OnToology, a tool for collaborative development of ontologies

Ahmad Alobaid [1], Daniel Garijo [1*], María Poveda-Villalón [1], Idafen Santana-Perez [1] and Oscar Corcho [1]

[1]Ontology Engineering Group, Universidad Politécnica de Madrid, Spain

ABSTRACT

In this demo we present OnToology, a tool for developing ontologies collaboratively using Github. OnToology addresses several steps of the ontology development lifecycle, including documentation, representation, evaluation and publication in a non-intrusive way.

1 INTRODUCTION

The rise of collaborative technologies has sped up the development of software on the last decade. When working as a team, it is common to use repositories for software development, open discussions and having a ticketing system that warns and keeps track of the main issues to be solved.

This paradigm is slowly moving towards other domains, like ontology development. Ontologies, like software, require a set of requirements to be stablished and are usually discussed in a group before agreeing on a design decision. Therefore, they benefit heavily from the ticketing system, versioning and decision tracking that collaborative environments offer. However, this is often not enough, as ontologies need to be further documented and published online. Although some tools cover part of these activities e.g. documentation and evaluation, there are no tools that integrate them with a collaborative environment.

In this demo we present OnToology[1] a tool for documenting, evaluating, presenting and publishing ontologies developed collaboratively. Section 2 describes the requirements for developing ontologies collaboratively, while Section 3 describes our approach. Finally Section 4 describes related work and Section 5 introduces our efforts for improving the tool.

2 ONTOLOGY DEVELOPMENT LIFE CYCLE

Typically, the ontology development process can be divided in several independent activities:

- Ontology requirements: before committing to implement an ontology, it is advised to write a set of competency questions (CQs) in an *ontology requirements specification document* as mentioned in NeOn methodology (Suárez-Figueroa *et al.*, 2012), which will be used to test the ontology.
- Ontology Implementation: once agreed on the ontology requirements, one can use an ontology editor such as NeOn-toolkit[2] or Protégé[3] to design the properties and classes of the proposed ontology.
- Ontology evaluation: the resultant ontology can be evaluated in two different ways: by checking whether the requirements (i.e., CQs) are answered properly and by checking whether the ontology follows design patterns and well stablished practices for its implementation or not.
- Ontology documentation: an ontology is unlikely to be reused unless it is documented properly with examples. This phase focuses in producing a human-readable documentation that allows users understand the OWL or RDFs file produced during the implementation phase.
- Ontology publication: in this phase the ontology has been agreed on and its ready for release. As the aim of the vocabularies and ontologies is normally to share the model for its reuse, the ontology is released with its documentation.

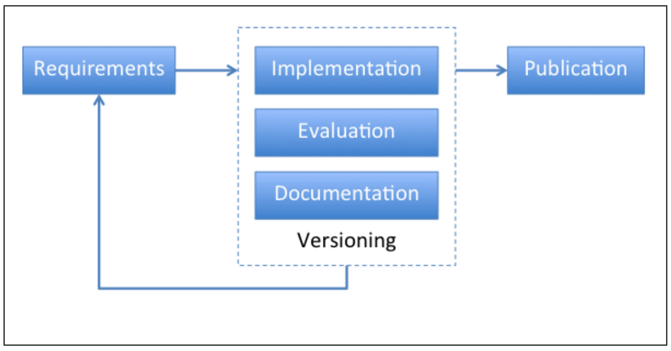

Fig. 1: Ontology development life cycle

Figure 1 presents an overview of the different phases of the ontology development lifecycle. As shown in the figure, this cycle benefits from a collaborative versioning environment that tracks the changes made to the ontology, requirements, documentation and diagrams; and keeps a log of the group discussions and decisions made.

3 COLLABORATIVE CREATION OF ONTOLOGIES WITH ONTOOLOGY

OnToology[4] is a web-based tool designed to automate part of the ontology development process in collaborative environments. In particular, OnToology is designed to work with Github[5], one of the most common environments for software development. After registering a repository to OnToology, developers just push their changes to Github and the tool will produce the documentation (with several proposals for diagram representation), evaluation and publication of the ontology in the user's repository. The phases covered by OnToology are further described below:

