Generating Semantic Media Wiki Content from Domain Ontologies

Dominik Filipiak[1,2] and Agnieszka Ławrynowicz[1]

Institute of Computing Science, Poznan University of Technology, Poland
Business Information Systems Institute Ltd., Poland

Abstract. There is a growing interest in holistic ontology engineering approaches that involve multidisciplinary teams consisting of ontology engineers and domain experts. For the latter, who often lack ontology engineering expertise, tools such as Web forms, spreadsheet like templates or semantic wikis have been developed that hide or decrease the complexity of logical axiomatisation in expressive ontology languages. This paper describes a prototype solution for an automatic OWL ontology conversion to articles in Semantic Media Wiki system. Our implemented prototype converts a branch of an ontology rooted at the user defined class into Wiki articles and categories. The final result is defined by a template expressed in the Wiki markup language. We describe tests on two domain ontologies with different characteristics: DMOP and DMRO. The tests show that our solution can be used for fast bootstrapping of Semantic Media Wiki content from OWL files.

1 Introduction

There is a growing interest in holistic ontology engineering approaches [1]. Those approaches use various ontological as well as non-ontological resources (such as thesauri, lexica and relational DBs) [2, 3] . They also involve multidisciplinary teams consisting of ontology engineers as well as non-conversant in ontology construction domain experts. Whilst an active, direct involvement of the domain experts in the construction of quality domain ontologies within the teams appears beneficial, there are barriers to overcome for such involvement to be effective. Those are mostly related to the high complexity of logic-based ontology modeling languages such as OWL[1].

In order to remove the barriers, various tools have been developed that hide or decrease the complexity of logical axiomatisation in expressive ontology languages. Among these tools are Web forms [4], spreadsheet like templates [5] and semantic wikis [6–9]. Recent works have shown that domain experts may be effectively involved in using such tools for knowledge gathering stage of ontology development when the core structure of the ontology is already established [5]. Recent works have also shown that ontology modeling tools based on wikis can contribute to collaboration between ontology engineering experts and domain experts [10].

The aim of this work is to deliver a solution for transformation of OWL files to Semantic Media Wiki (SMW)[2][6] content. By transformation we mean an automatic

[1] `http://www.w3.org/TR/owl2-overview/`
[2] `http://semantic-mediawiki.org`

conversion of these files to Wiki articles and category pages. The final result is defined by a template written in the Wiki markup language. The purpose is to bootstrap a collaboration between ontology stakeholders & engineers and a wider community of researchers in constructing a domain ontology once a core structure of the ontology is established.

The rest of this paper is structured as follows. In Section 2 we discuss the work related to ours. In Section 3, we present a solution for transforming OWL files to Semantic Media Wiki content that is based on SPARQL and user defined Wiki templates. In Section 4 we present a simple evaluation of the implemented solution with two ontologies: DMOP and DMRO. Section 5 contains the discussion, and in Section 6 we conclude.

2 Related Work

In [11] an SMW extension consisting of a solution for transformation of ontological knowledge was presented. The focus was on transforming instance data (ABox), where the user could select a subset of instance assertions to import into SMW, and simple schema information. The more expressive ontology model was considered as an external source of knowledge, providing constraints into the domain model stored in the SMW installation. The paper discussed several use cases including coordinating a project team within a company and bootstrapping the contents and vocabulary of the semantic wiki of a conference system.

The Halo extension to SMW, contained in SMW+[3], was developed in order to facilitate the use of Semantic Wikis for large communities of users and thus consisted of a toolset for increasing the ease of use of SMW features. Among the tools, it provided an import and export functionality for OWL ontologies. Currently, the Halo extension is unmaintained.

MoKi [8] is a tool based on SMW that extends SMW by offering specific support for enterprise modelling. It provides support for domain experts in modeling business domains (domain ontologies) and simple processes (process models). MoKi offers a functionality to upload in MoKi an existing domain ontology modeled in OWL. This import functionality generates a MoKi page for each concept, property and individual from the ontology. The templates for these ontology entities are automatically filled based on the axioms modeled in the ontology such as, for example, is-a relation between concepts, domain and range of properties, and individuals being members of concepts.

The author of [12] presents a solution based on OWL Wiki Forms (a Semantic MediaWiki extensions that map Semantic Web ontologies to a Semantic Forms-based semantic wiki) and Fresnel (an ontology for specifying browsing interfaces for Semantic Web data). The solution consists of a mapping from any ontology to Fresnel style data and from Fresnel data to form-based semantic wikis. A technique for automatic generation of Fresnel lenses triples from given ontologies is presented, where the Fresnel lenses triples define a default target interface for data using those ontologies. It is also possible for the user to define custom Fresnel that can cascade over the default interface style, similarly as it is done in CSS.

