Ontology testing based on requirements formalization in collaborative development environments

Alba Fernández-Izquierdo

Ontology Engineering Group, Universidad Politécnica de Madrid
albafernandez@fi.upm.es

Abstract. In the last years, ontologies are commonly developed in collaborative environments where there are several experts involved in the development process. When the ontology engineers deliver an ontology the users expect a complete ontology which fulfils all the requirements they ask for. However, in most cases the ontology engineers cannot prove this completeness, as they did not carry out any testing process. Adopting testing as an activity in ontology development will lead to an improvement on the quality of ontologies regarding its completeness, as they are going to be tested before its publication. In the PhD proposed, we introduce an ontology testing framework whose aim is to assure both ontology engineers and users the completeness of an ontology.

1 Problem statement

The increasing uptake of semantic technologies and ontologies has led during the past years to the study of collaborative ontology engineering [7], where ontology engineers can be working together with different domain experts in a distributed environment to build an ontology. These domain experts may have their own interests about which knowledge needs to be modelled in the ontology.

This situation leads to the fact that an ontology engineer needs to represent in one ontology several ontological requirements gathered from different sources and which may represent different interests. The ontology engineer should guarantee that all these requirements are satisfied. Besides this, the alignment with standardized ontologies, such as SNOMED, is also becoming a popular request from domain experts. Because of this, it is useful for ontology engineers to have a testing process which supports the verification and validation of the ontology regarding the requirements the domain experts ask for.

Up to now, the majority of traditional methodologies for ontology development [2] consider a centralized scenario where the ontology is created by a few experts. Even if they consider ontology testing, it is set in a non-collaborative environment and oriented to ontology engineers without taking into account domain experts. In the last years, approaches for ontology development which consider collaborative scenarios (e.g., [12], [11]) and which take into consideration the validation of requirements (e.g., [3]) have been introduced, but none of

them provides a testing framework. Apart from these methodologies some ontology testing approaches [1][5] also exist, but they consider a centralized ontology development and do not support all the aspects needed for a testing framework, i.e., methodological background, tests generation and test execution.

The main idea of the proposed work is to provide an ontology testing framework which can be integrated in collaborative ontology development to validate and verify an ontology regarding its requirements. The resulting product resultant can be used by ontology engineers and domain experts to assure ontologies completeness. To accomplish this goal, we propose to formalize the ontological requirements into SPARQL queries and to store them in RDF files with metadata.

2 Relevancy

Nowadays, in software engineering it is inconceivable to deliver a software without its pertinent tests which guarantee its correct behaviour; ontology engineering should follow the same principle. Ontology engineers should have artefacts associated to their delivered ontologies which guarantee ontology quality, e.g. evaluation reports and documentation.

In this PhD we focus on ontology testing as a process needed by ontology engineers to verify and validate requirements, which are essential for ontology development as they represent the reason the ontology is constructed. This testing process can be used by ontology engineers to know if the ontology is complete regarding the users' requirements, and also by users to know if the ontology represents their interests.

3 Related work

Several approaches which defend the importance of the validation of ontological requirements have been developed. Each of these approaches focuses on some testing aspects: methodological background, test design and implementation, or traceability between the ontology and the tests. Their main limitation lies in the fact that, even if they acknowledge the need for evaluation, none of them covers in detail all the tasks of the testing process.

Regarding the methodological background of ontology testing, Vrandevcic and Gangemi [16] introduce the notion of testing ontologies borrowing ideas from software engineering and unit testing. They explore different options for testing, e.g. testing with the axioms and negations or formalizing competency questions [6]. A more recent work was the one presented by Peroni [3] who describes SAMOD, an agile and collaborative methodology for ontology development which uses tests to validate the ontology. SAMOD proposes some steps and principles to define test cases from the ontological requirements. These approaches are focused on methodology, however, they do not mention how to design and implement the tests or how to maintain traceability.

Regarding the test design and implementation, Keet and Lawrynowicz [9] proposed a test-driven development of ontologies, where the competency questions are formalized into axioms and added to the ontology if they are not present. In their work they focus on how the tests should be defined depending on the axiom to be added. Dealing with the same testing aspect, the OntologyTest tool [5] allows a user to define and execute a set of tests to check functional requirements over an ontology. These tests are stored in an XML file for future reuse. Another approach to define and implement test cases is the one presented by Ren et al. [13] which uses Competency-Question-driven authoring. In this work the authors use natural language processing and patterns to analyse the competency questions and automatically obtain SPARQL queries to test the ontology. Even these approaches are focused on design and implementation, they neither mention how to maintain traceability between the tests and the ontology nor describe a methodology to support their implementation.

