OBM2OWL Patterns: Spotlight on OWL Modeling Versatility

Marek Dudáš, Tomáš Hanzal, Vojtěch Svátek, and Ondřej Zamazal

Department of Information and Knowledge Engineering, University of Economics, W. Churchill Sq.4, 13067 Prague 3, Czech Republic, {marek.dudas|tomas.hanzal|svatek|ondrej.zamazal}@vse.cz

Abstract. We present an experimental implementation of an ontological engineering support method that allows to see multiple OWL modeling options for a certain modeled situation. Initial style-independent background model (PURO OBM) is automatically transformed to alternative OWL fragments, which can then be visualized and chosen from.

1 Introduction

During ontology engineering in OWL, the developer can often choose different combinations of language constructs to model the same situation. The choice might be driven by the intended usage of the ontology: web markup vocabularies often favor 'feature' assignment to entities through data properties, while linked data vocabularies prefer object properties for this purpose; reasoning-enabled ontologies, in turn, express 'features' as classes. We call consistent sets of such choices *modeling styles*. Viewing different alternatives may help supporting the choice of style for a specific use case. We propose a method where a single styleindependent model – ontological background model (OBM) in PURO language [6] – is created manually, and the style-specific OWL representations are then automatically generated using OBM2OWL transformation patterns. While not being a way to author an ontology in bulk, this approach can serve as 'spotlight' on different modeling options.

PURO OBMs are based on two main distinctions: between particulars and universals and between relationships and objects (hence the PURO acronym). There are six basic entity types: B-object (particular object), B-type (type of object/type), B-relationship (particular relationship), B-relation (type of relationship), B-valuation (particular assertion of quantitative value) and B-attribute (type of valuation). A PURO OBM consists of named instances of these types, plus of *subTypeOf* and *instanceOf* relationships. It captures a concrete situation that serves as basis for analyzing existing ontologies or generating new ones.

Related research Compared to heavier-weighted OBM formalisms such as OntoUML [1], PURO is supposed to be 'mentally closer' to OWL: there are no tough notions such as 'rigidity' (as in OntoClean [3]), and its 'object-type-relation' triad corresponds to the 'individual-class-property' triad of OWL, except that 1) PURO does not limit the *arity* of relations and allows *higher-order* classes, and, consequently, 2) allows abstracting from modeling style choices that are enforced by these limitations in OWL. Generation of OWL from OntoUML has also been investigated, however without considering OWL modeling styles. On the other hand, 'pattern families' capturing alternative modeling styles have been investigated by the W3C pattern group,¹ for the 'class as property value', 'specified values' and 'n-ary relation' modeling issues, but with no reference to automatic transformation. Suitability of different OWL modeling styles was also analyzed by Dermeval et al. [2], yet in the context of feature modeling only.

Motivating example 2 If we need to describe that a book was published in a city, different ontologies propose us different types of OWL constructs:³

- data property: d:b1 o:publishedIn "Prague". (cf. RDA Registry ont.)
- *object property*: d:b1 o:publishedIn d:Prague. (cf. *purl.org/library*)
- reified relationship: d:_rr1 a o:publishedIn; o:book d:b1; o:place d:Prague. (cf. British Library Data Model)
- *dedicated class instantiation*: d:b1 a o:BookPublishedInPrague. (this kind of lexical encapsulation is common in the DBpedia category system).

Another example: to say that '025.04' is a topic in the Dewey Decimal Classification (DDC), at least two ways are possible in OWL:

- class instantiation: o:DDC_Topic rdfs:subClassOf d:Topic.
 d:025.04 a o:DDC_Topic. (cf. British Library Data Model)
- dedicated 'type' class + object property: o:ddc_topic a o:TopicType. d:025.04 o:topicType o:ddc_topic. (cf. the openlibrary.org dataset)

Rather than aligning such choices pairwise, as we did in our previous project, PatOMat [5], we propose the PURO OBM representation as an 'interlingua.'

2 PURO OBM2OWL Transformation

The OBM2OWL transformation patterns are currently implemented as SPARQL UPDATE queries, with structure based on OWL2OWL patterns from PatOMat. The SELECT part describes a fragment of OBM, e.g., a B-relationship connecting two B-objects. The INSERT part describes a corresponding OWL structure. We so far created six patterns⁴ covering rather intuitive transformations. The transformation functionality was embedded into PURO Modeler,⁵ our web-based graphical editor of OBM. Once the user creates an OBM, it can be serialized into RDF and sent to a web service that executes the SPARQL queries and returns OWL ontology fragments separated into RDF graphs. Each graph is then

¹ http://www.w3.org/2001/sw/BestPractices/OEP/

² Based on a survey from http://tomhanzal.github.io/owl-modeling-styles/.