[3] `http://semantic-mediawiki.org/wiki/Semantic_MediaWiki_Plus`

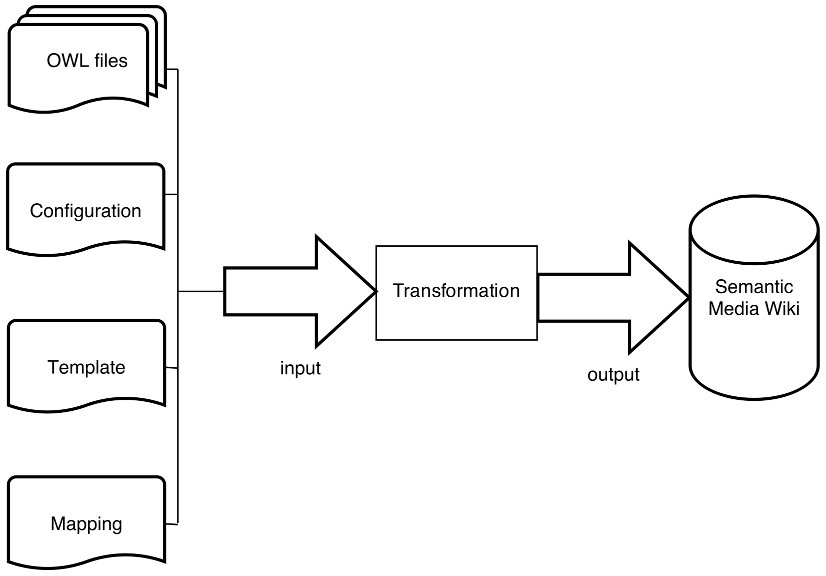


Fig. 1. Input and output of the proposed algorithm.

Our presented solution allows for a direct mapping from an ontology to user defined Wiki templates. In such way, it allows for a direct transformation from arbitrary (user selected) ontology entities to arbitrary Media Wiki form elements (taking into account the distinction between OWL entity types).

3 Approach

3.1 Overview

Figure 1 illustrates the general idea of our approach. The input to our transformation algorithm is a set of at least four files: an OWL file, a configuration file, a template file and a mapping file. There can be one or more OWL files. The configuration file contains information about Semantic Media Wiki adress, login and password. Moreover, it points out to the *root class* - it is an URI which defines which part of an ontology should be processed. The template file, based on Wiki markup syntax, defines how created pages should look like by specifying attributes URIs. Finally, the mapping file connects template and OWL files by assigning variables to entity URIs. Processing these files should result in an updated set of Semantic Media Wiki articles, including article pages, category pages and a template page.

3.2 Design Choices

The design issues, with which we had to deal, concerned, among others, selecting categories and attributes, and representing many attributes for a single entity. By an attribute we mean a Wiki counterpart of a single OWL property describing a given entity, which is presented mostly in an *infobox* in our templates.

We can describe hierarchy in OWL files as a set of classes, which can be related with each other by child (subclass), parent (superclass) or equivalent class relation. However, Media Wiki articles can be shown as a directed acyclic graph, where each edge represents child/parent relation, according to its direction. Hence, we decided to treat equivalent (with additional constraints) classes as child classes. Entities described as *Named Individuals* are candidates for articles. Similarly, classes with *subClassOf*, *equivalentClass* or *rdf:type* attribute are categories stubs. Due to possibility of numerous attributes, we decided to extract only these which are declared by the user in a template file. Wiki markup syntax does not allow to make implicit declaration of a set of values for one attribute. It cannot predict the number of the values for a given attribute without any additional configuration or modification. Hence, we used Semantic Forms[4] with parser functions as a solution to this problem. The design choices for the transformation are listed in Table 1.

Table 1. Ontology to Semantic MediaWiki transformation choices.