Finally, Blomqvist et al. [1] present an agile approach which includes a methodological background for testing and introduce in rough outlines several types of tests. The paper distinguishes three types of tests focused on testing over instances of the ontology: competency question verification, inference verification and error provocation. The first two are concerned with verifying the correct implementation of a requirement and the third is intended to expose faults. All the tests are stored in a different OWL ontology with information about the requirement, e.g., type of test or expected output. By saving these test suites in an ontology it allows the user to reuse them and to maintain traceability between the requirements and all the associated information. However, this methodology does not explain how to design the tests it describes.

Even if all the mentioned works introduce testing through requirements, none of them proposes a complete testing process which could cover formally all the mentioned testing aspects, i.e. methodology, test design and implementation, and traceability. In addition to this, it is also necessary to consider that ontology development is increasingly becoming a collaborative process with different roles involved, and ontology testing approaches should provide support for it.

4 Research questions

The main research question of this PhD proposal directly derives from the problem statement and can be declared as follows:

- *Does the formalization of ontological requirements improve the quality of ontologies?*

Important corresponding sub-questions are:

- *Does the formalization of ontological requirements into a set of SPARQL queries and their storage in the machine-processable RDF format improve the validation of the ontologies reducing the coverage analysis time and errors?*
- *Does having the formalization of ontological requirements into a set of SPARQL queries before the implementation of the ontologies improve the development regarding the reduction of development time and errors?*

5 Hypotheses

The hypotheses associated to the research questions are the following:

- *Having the ontological requirements formalized into a set of SPARQL queries and stored in RDF files reduce the time and errors in ontology validation compared to manual ontological requirements validation.*
- *Having the ontological requirements formalized into a set of SPARQL queries and stored in RDF files before the ontology implementation reduces the time of the implementation of each requirement in comparison with the traditional ontology development.*

6 Approach

As mentioned, the PhD thesis aims to provide an approach for ontology testing to verify and validate the ontology regarding its requirements. We first offer a general approach which includes the phases that should be carried out in any ontology testing process. Besides these general phases, we introduce an ontology testing framework and several development scenarios where it can be integrated.

6.1 Generic phases in ontology testing

This subsection introduces the phases proposed to be carried out in any ontology testing process. In the literature [10][5] the testing process is usually divided into two phases, i.e., *test implementation* and *test execution*, even if there are no formal descriptions of each one. In this approach we describe the goals of these phases and also propose a new one, *test design*, in order to facilitate the reuse of tests over the same or other ontologies. The phases proposed here can be adopted by any ontology testing process regardless of the ontology development methodology in which this testing activity is integrated.

- **Test design**. During this phase a first draft of the test cases is designed, providing an overview of the questions the ontology should be able to answer. These test cases may not include technical information, e.g., URIs of the ontology, so they can be reused by other ontologies.
- **Test implementation**. The tests have to be completed regarding the technical information of the ontology to which the requirements belong. This information needs to be given to the responsible of creating the tests cases.
- **Test execution.** Once the tests cases are completed, the ontology engineers can check the test cases to validate the completeness of the ontology.

6.2 Ontology testing framework

Exploiting the testing phases introduced in the previous subsection we propose an ontology testing framework which can be integrated as an activity in any ontology development methodology.

Ontology testing roles. This framework considers the involvement of three roles in the testing process. The two first roles are the most popular ones identified in recent ontology development methodologies [14], i.e., *Domain expert* and *Ontology engineer*. The third role is the *Ontology tester*. Each role has specific responsibilities and consequently will have specific associated tasks.

- *Domain expert.* She is the individual responsible of giving the information needed to model the domain and identifying the ontological requirements.
- *Ontology engineer.* She is the individual who has to implement the ontology following the requirements and constraints identified by the domain experts.
- *Ontology tester.* She is the individual responsible of creating the tests. The same person can play the role of ontology tester and ontology engineer, both need knowledge about ontologies.

Ontology testing phases. In order to guide the testing process, we propose a description of the tasks of each testing phase and how they have to be carried out. Figure 1 summarizes the phases and each input and output.

