Automatic Support for Formative Ontology Evaluation [*]

Viktoria Pammer
Know-Center, Austria
vpammer@know-
center.at

Chiara Ghidini
FBK-irst, Trento, Italy
ghidini@fbk.eu

Marco Rospocher
FBK-irst, Trento, Italy
rospocher@fbk.eu

Luciano Serafini
FBK-irst, Trento, Italy
serafini@fbk.eu

Stefanie Lindstaedt
Know-Center and KMI TU
Graz, Austria
slind@know-center.at

ABSTRACT
Just as testing is an integral part of software engineering, so is ontology evaluation an integral part of ontology engineering. We have implemented automated support for formative ontology evaluation based on the two principles of i) checking for compliance with modelling guidelines and ii) reviewing entailed statements in MoKi, a wiki based ontology engineering environment. These principles exist in state of the art literature and good ontology engineering and evaluation practice, but have not so far been widely integrated into ontology engineering tools.

1. INTRODUCTION
State of the art ontology evaluation practice relies on guidelines and best practices in ontology engineering such as [7, 8], on ontology evaluation methodologies such as competency questions [12], and on reasoning to detect logical inconsistencies. The work we present here follows up on such existing work by automatically checking an ontology in progress for *compliance with modelling guidelines* to detect potential modelling errors and motivating ontology engineers to *review entailed statements* throughout the modelling process in MoKi, a wiki based ontology engineering tool [6, 11] that has recently been released as open-source.

Through integrating such support for ontology evaluation directly into an ontology engineering tool, ontology evaluation can finally become formative, since feedback for potential improvement or review is given in the same "place" where ontology engineering happens. In this regard, formative ontology evaluation is inherently different from ontology evaluation metrics that aim to measure an ontology's characteristics only when it is regarded as "finished enough" to merit evaluation.
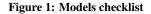
2. COMPLIANCE WITH GUIDELINES
Modelling guidelines provide guidance to the modellers during the ontology construction process but do not impose strict constraints on the ontology engineer. Hence, checking the compliance of an ontology to modelling guidelines can be indicative only of *potential* modelling errors. For instance, a typical modelling guideline is to verbally describe model elements (concepts, roles and to a certain extent also individuals) and document design decisions. While it is impossible with the current state of the art to automatically determine how good a description really is, it is possible to automatically check for model elements that are not documented at all. In MoKi, a models checklist page (Fig. 1) lists modelling guidelines, and for each guideline those model elements (concepts, properties, individuals) that do not comply with the guideline. A quality indicator visualises the "degree" to which a single model element complies to the whole set of modelling guidelines (Fig. 2). Such a functionality is not available in comparable ontology engineering environments.

Interviews with ontology engineers who have used the a prior version of the models checklist to iteratively refine and improve their ontologies indicate that such a functionality indeed supports the modelling activity. The models checklist was also deemed to be helpful in evaluating the remaining amount of work by giving an overview of the "status" of the model [1].

Figure 1: Models checklist

Figure 2: Quality indicator on a concept page.

Figure 3: Assertional effects are displayed on a concept page after it has been edited (effects are enlarged on the picture).

3. REVIEWING LOGICAL ENTAILMENTS

A key benefit of using a logically grounded language such as the Web Ontology Language OWL [2] for specifying an ontology is the possibility to automatically reason over such an ontology. The associated drawback is of course, that the larger and more complex the ontology, the more difficult it becomes for a single ontology engineer to keep an overview over whether statements that logically follow from the ontology are true.

State of the art ontology engineering tools such as Protégé and the NeOn toolkit therefore contain the functionality to list entailed statements provide explanations for them [4, 5]. A similar functionality in MoKi is called *ontology questionnaire*. While it does not technically extend state of the art, its integration into MoKi's user interface puts an emphasis on motivating the ontology engineers to review logical entailments and act on them if the find they do not agree with them. For instance, instead of "Entailed statements" or similar, the ontology questionnaire functionality is called "Inferences - Do You Agree?". The ontology questionnaire's user interface has been redesigned following the feedback on a prior version (not integrated in MoKi) described in [9].