Issue	**Solution**
Article categorization	Either equivalent classes (*owl:equivalentClass*) or parent classes (*rdfs:subClassOf*) are chosen for parent articles of a category
Selection of entities for articles	Resources classified as *owl:NamedIndividual* are chosen for ordinary articles
Selection of entities for category pages	Resources classified as *owl:equivalentClass* or *rdfs:subClassOf* with respect to a base category are chosen for category pages
Selection of properties	Properties for transformation are defined by the user in the template file and in the mapping file
Multiple values for one attribute	Semantic Forms with parser functions are used

3.3 Algorithm

Our approach is described by Algorithm 1. It is necessary to have all mentioned input files in order to define the configuration. To begin the transformation process all classes and individuals which are in a child relation with the root class (including transitivity) have to be found. It can be done by SPARQL queries. We present our solution for classes in Listing 1.1 and for individuals in Listing 1.2. The list of articles stubs is prepared from each acquired entity. Filling all stubs with content from classes and individuals is based on the template and the mapping. In order to do this the ontology has to be searched using SPARQL queries one more time. Article/Category title is based on entity URI. Since there is a requirement for each article to have a title, blank nodes are omitted. The last step consists of transferring the obtained data to Semantic Media Wiki.

[4] `http://www.mediawiki.org/wiki/Extension:Semantic_Forms`

```
Data: $OWLfiles$, $configuration$, $template$, $mapping$
Result: An updated set of Semantic Media Wiki articles
load $OWLfiles$;
$articles \leftarrow$ process $OWLfiles$;
for $article$ in $articles$ do
    if $article$ is a template then
        prepare $template$
    end
    if $article$ is a category or an article then
        set a title to $template$;
        fill $article$ with data corresponding to $template$ and $mapping$;
        add category footer to $article$;
    end
end
Connect to Semantic Media Wiki;
for $article$ in $articles$ do
    save $article$ to Semantic Media Wiki;
end
```

Algorithm 1: OWL files conversion to Semantic Media Wiki articles

Listing 1.1. SPARQL query for finding classes

```
SELECT DISTINCT ?subclass
WHERE {
  ?subclass ((owl:equivalentClass/owl:intersectionOf/rdf:
     ↪ rest*/rdf:first)|rdfs:subClassOf)+ <rootClass>.
}
```

Listing 1.2. SPARQL query for searching for individuals

```
SELECT DISTINCT ?individual
WHERE {
?individual rdf:type ?type.
?type ((owl:equivalentClass/owl:intersectionOf/rdf:rest*/
   ↪ rdf:first)|rdfs:subClassOf|rdf:type)+ <rootClass>.
FILTER (isURI(?individual) && !isBLANK(?individual)).
}
```

4 Evaluation

4.1 Materials

We prepared an application[5] written in Java, which implements the described algorithm. We used Apache Jena[6] to read and query OWL files. Tests of the transformation were performed with two different ontologies - *DMOP* and *DMRO*.

DMOP is an abbreviation for the Data Mining OPtimization Ontology [13, 14]. The ontology is focused on description of numerous data mining algorithms and their charateristics. The primary goal of DMOP is to support making decisions at each step of data mining process which determines the outcome of the process. DMOP is richly axiomatised–it uses almost all features of OWL 2 DL. Majority of DMOP entities are its 'own' entities defined in DMOP's namespaces. Moreover, it imports a part of DOLCE foundational ontology. DMOP has been successfuly used for meta-mining within the Intelligent Discovery Assistant (comprised of an AI planner and semantic meta-miner) that is deployed in the data mining environment RapidMiner.

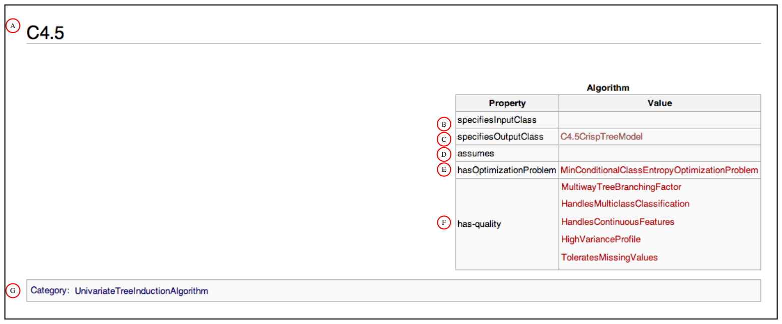
DMRO, a Digital Multimedia Repositories Ontology [15], has different characteristics than DMOP. It was constructed as a lighweight ontology network using NeoN methodology [2, 3] and various ontology design patterns. The main file imports several ontology modules describing: multimedia resources, users, events, reviews, Web Usage Mining related concepts, and the domain topics. The modules re-use various ontologies and vocabularies such as Dublin Core[7], FOAF[8], RDF Review[9], OBO Relation Ontology[10], and OAI-ORE[11].

4.2 Results

We made simple tests consisting of transforming chosen branches of the ontologies to Semantic Media Wiki.

