Ontology Design Patterns in WebProtégé

Karl Hammar[1,2]

[1] Information Engineering Group, Jönköping University, Sweden
[2] Department of Computer and Information Science, Linköping University, Sweden
karl.hammar@ju.se

Abstract. The use of Ontology Design Patterns (ODPs) in ontology engineering has been shown to have beneficial effects on the quality of developed ontologies, and promises increased interoperability of those same ontologies. Unfortunately, the lack of user-friendly integrated ODP tooling has prevented the adoption of pattern use. This paper demonstrates an extension to the WebProtégé ontology engineering environment supporting the finding, specialisation, and integration of ODPs. The extension combines existing approaches with new developments in ODP search, specialisation strategies, and alignment[3].

1 Introduction

Content Ontology Design Patterns (ODPs) were introduced by Gangemi [4] and Blomqvist & Sandkuhl [2] in 2005, as a means of simplifying ontology development. ODPs are intended to guide non-expert users, by packaging best practices into reusable blocks of functionality, to be adapted and specialised by those users in individual ontology development projects. Presutti et al.[8] defines a typology of ODPs, including patterns for reasoning, naming, transformation, etc. The most common type of pattern, which the author in this paper will subsequently intend when using the ODP abbreviation, are Content Patterns. A Content Pattern can be considered roughly analogous to a software design pattern, with the added benefit that it includes a reference base implementation (in the form of an OWL building block) ready for immediate customisation. Studies indicate that the use of ODPs can lower the number of modelling errors and inconsistencies in ontologies, and that they are by the users perceived as useful and helpful [1,3].

The use and understanding of ODPs have been heavily influenced by the work taking place in the NeOn Project, one result of which is the eXtreme Design (XD) ontology development method, based on ODP use. XD is influenced by the eXtreme Programming agile software development method, and like it, emphasises incremental test driven development, refactoring, and a divide-and-conquer approach to problem-solving [7]. Presutti & Gangemi [9] introduces and discusses ODP usage in XD, and lists a set of operations on ODPs, e.g., *import*,

[3] This work was partially supported by the EU FP7 project Visual Analytics for Sensemaking in Criminal Intelligence Analysis (VALCRI) under grant number FP7-SEC-2013-608142.

specialisation, and *composition*. Using ODPs in an efficient manner requires tool support for performing such operations. While tool support was also developed within the NeOn project, it was implemented as a set of plugins for NeOn Toolkit ontology IDE, which early on lost development traction. Consequently, until now, ontology engineers who wanted to use ODPs in more widely used IDEs had no appropriate support tooling available to them.

The eXtreme Design for WebProtégé (XDP) extension[4] remedies this situation by providing tool support for ODP usage in the modern collaborative ontology engineering platform WebProtégé. Additionally, XDP integrates a set of novel new developments[5]:

- **Composite search engine** - This search engine combines the results of a traditional vector search with a comparison of query and ODP competency questions based on relative Levenshtein edit distances, and a comparison of query term hypernyms to ODP concept synonyms. This method has shown to improve recall over a test dataset 3-4 times when compared to an existing state of the art ODP search engine [6].
- **ODP specialisation strategy support** - As the author has previously shown [5], the intended use of properties in ODPs can be specialised using different strategies. XDP supports specialisation using property subsumption, using existential or universal quantification restrictions, and combining both approaches.
- **ODP specialisation alignment suggestions** - XDP helps users align specialised entities with existing entities in the ontology project. This is an important and previously unsupported operation, particularly in projects without a dedicated release manager, where developers may need assistance merging their work with the existing project in a consistent manner. Alignments are suggested based on string matching techniques over entity labels and IRI fragments. Initial experimentation shows that even such simple techniques can cover over 60 % of the alignments in a real world dataset [6].

2 System Design and Features

In order to make the integration into the existing WebProtégé code base as small and maintainable as possible, the XDP architecture has been designed to consist of two loosely connected subsystems:

- Integrated in a fork of the main WebProtégé code are client UI components and the necessary server-side components needed to persist an ODP to a WebProtégé project. These components interoperate by extending the WebProtégé standard GWT-RPC-based dispatch mechanisms.

[4] Online demo: `http://wp.xd-protege.com`, video walkthrough: `https://youtu.be/ZRH6vGXocqU`, code: `https://github.com/hammar/webprotege`

[5] Note that as WebProtégé does not at the time of writing support *owl:imports*, import and specialisation has been implemented through duplicating ODP entities in the target ontology.

– Supporting these components, a REST service enables clients to search for ODPs, to browse ODPs by category, and to fetch the documentation and OWL representation of a given ODP. This back-end service queries a Lucene index built from a set of input ODPs and mapping metadata extracted from the community ODP portal[6].