- *Test design.* During this phase the ontological requirements, which will be written in the form of competency questions and stored in an ORSD (Ontology Requirements Specification Document)[15], are formalized into SPARQL queries by the ontology testers to avoid the ambiguities inherent to natural language. The test cases are defined in the RDF language and include the SPARQL queries and metadata to provide traceability between the requirements and the ontology. The test cases are stored in files with the aim of using them also as regression tests or to share them.
- *Test implementation.* During this phase the test cases are completed regarding the ontology technical information, such as URIs. This technical information is given to the ontology tester in the form of a glossary of terms [4]. The tests cases are published online to allow reuse between other developers.
- *Test execution.* During this phase the SPARQL queries of each test case are executed over the ontology.

6.3 Scenarios for applying the ontology testing framework: Test-driven development of ontologies

This ontology testing framework can be integrated in different scenarios. Concretely, we integrate it in *test-driven development of ontologies* (TDD), which is an agile approach already introduced by Keet and Ławrynowicz [9] based on software engineering [17]. TDD can be summarised as: (1) Write a test based on a requirement, (2) Run all tests to check that the new test fails, (3) Update the ontology to pass the test, (4) Run the test to verify it passes, (5) Refactor the ontology, and (6) Run all tests to verify the completeness of the ontology.

In this scenario the test design and implementation are carried out before the representation of the associated requirement in the ontology with the aim of guiding the ontology development. The first execution of the tests is also

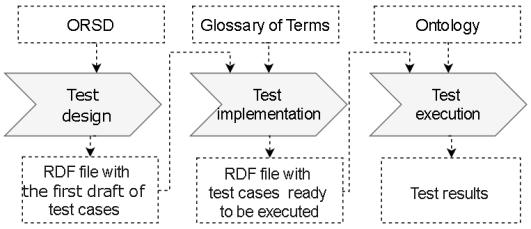

Fig. 1: Ontology testing framework phases with their inputs and outputs

performed before the ontology implementation, to assure that the requirements formalized are not implemented yet in the ontology and, consequently, not redundant. Figure 2 represents the tasks needed to be carried out by each role for developing each ontological requirement following test-driven development.

Since TDD is an agile approach, the ontologies are developed incrementally by iterations according to the domain experts needs. The proposed ontology testing framework supports this scenario, because the tests cases can be incrementally generated in an RDF file allowing for regression testing.

Alternative scenarios. Other scenarios can be considered for applying this testing process. Examples of them are *ontology certification*, where the tests are used at the end of the development process to certify the alignment with an external ontology, or *traditional ontology development*, where the tests are generated and executed at the end of the development process to validate it.

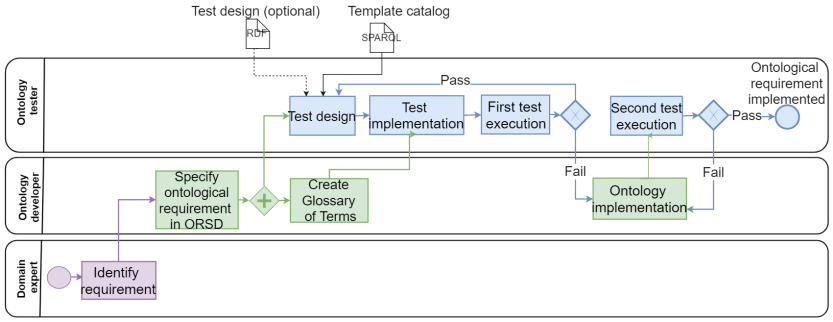

Fig. 2: Tasks and roles involved in the implementation of a requirement

7 Evaluation plan

The proposed approach will be evaluated through an experiment with users in four different ontology development scenarios:

- **Scenario 1**: The ontology engineer will implement the ontology based on the requirements in natural language and validate manually its completeness.
- **Scenario 2**: The ontology engineer will implement the ontology based on the requirements in natural language. After this, the test cases written in the RDF language need to be generated to validate the ontology.
- **Scenario 3**: The ontology engineer will implement the ontology using the requirements in natural language and the test cases in RDF. These test cases in RDF are used after the ontology implementation to validate the ontology.
- **Scenario 4**: The ontology engineer will generate the test cases in RDF from the natural language requirements. Then she will implement the ontology using them. These test cases are used to validate the ontology implemented.

We plan to extract from the experiment:

- The implementation time of each requirement in the different scenarios to analyse the impact of having formalized tests in the ontology development.
- The implementation time of each test case in Scenarios 2 and 4, to analyse if it is easy to formalize the requirements before the ontology implementation.
- The test results in Scenarios 2, 3 and 4, to analyse if the developers can improve the efficiency of the validation of the ontology with the tests.
- The validation results in Scenario 1 to analyse if the developers can obtain better results with automatic tests than with manual tests.