In case of DMOP, we transformed all classes with related entities, which are in a child relation with *DM-Algorithm* class (examples in Listings 1.3 and 1.4). The sample Wiki article about *C4.5* algorithm (named individual in DMOP) is shown in Figure 2. The article's title results from the URI of the individual, which is shown in Listing 1.3 (*rdf:about* attribute) and marked as A in Figure 2). Attributes are labeled as B, C, D, E in the same figure. Due to lack of information in the ontology files not all of infobox values are filled up (they are marked as B and D on Figure 2). Thanks to Semantic Forms, *has-quality* attribute (marked as F in the figure) has multiple values, what was implicitly declared in the template. Class membership is labeled by G.

[5] `https://github.com/mimol/owl2wiki`
[6] `https://jena.apache.org`
[7] `http://dublincore.org/documents/dcmi-terms/`
[8] `http://xmlns.com/foaf/spec/`
[9] `http://vocab.org/review/terms.html`
[10] `http://obofoundry.org/ro/`
[11] `http://www.openarchives.org/ore/`



C4.5

Algorithm

Property	Value
specifiesInputClass	
specifiesOutputClass	C4.5CrispTreeModel
assumes	
hasOptimizationProblem	MinConditionalClassEntropyOptimizationProblem
has-quality	MultiwayTreeBranchingFactor HandlesMulticlassClassification HandlesContinuousFeatures HighVarianceProfile ToleratesMissingValues

Category: UnivariateTreeInductionAlgorithm

Fig. 2. The result of DMOP ontology transformation

Listing 1.3. Structure of a sample DMOP entity classified as an article

```
<owl:NamedIndividual rdf:about="&PD;C4.5">
 <DOLCE-Lite:has-quality rdf:resource="&DMOP;
      ↪ HighVarianceProfile"/>
  (...)
 </owl:NamedIndividual>
```

Listing 1.4. Structure of a sample DMOP entity classified as a category

```
<owl:Class rdf:about="&DMOP;SomeClass">
 <rdfs:subClassOf rdf:resource="&DMOP;SomeSubClass"/>
 </owl:Class>
```

We also successfully conducted another experiment with DRMO whose structure differs from DMOP's. Although category entities are similarly declared explicitely as OWL classes (as shown in Listing 1.6), the individuals are not declared as named individuals, but are declared as members of *owl:Thing* (Listing 1.5). Nevertheless, we were able to transform a branch rooted at *DMRO:Event*.

Listing 1.5. Structure of example DMRO entity classified as an article

```
<owl:Thing rdf:about="#Event2305">
 <rdf:type rdf:resource="&DMRO-Event;EventSection"/>
  (...)
 </owl:Thing>
```

Listing 1.6. Structure of example DMRO entity classified as a category

```
<owl:Class rdf:about="&dul;Event">
 <rdfs:subClassOf rdf:resource="&dul;Entity"/>
</owl:Class>
```

5 Discussion

We tested our solution with ontologies having different characteristics. Since SPARQL engines by default are not supposed to perform reasoning, SPARQL may turn very structure-sensitive. Although, in the OWL serialization that we used the structure of category entity is based on *owl:Class* in both, DMOP and in DMRO (Listings 1.4 and 1.3, respectively), there are differences in individual selection. In DMOP, individuals are explicitly declared as a *owl:NamedIndividual*, while in DMRO they are not. In the latter case, the entities we classify as individuals are instances of *owl:Thing*. That is why it is important to consider possible cases and take them into account in SPARQL queries or transform OWL files to a canonical representation before SPARQL is applied to query them.
Alternatively, as a more standard solution, we could use an API for handling ontologies like OWL API or Jena ontology API. However, while designing our SPARQL-based solution we kept in mind that it can be further flexibly extended to transform remote (linked) data from SPARQL endpoints to Semantic Media Wiki content.

Our main motivation is the real need for Wiki based tools in the context of such portals as *DMO Foundry* (`http://www.dmo-foundry.org`) or *OpenML* (`http://openml.org`). Using the presented in this paper preliminary solution we have generated content that is a human-readable, structured and organised knowledge base. We envisage that it could be used by such domain users as researchers trying to find out which algorithm would match their expectations or even students during classes.

6 Conclusions

In this paper, we have presented a prototype solution for transformation of OWL ontologies to Semantic Media Wiki content. The solution is based on a mapping between selected ontology entities and user-defined Wiki templates, and on using SPARQL. We have successfuly applied our implemented prototype to two different ontologies: DMOP and DMRO.