*To whom correspondence should be addressed: dgarijo@fi.upm.es

[1] http://purl.org/net/OnToology

[2] http://neon-toolkit.org/

[3] http://protege.stanford.edu/

[4] https://github.com/OnToology/OnToology

[5] http://www.github.com/

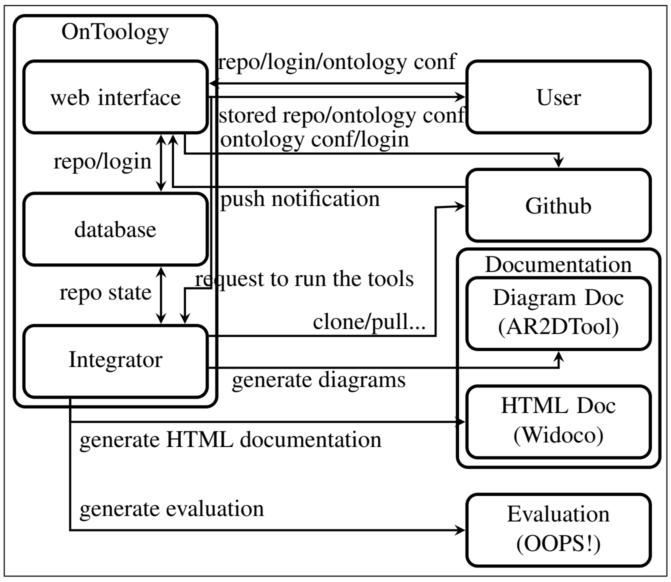

Fig. 2: OnToology Architecture

- **Ontology documentation and depiction:** OnToology integrates AR2DTool[6] for creating taxonomy and entity relationship diagrams of the ontology and Widoco[7], which helps you to create a complete HTML documentation by following a series of steps in a wizard. Widoco is based on LODE (Peroni, S. *et al.*, 2012) and guides the user along the documentation process. It also extracts some of the metadata properties from the ontology, annotates the documentation in RDF-a and creates provenance summary that is W3C PROV-O[8] compliant.
- **Ontology Evaluation:** OnToology integrates OOPS! (OntOlogy Pitfall Scanner!) (Poveda-Villalón *et al.*, 2014), a web based system[9] that helps in the evaluation of OWL ontologies relying mainly on structural and lexical patterns that identify pitfalls in ontologies. OOPS! has been designed to detect up to 33 pitfalls among those defined in the catalogue, and has been used worldwide in different domains. For each ontology, OnToology will create a single issue in Github with a summary of the pitfalls detected by OOPS!, pushing as well an extended explanation to the repository for more information.
- **Ontology Publication:** by using Widoco, OnToology produces a bundle with the documentation that is ready to be deployed on a server.

OnToology allows developers to customize which of the integrated tools are enabled or disabled through a configuration file.

3.1 OnToology Architecture

OnToology is composed of two main parts and is integrated with four external systems as can be seen in Figure 2. The two main parts of the tool are the *web interface* and the *integrator*. The purpose of the *web interface* is to handle Github notifications of the changes of a registered ontology repository and to serve the webpage where users register their repositories after giving the tool access to the repository. There is another webpage served by the *web interface* where users can log into and configure their ontology. The configuration file of an ontology is used by OnToology to enable/disable the generation of each of the tools.

The *integrator* talks to four systems: AR2DTool, Widoco, OOPS! and Github. The first three systems are used by OnToology to produce the diagrams, the HTML documentation and an evaluation report respectively. The integration with Github consists on cloning the repository, adding OnToology user as a collaborator, creating webhooks (that are responsible for notifying OnToology of the changes on repository), aggregating the produced files from the integrated systems and submitting them in a pull request to the repository, where the maintainer can review the changes and merge them later on. The *integrator* also opens an issue in Github including the summary of the pitfalls generated by OOPS! with a link to a full extended explanation.

[6] https://github.com/idafensp/AR2DTool/

[7] https://github.com/dgarijo/Widoco/

[8] http://www.w3.org/TR/2013/REC-prov-o-20130430/

[9] http://oops.linkeddata.es/

4 RELATED WORK

Neologism (Cosmin Basca *et al.*, 2008) is a web-based editor for vocabulary editing and publishing. Unlike OnToology, it lacks revision tracking and ontology best practice evaluation. VoCol (Niklas Petersen *et al.*) is another tool integrated with Github for a collaborative approach. The tool suffers from strictness and lack of freedom on the generated output, while OnToology provides full control over the generated output. Finally, WebProtégé is a web-based ontology editor for the collaborative development of ontologies. WebProtégé is focused on the implementation of the ontologies, which can only be edited online. In OnToology, the creation can be done offline as well. WebProtégé also lacks the evaluation of ontologies like the one provided by OOPS!, which is also used by OnToology.

5 CONCLUSIONS AND FUTURE WORK

In this demo we have introduced OnToology, a tool for improving the ontology development lifecycle in collaborative environments. OnToology helps documenting, depicting, evaluating and publishing. As future work, we are currently working on addressing automatic deployment and archival of the ontology releases under demand.

ACKNOWLEDGEMENTS

This research has been funded by the project “4V: Volumen, Velocidad, Variedad y Validez en la Gestin Innovadora de Datos” (TIN2013-46238-C4-2-R).

REFERENCES

Poveda-Villalón, M., Gómez-Pérez, A., and Suárez-Figueroa, M. C. (2014). Oops!(ontology pitfall scanner!): An on-line tool for ontology evaluation. *International Journal on Semantic Web and Information Systems (IJSWIS)*, **10**(2), 7–34.

Suárez-Figueroa, M.C.; Gómez-Pérez, A; Motta, E.; Gangemi, A. (2012). *The NeOn Methodology for Ontology Engineering*.

Peroni, S., Shotton, D., Vitali, F. (2012). *Easy Vocabulary Publishing. Proceedings of the I-SEMANTICS 2012 Posters & Demonstrations Track, pp. 63-67, 2012*

Cosmin Basca, Stéphane Corlosquet, Richard Cyganiak, Sergio Fernández and Thomas Schandl(2008). *Neologism: Easy Vocabulary Publishing*.

Niklas Petersen, Lavdim Halilaj, Christoph Lange, and Sören Auer. *Neologism: Easy Vocabulary Publishing*.