools do not import language identification attributes (*xml:lang*) in tags.

B. Export Results

In general, ontology development tools export correctly to RDF(S) all or nearly all of the combinations of components that they model without losing information, though KAON poses problems only when exporting to RDF(S) datatype properties without range and datatype properties with multiple domains and with an XML Schema datatype as range, whereas Protégé causes problems only when exporting to RDF(S) classes or instances that are instances of multiple classes and template slots with multiple domains.

When these tools export components that are present in their knowledge model but cannot be represented in RDF(S), such as their own datatypes, they usually insert new information in the ontology though some information is lost.

When dealing with concepts and properties whose names do not fulfil URI character restrictions, each ontology development tool behaves differently. When names do not start with a letter or "_", some tools leave the name unchanged while others replace the first character with "_", spaces in names are replaced by "-" or "_", depending on the tool, and URI reserved characters and XML delimiter characters are left unchanged, replaced by "_", or encoded.

V. Interoperability Results

The RDF repositories (Corese, Jena and Sesame) interoperate correctly among themselves as they always import and export from/to RDF(S) correctly. This causes that interoperability between the ontology development tools and the RDF repositories depends only on the capabilities of the former to import and export from/to RDF(S) and, therefore, the results about this interoperability are identical to those presented in the previous section.

The import and export results presented in the previous sections showed that few problems arise when importing and exporting ontologies. Nevertheless, the interoperability results present more problems.

As a general comment we can say that interoperability between the tools depends on: a) the correct working of their

RDF(S) importers and exporters; and b) the way selected for serializing the exported ontologies in the RDF/XML syntax.

Furthermore, we have observed that some problems in any of these factors affect the results of not just one but of several benchmarks. This means that, in some cases, correcting a single import or export problem or changing the way of serializing ontologies can cause significant interoperability improvements.

Next, we list the components that can be interchanged between the tools.

1) Interoperability using the same tool: Ontology development tools seem to have no problems when the source and the destination of an ontology interchange are the same tool. The only exception resides in Protégé when interchanges classes that are instances of multiple metaclasses and instances of multiple classes, since Protégé does not import resources that are instances of multiple metaclasses.

2) Interoperability between each pair of tools: Interoperability between different tools varies according to the tools. Besides, as the detailed interoperability results show, in some cases the tools are able to interchange certain components from one tool to another, but not the other way round.

When **KAON** interoperates with **Protégé**, both can interchange correctly some of the common components that they are able to model. But the problems arise with classes that are instances of a single metaclass or of multiple metaclasses, with datatype properties without domain or range, with datatype properties whose range is *String*, with instances of multiple classes, and with instances related via datatype properties.

When **KAON** interoperates with **WebODE**, they can interchange correctly almost all the common components that these tools can model. The only exception occurs when they interchange datatype properties with domain whose range is *String*.

When **Protégé** interoperates with **WebODE**, they can interchange correctly all the common components that these tools can model.

3) Interoperability between all the tools: Interoperability between **KAON**, **Protégé** and **WebODE** can be achieved with nearly all the common components that these tools can model. The only components that these tools cannot use are: datatype properties with domain and whose range is *String*, and instances related through datatype properties.

Therefore, interoperability can be achieved among the tools that have participated in the benchmarking using classes, class hierarchies without cycles, object properties with domain and with range, instances of a single class, and instances related through object properties.

4) Interoperability regarding URI character restrictions.: Interoperability is low when tools interchange ontologies containing URI character restrictions in class and property names. This is mainly due to the fact that tools usually encode some or all the characters that do not comply with these restrictions, which provokes changes in class and property names.

VI. Recommendations

A. Recommendations for ontology engineers

This section offers recommendations for ontology engineers which use more than one ontology tool to build ontologies. Depending on the tools used, the level of interoperability may be greater or lower, as can be seen in Section V.

If the ontology is being developed bearing in mind interoperability, the ontology engineers should be aware of the components that can be represented in the ontology development tools and in RDF(S). Also, they should try to use in their ontologies the common knowledge components of these tools in order to avoid the knowledge losses known already.

Ontology engineers should also be aware of the semantic equivalences and differences between the knowledge models of the tools and the interchange language.

It is not recommended to name resources using spaces or any character that is restricted in the RDF(S), URI or XML specifications.

In the case of interoperability in the RDF repositories, although these repositories export and import correctly to RDF(S), ontology engineers should consider the limitations that other tools have when exporting their ontologies to RDF(S) with the aim of interchanging them.

B. Recommendations for tool developers

This section includes general recommendations for improving the interoperability of the tools when developing them. In [9], we offer more detailed recommendations to improve each of the participating tools according to the results and practices found. Although it is not compulsory to follow these recommendations, they help correct interoperability problems as we could observe when analysing the results.

Interoperability between ontology tools using RDF(S) as interchange language depends on how the importers and exporters of these tools work, whereas the way they work depends on the development decisions made by tool developers, who are different people with different needs. Therefore, it is not straightforward to provide general recommendations for developers. Nevertheless, some comments can be extracted from the analysis of the benchmarking results.

Seldom, a development decision will produce an interoperability improvement with some tools but a loss with others. For example, when exporting classes that are instances of a metaclass, some tools require that the class be defined as instance of *rdfs:Class* while some others require the opposite. The collateral consequences of the development decisions should be analysed by the tool developers.