Despite of describing a working prototype, we still consider this research as a work in progress. In future work, we plan to consider ideas for cascading templates (similarly to the work of [12]) and text mining (especially named entity recognition). We also plan to extend our solution by support for exporting knowledge from the Wiki to OWL ontologies. Finally, our plans are to use the solution in real world use cases like within data mining portals such as DMO Foundry or OpenML to provide Wiki based tools for community of researchers in data mining or in other disciplines. We plan to test the prototype in such settings, where the collaboration between ontology engineers and normal users will be investigated. We plan to conduct a case study to investigate how the collaboration could be improved based on our technical solution.

References

1. Denaux, R., Dolbear, C., Hart, G., Dimitrova, V., Cohn, A.G.: Supporting domain experts to construct conceptual ontologies: A holistic approach. Web Semantics: Science, Services and Agents on the World Wide Web **9**(2) (2011)

2. Gómez-Pérez, A., Suárez-Figueroa, M.: Scenarios for building ontology networks within the NeOn methodology. In: K-CAP. (2009) 183–184
3. Suárez-Figueroa, M.: NeOn Methodology for Building Ontology Networks: Specification, Scheduling and Reuse. PhD thesis, Universidad Politécnica de Madrid, Spain (2010)
4. Tudorache, T., Nyulas, C., Noy, N.F., Musen, M.A.: WebProtege: A collaborative ontology editor and knowledge acquisition tool for the web. Semant. web **4**(1) (January 2013) 89–99
5. Jupp, S., Horridge, M., Iannone, L., Klein, J., Owen, S., Schanstra, J., Wolstencroft, K., Stevens, R.: Populous: a tool for building OWL ontologies from templates. BMC Bioinformatics **13**(S-1) (2012) S5
6. Krötzsch, M., Vrandečić, D., Voelkel, M.: Semantic MediaWiki. In Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L., eds.: The Semantic Web - ISWC 2006. Volume 4273 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2006) 935–942
7. Auer, S., Dietzold, S., Riechert, T.: OntoWiki – a tool for social, semantic collaboration. In: Proceedings of the 5th International Conference on The Semantic Web. ISWC'06, Berlin, Heidelberg, Springer-Verlag (2006) 736–749
8. Ghidini, C., Kump, B., Lindstaedt, S., Mahbub, N., Pammer, V., Rospocher, M., Serafini, L.: MoKi: The enterprise modelling wiki. In Aroyo, L., Traverso, P., Ciravegna, F., Cimiano, P., Heath, T., Hyvnen, E., Mizoguchi, R., Oren, E., Sabou, M., Simperl, E., eds.: The Semantic Web: Research and Applications. Volume 5554 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2009) 831–835
9. Nalepa, G.: Loki – semantic wiki with logical knowledge representation. In Nguyen, N., ed.: Transactions on Computational Collective Intelligence III. Volume 6560 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2011) 96–114
10. Di Francescomarino, C., Ghidini, C., Rospocher, M.: Evaluating wiki collaborative features in ontology authoring. IEEE Transactions on Knowledge and Data Engineering **(to appear)** (2014)
11. Vrandečić, D., Krötzsch, M.: Reusing ontological background knowledge in semantic wikis. In Völkel, M., Schaffert, S., Decker, S., eds.: 1st Workshop on Semantic Wikis. Number 206 in CEUR Workshop Proceedings, Aachen (2006)
12. Rutledge, L.: From ontology to wiki generating cascadable default fresnel style from given ontologies for creating semantic wiki interfaces. In: Workshop on Semantic Web Collaborative Spaces (SWCS2013). (2013)
13. Hilario, M., Nguyen, P., Do, H., Woznica, A., Kalousis, A.: Ontology-based meta-mining of knowledge discovery workflows. In: Meta-Learning in Computational Intelligence. Volume 358 of Studies in Computational Intelligence. Springer (2011) 273–315
14. Keet, C.M., Lawrynowicz, A., d'Amato, C., Hilario, M.: Modeling issues, choices in the Data Mining OPtimization Ontology. In Rodriguez-Muro, M., Jupp, S., Srinivas, K., eds.: OWLED. Volume 1080 of CEUR Workshop Proceedings., CEUR-WS.org (2013)
15. Lawrynowicz, A., Palma, R.: Applications of ontology design patterns in the transformation of multimedia repositories. In Blomqvist, E., Gangemi, A., Hammar, K., Suárez-Figueroa, M.C., eds.: WOP. Volume 929 of CEUR Workshop Proceedings., CEUR-WS.org (2012)