³ The "d" prefix always refers to a fictional dataset and "o" to a fictional ontology.

⁴ Available at http://lod2-dev.vse.cz/puromodeler/patterns/.

⁵ http://lod2-dev.vse.cz/puromodeler

visualized using WebVOWL [4] and offered for download. Although the OBM is created as example of a concrete situation, incl. sample instances, the resulting OWL fragments only consist of the T-box.

Consider, again, a designer of a bibliography ontology who wants to model book topics and place of publishing. In PURO Modeler s/he creates/opens the OBM of an example situation with *book1* about DDC topic 025.04 published in *Prague* (Fig. 1). The tool serializes the OBM into an RDF graph⁶ and returns four alternative OWL ontology fragments from Fig. 2, which combine the style options described in Section 1. In fragment A, object properties are used where possible: even the topic type assignment is modeled as a property (*topic_hasType*). In B, both relationships are reified (*hasTopic_Relationship*, *publishedIn_Relationship*) and the topic type assignment becomes class instantiation. C is similar to A, but topic type assignment is modeled as in B. In D, all relationships are modeled as data properties; the topic type is omitted since the topic is now a literal. The designer can now choose the most suitable OWL fragment, download it and integrate into the ontology using his/her favorite editor.

In the future, generation of OWL fragments could be restricted to those complying with best practices; e.g., lexical encapsulation (*BookPublishednInPrague*) could be omitted as suboptimal, requiring a new class for every location. For some OBM structures the number of variants will also be smaller by definition, e.g., an n-ary relation with n>2 cannot be modeled as plain OWL property.

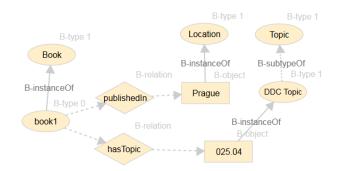


Fig. 1. Example of OBM describing where a book was published and its topic.

3 Conclusions and Future Work

The presented approach of ontology style bootstrapping from an OBM is still in its infancy and the initial set of OBM2OWL patterns is merely tentative; yet, thanks to tool availability, feedback can already be collected. The longterm vision is that of a portal (integrated with *ontologydesignpatterns.org*) where

⁶ Available at http://lod2-dev.vse.cz/puromodeler/bookLocationTopicOBM.rdf

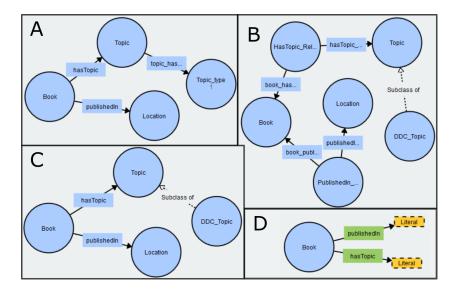


Fig. 2. Four variants of OWL ontology fragments generated from the OBM on Fig. 1.

experts trained in OBM ('PURO writers') could build and share the OBMs while novice ontological engineers ('PURO readers') could pick fragments of the OBMs and let the OBM2OWL service generate the corresponding OWL fragments for them. Systematic elaboration and analysis of advantages and disadvantages of the modeling styles is planned for future research. Use of lexicographic resources to generate sound labels for new elements is also foreseen.

The research is supported by UEP IGA F4/90/2015 and by long-term institutional support of research activities by Faculty of Informatics and Statistics, Univ. of Economics, Prague. Ondřej Zamazal is supported by CSF 14-14076P.

References

- Benevides, A.B., Guizzardi, G.: A model-based tool for conceptual modeling and domain ontology engineering in OntoUML. In: Enterprise Information Systems, pp. 528–538. Springer (2009)
- Dermeval, D., Tenório, T., et al.: Ontology-based feature modeling: An empirical study in changing scenarios. Expert Systems with Applications 42(11) (2015)
- 3. Guarino, N., Welty, C.A.: An overview of ontoclean. In: Handbook on ontologies, pp. 201–220. Springer (2009)
- Lohmann, S., Negru, S., Haag, F., Ertl, T.: VOWL 2: user-oriented visualization of ontologies. In: EKAW, pp. 266–281. Springer (2014)
- Šváb-Zamazal, O., Dudáš, M., Svátek, V.: User-friendly pattern-based transformation of OWL ontologies. In: EKAW, pp. 426–429. Springer (2012)
- Svátek, V., Homola, M., et al.: Metamodeling-based coherence checking of OWL vocabulary background models. In: OWLED. CEUR (2013)