With this information we expect to validate the hypothesis stated in Section 5 proving that this ontology testing framework reduces errors regarding the validation of the ontology and also reduces implementation time of the ontologies by integrating test-driven development with ontology development.

8 Reflections

The importance of requirements is emphasized in the majority of ontology development methodologies, even if it is not considered an independent activity. We expect that adopting ontology testing as an activity in ontology development will improve the quality of ontologies regarding its completeness.

Furthermore, as in software engineering there are testing standards, e.g. ISO/IEC/IEEE 29119 [8], we think that having a testing framework which provides methodological and technological support could motivate the creation of standardized representations of tests in ontology engineering. This standardization could allow the reuse of tests over ontologies, the import of already published tests or the guarantee that the ontologies satisfy a minimum of completeness.

Finally, we think that the RDF language is enough to describe the metadata we consider necessary to provide traceability and understanding, but in future works we will analyse if adding semantics using the OWL language can benefit the tests.

Acknowledgements: I would like to acknowledge my advisor Raúl García-Castro. This work has been partially supported by the VICINITY H2020 project (688467).

References

1. Eva Blomqvist, Azam Seil Sepour, and Valentina Presutti. Ontology testing-methodology and tool. In *International Conference on Knowledge Engineering and Knowledge Management*, pages 216–226. Springer, 2012.
2. Oscar Corcho, Mariano Fernández-López, and Asunción Gómez-Pérez. Methodologies, tools and languages for building ontologies. Where is their meeting point? *Data & knowledge engineering*, 46(1):41–64, 2003.
3. DISI DASPLab. A Simplified Agile Methodology for Ontology Development. In *OWL: Experiences and Directions–Reasoner Evaluation: OWLED-ORE 2016, Bologna, Italy, November 20, 2016, Revised Selected Papers*, volume 10161, page 55. Springer, 2017.
4. Mariano Fernández-López, Asunción Gómez-Pérez, and Natalia Juristo. Methontology: from ontological art towards ontological engineering. 1997.
5. Sara García-Ramos, Abraham Otero, and Mariano Fernández-López. Ontologytest: A tool to evaluate ontologies through tests defined by the user. In *International Work-Conference on Artificial Neural Networks*, pages 91–98. Springer, 2009.
6. Michael Grüninger and Mark S Fox. Methodology for the Design and Evaluation of Ontologies. 1995.
7. Clyde W Holsapple and Kshiti D Joshi. A collaborative approach to ontology design. *Communications of the ACM*, 45(2):42–47, 2002.
8. ISO/IEC/IEEE. Software and systems engineering - software testing - Part 3: Test documentation, 2013.
9. C Maria Keet and Agnieszka Ławrynowicz. Test-driven development of ontologies. In *International Semantic Web Conference*, pages 642–657. Springer, 2016.
10. Agnieszka Lawrynowicz and C Maria Keet. The TDDonto tool for Test-Driven Development of DL Knowledge bases. In *Description Logics*, 2016.
11. Raúl Palma, Peter Haase, Oscar Corcho, Asunción Gómez-Pérez, and Qiu Ji. An editorial workflow approach for collaborative ontology development. *ASWC*, 8:227–241, 2008.
12. H Sofia Pinto, Steffen Staab, and Christoph Tempich. DILIGENT: Towards a fine-grained methodology for distributed, loosely-controlled and evolving engineering of ontologies. In *Proceedings of the 16th European Conference on Artificial Intelligence*, pages 393–397. IOS Press, 2004.
13. Yuan Ren, Artemis Parvizi, Chris Mellish, Jeff Z Pan, Kees Van Deemter, and Robert Stevens. Towards competency question-driven ontology authoring. In *European Semantic Web Conference*, pages 752–767. Springer, 2014.
14. Elena Simperl and Markus Luczak-Rösch. Collaborative ontology engineering: a survey. *The Knowledge Engineering Review*, 29(01):101–131, 2014.
15. Mari Carmen Suárez-Figueroa, Asunción Gómez-Pérez, and Boris Villazón-Terrazas. How to write and use the ontology requirements specification document. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, pages 966–982. Springer, 2009.
16. Denny Vrandečić and Aldo Gangemi. Unit tests for ontologies. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, pages 1012–1020. Springer, 2006.
17. Laurie Williams, E Michael Maximilien, and Mladen Vouk. Test-driven development as a defect-reduction practice. In *Software Reliability Engineering, 2003. ISSRE 2003. 14th International Symposium on*, pages 34–45. IEEE, 2003.