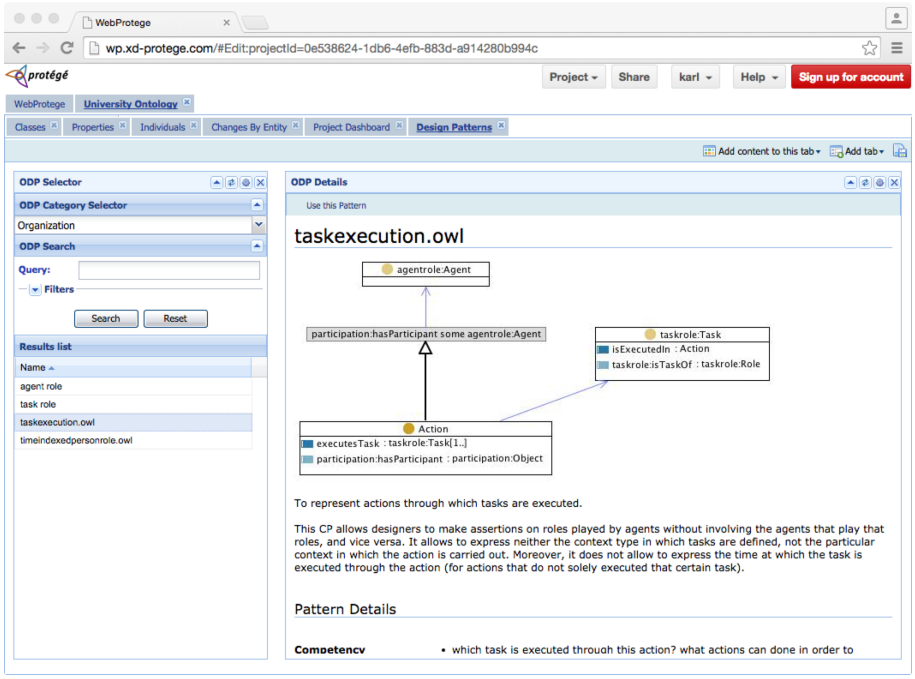

Fig. 1. eXtreme Design for WebProtégé UI

XDP's user-facing components are housed in a WebProtégé UI tab titled "Design Patterns" (see Figure 1). This tab houses an ODP Selector portlet, and an ODP Details portlet. The former provides an interface where the user can browse for ODPs by category, or search over all ODPs using an query string. Browsing or search results are displayed in the same portlet; when browsing listed alphabetically, and when searching listed by search engine confidence score.

Upon selecting an ODP from the result list, the illustration and documentation for that ODP is displayed in the ODP Details portlet. When the user has found an appropriate ODP they can run the Specialisation Wizard component, which guides them through selecting a specialisation strategy to use, specialising ODP classes and/or properties by subsumption, constraining the semantics of

[6] `http://ontologydesignpatterns.org`

property specialisations, aligning the resulting model with the existing ontology, and finally, persisting the ODP specialisation into the existing ontology project.

3 Discussion and Future Work

The XDP extension bridges an important tooling gap, by enabling the use of ODPs in real world projects by real users. This will bring tangible benefits both to practitioners doing ontology engineering work and to researchers, who will be able to use XDP as an environment in which to evaluate realistic ODP usage.

In addition to the novel features discussed in Section 1, the integration with WebProtégé allows XDP to enable truly distributed ontology engineering with ODPs, which is potentially a beneficial side effect. The XD method emphasises pair development. For small distributed teams it can be hard to motivate travel and accommodation costs just to put two people in the same room. Using XDP, those developers can collaborate on a project across geographical boundaries.

There are some important steps and operations in the XD method that still lack appropriate tool support, including requirements management and ontology testing tasks, as well as ODP composition. Based on the results of initial user evaluation of XDP in real projects, the author aims to implement functionality to support these steps, so that XDP can in the future support the entire XD ontology engineering workflow.

References

1. Blomqvist, E., Gangemi, A., Presutti, V.: Experiments on Pattern-based Ontology Design. In: Proceedings of the Fifth International Conference on Knowledge Capture. pp. 41–48. ACM (2009)
2. Blomqvist, E., Sandkuhl, K.: Patterns in Ontology Engineering: Classification of Ontology Patterns. In: Proceedings of the 7th International Conference on Enterprise Information Systems. pp. 413–416 (2005)
3. Blomqvist, E., Presutti, V., Daga, E., Gangemi, A.: Experimenting with extreme design. In: Knowledge Engineering and Management by the Masses, pp. 120–134. Springer (2010)
4. Gangemi, A.: Ontology Design Patterns for Semantic Web Content. In: The Semantic Web–ISWC 2005, pp. 262–276. Springer (2005)
5. Hammar, K.: Ontology design pattern property specialisation strategies. In: Knowledge Engineering and Knowledge Management, pp. 165–180. Springer (2014)
6. Hammar, K.: Ontology design patterns: Improving findability and composition. In: The Semantic Web: ESWC 2014 Satellite Events, pp. 3–13. Springer (2014)
7. Presutti, V., Blomqvist, E., Daga, E., Gangemi, A.: Pattern-Based Ontology Design. In: Ontology Engineering in a Networked World, pp. 35–64. Springer (2012)
8. Presutti, V., Gangemi, A., David, S., Aguado de Cea, G., Suárez-Figueroa, M.C., Montiel-Ponsoda, E., Poveda, M.: D2.5.1: A Library of Ontology Design Patterns: Reusable Solutions for Collaborative Design of Networked Ontologies. Tech. rep., NeOn Project (2007)
9. Presutti, V., Gangemi, A.: Content ontology design patterns as practical building blocks for web ontologies. In: Conceptual Modeling-ER 2008, pp. 128–141. Springer (2008)