Tool developers should be aware of the semantic equivalences and differences between the knowledge models of their tool and the interchange language; on the other hand, the tools should notify the user when the semantics is changed.

The first requirement for achieving interoperability is that tool importers and exporters are robust and work correctly when dealing with unexpected inputs. Although this is an evident comment, the results show that this requirement is

not fulfilled by the tools and that some tools even crash when importing some combinations of components.

Above all, tools should deal correctly with the combinations of components that can be present in the interchange language but that cannot be modelled in them. For example, cycles in class and property hierarchies cannot be modelled in ontology development tools. Nevertheless, these tools should be able to import these hierarchies by eliminating the cycles.

To export components commonly used by ontology development tools, they should be completely defined in the file. This means that metaclasses and classes in class hierarchies should be defined as instances of *rdfs:Class*, properties should be defined as instances of *rdf:Property*, etc.

Exporting complete definitions of other components can cause problems if these are imported by other tools. Not every tool deals with datatypes defined as instances of *rdfs:Datatype* in the file or with *rdf:datatype* attributes in properties.

Every exported resource should have a namespace if the document does not define a default namespace.

C. Recommendations for benchmarking

This section offers recommendations to perform benchmarking activities that were extracted from the lessons learnt while instantiating the methodology.

Benchmarking is not about comparing the results of the tools but the practices leading to these results. Therefore, experimentation should be designed to obtain these practices as well as the results.

Resources are needed mainly in three tasks: benchmarking organisation, experimentation definition and execution, and results analysis. It should be ensured that enough resources are allocated to each of these tasks.

Benchmarking is an activity that takes long time as it requires tasks that are not immediate: announcements, agreements, etc. Therefore, benchmarking activities should start early in time and the benchmarking planning should consider a realistic duration of the benchmarking.

Benchmarking needs the participation of relevant experts in the domain together with the best-in-class tools. Although it is not required that the tool developers participate in the benchmarking and perform the experiments over their tool, their involvement facilitates the execution and analysis of the experimentation results to a large extent. In all the cases where tool developers performed the experimentation on their own tools, they were able to detect problems and improve their tools while executing the benchmark suites.

VII. Conclusions and Future Work

Seamless interoperability among ontology tools greatly facilitates the development and deployment of ontologies. In this paper we present a set of concrete RDF(S) benchmarks and results and best practices obtained after applying them in an extensive field study on a number of well-known ontology tools.

Our benchmarks may be used by any ontology tool developer to improve import and export functionalities and to ensure interoperability with other tools on a very fine-grained level.

Our field study wants to show ontology engineers to which extent state-of-the-art tools are interoperable *right now*. However, ontology engineers may use the benchmarks themselves to benchmark relevant tools, e.g. in early ontology development project stages to facilitate the selection of appropriate tools.

During this benchmarking activity, tool developers sometimes automated the execution of the benchmark suites, but the experimentation was mainly done by hand. Carrying out experiments manually and analysing the results is expensive since both tasks depend on the expertise of the people performing them, and can be influenced by human errors. Therefore, these tasks should be automated as much as possible and mechanisms should be set up to detect human errors.

Future work mainly includes the development of means to automatize experimentation as much as possible and the development of appropriate OWL benchmark suites to be used in a future benchmarking activity that will consider interoperability using OWL as interchange language.

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 all the people that have participated in the RDF(S) interoperability benchmarking: Olivier Corby, Jesús Prieto-González, Moritz Weiten, and Markus Zondler. Thanks to Rosario Plaza for reviewing the grammar of this paper.

References

[1] ISO/IEC, *ISO/IEC 14598-1: Software product evaluation - Part 1: General overview*, 1999.

[2] M. Spendolini, *The Benchmarking Book*. New York, NY: AMACOM, 1992.

[3] R. García-Castro, "Keynote: Towards the improvement of the Semantic Web technology," in *Proceedings of the Second International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS2006)*, Athens GA, USA, November 2006.

[4] J. Grant and D. B. (eds.), "RDF test cases. W3C recommendation 10 february 2004," W3C, Tech. Rep., February 2004.

[5] Y. Sure and O. Corcho, Eds., *Proceedings of the 2nd International Workshop on Evaluation of Ontology-based Tools (EON2003)*, ser. CEUR-WS, vol. 87, Florida, USA, October 2003.

[6] R. García-Castro, D. Maynard, H. Wache, D. Foxvog, and R. González-Cabero, "D2.1.4 Specification of a methodology, general criteria and benchmark suites for benchmarking ontology tools," Knowledge Web, Tech. Rep., December 2004.

[7] R. García-Castro and A. Gómez-Pérez, "Guidelines for benchmarking the performance of ontology management APIs," in *Proceedings of the 4th International Semantic Web Conference (ISWC2005)*, ser. LNCS, no. 3729. Galway, Ireland: Springer-Verlag, November 2005, pp. 277–292.

[8] ——, "Benchmark suites for improving the RDF(S) importers and exporters of ontology development tools," in *Proceedings of the 3rd European Semantic Web Conference (ESWC2006)*, ser. LNCS 4011. Budva, Montenegro: Springer-Verlag, June 2006.

[9] R. García-Castro, Y. Sure, M. Zondler, O. Corby, J. Prieto-González, E. P. Bontas, L. Nixon, and M. Mochol, "D1.2.2.1.1 Benchmarking the interoperability of ontology development tools using RDF(S) as interchange language," Knowledge Web, Tech. Rep., June 2006.