It is also possible to consider the dynamics of an ontology, i.e. to follow the changes made to an ontology and to feedback the logical consequences of the changes to the ontology engineer. When only the terminological axioms in an ontology are considered, such considerations are made under the name of "conservative extensions" in description logics [3]. Analogously, it is possible to look for consequences on data, i.e. to ask "If new statements about concepts and roles are added/removed, how does this affect individuals in the ontology?" (assertional effects, see [10]). As an example, consider a knowledge base about the academic world. The knowledge base contains the fact that "EKAW 2010 is a conference". An ontology engineer formalises the knowledge that conferences are a particular kind of event, and that conferences produce proceedings. (S)He adds the statements "Every conference is an event" and "Every conference outputs only proceedings". Assertional effects of these changes are the facts that "EKAW2010 is a conference" and "EKAW2010 outputs only proceedings" (see Fig. 3 for how this is displayed in MoKi). Such effects are displayed in MoKi directly after the ontology is changed. The assertional effects functionality in MoKi therefore makes ontology evaluation dynamic, by pointing out potentially interesting inferences directly after they are gained (or lost, when statements are removed). Such a functionality is not available in comparable ontology engineering environments.

4. CONCLUSION

In this work we describe the integration of two state of the art principles for formative ontology evaluation into MoKi. Integration of ontology evaluation functionalities in ontology engineering tools is, we believe, a prerequisite for ontology evaluation to become formative, which again is necessary for an ontology engineering process to become more iterative, more lively and thus more prone to support the evolutionary engineering of ontologies.
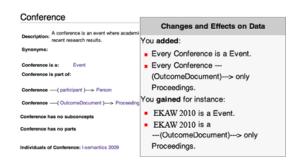
5. REFERENCES

[1] APOSDLE Deliverable 1.6. Integrated modelling methodology version 2, April 2009.

[2] B. Cuenca Grau, I. Horrocks, B. Motik, B. Parsia, P. Patel-Schneider, and U. Sattler. OWL 2: The next step for OWL. *Journal of Web Semantics*, 6(4):309–322, Nov. 2008.

[3] S. Ghilardi, C. Lutz, and F. Wolter. Did I damage my ontology? A case for conservative extensions in description logics. In *Proceedings of KR2006: the 20th International Conference on Principles of Knowledge Representation and Reasoning*, pages 187–197. AAAI Press, 2006.

[4] M. Horridge, B. Parsia, and U. Sattler. Explanation of OWL entailments in protege 4. In *International Semantic Web Conference (Posters & Demos)*, volume 401 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.

[5] Q. Ji. RaDON - Repair and diagnosis in ontology networks, http://www.neon-toolkit.org/wiki/RaDON. Last visited: 2010-07-28.

[6] MoKi - The MOdelling WiKI. http://moki.fbk.eu. Last visited: 2010-07-28.

[7] N. F. Noy and D. L. McGuinness. Ontology development 101: A guide to creating your first ontology. Technical report, Stanford Knowledge Systems Laboratory and Stanford Medical Informatics, 2001.

[8] Ontology Design Patterns. http://ontologydesignpatterns.org. Last visited: 2010-07-28.

[9] V. Pammer and S. Lindstaedt. Ontology evaluation through assessment of inferred statements: Study of a prototypical implementation of an ontology questionnaire for OWL DL ontologies. In *Knowledge Science, Engineering and Management, Third International Conference, KSEM 2009*, number 5914 in LNAI, pages 394–405. Springer, 2009.

[10] V. Pammer, L. Serafini, and S. Lindstaedt. Highlighting assertional effects of ontology editing activities in OWL. In *Proceedings of the 3rd International Workshop on Ontology Dynamics, (IWOD 2009), collocated with ISWC2009*, volume 519. CEUR Workshop Proceedings, 2009.

[11] M. Rospocher, C. Ghidini, V. Pammer, L. Serafini, and S. Lindstaedt. MoKi: The Modelling Wiki. In *Proceedings of the Forth Semantic Wiki Workshop (SemWiki 2009), co-located with ESWC 2009*, volume 464 of *CEUR Workshop Proceedings*, pages 113–128, 2009.

[12] M. Uschold and M. Grüninger. Ontologies: Principles, methods, applications. In *Knowledge Engineering Review*, volume 11, pages 93–155. 1996.