Visualizing and Editing Ontology Fragments with OWLGrEd

Renars Liepins[**], Karlis Cerans[*], Arturs Sprogis[**]

Institute of Mathematics and Computer Science, University of Latvia
{ Renars.Liepins, Karlis.Cerans, Arturs.Sprogis}@lumii.lv

Abstract. The OWLGrEd ontology editor allows graphical visualization and authoring of OWL 2.0 ontologies using a compact yet intuitive presentation that combines UML class diagram notation with textual Manchester syntax for class expressions. Here we show, how to integrate OWLGrEd with ontology module mechanism from OWL API to obtain on-demand ontology fragment visualization that is essential for many existing large ontologies that do not fit in a single reasonably perceivable UML class diagram.

Keywords: OWL ontologies, visualization, UML/OWL profile, OWLGrEd, ontology decomposition, ontology modules

1 Introduction

Intuitive ontology visualization is a key for their learning, exchange, as well as their use in conceptual modeling and semantic database schema design. A number of tools and approaches exist for rendering and/or editing OWL [1] ontologies in a graphical form, including UML Profile for OWL DL [2], ODM [3], TopBraid Composer [4], Protégé [5] plug-in OWLViz [6] and OWLGrEd [7,8]. The approaches of [2,3,7,8] use UML [9] class diagrams to visualize OWL ontologies. This is achieved by visualizing an independent hierarchy of ontology classes and then structuring the data and object property visualizations along the property domain and range classes. The OWL ontology constructions not having direct UML counterparts (e.g. class expressi-ons, properties with more than one domain assertion, sub-property relations etc.) are usually handled by some auxiliary means in the notation and the editor. OWLGrEd uses textual OWL Manchester syntax [10] for class expressions where the graphical notation is not available or is not desired thus allowing compact and comprehensible presentation of up to medium-sized ontologies within a single diagram.

The main focus of this demo is on using the compact UML-style notation, offered by OWLGrEd, on large ontologies that do not fit within any reasonably-sized class diagram, or whose rendering appears to be too complicated due to a kind of "spider web" effect produced by many classes and relations. Its key idea consists in splitting the ontology into meaningful fragments of conceivable size and then visualizing each of the fragments in a separate diagram .

[**] Partially supported by Latvian 2010.-2013. National Research Program Nr.2 project Nr.5.
[*] Partially supported by European Union via European Regional Development Fund project 2011/0009/2DP/2.1.1.1.0/10/APIA/VIAA/112.

OWLGrEd already has the means to partition ontology into sub-diagrams (fragments) when authoring or reengineering an existing ontology. But there was no way to automatically partition an ontology that is imported into OWLGrEd for visualization. In this demo we will present an extension to OWLGrEd visualization capabilities, that allows automatic partitioning of an ontology into logical fragments. The addition is based on Automatic Decomposition [11] that was recently implemented in the OWL API[1]. The decomposition is based on signatures, i.e. for each fragment a user selects some entities that should be included in the fragment. Then the fragment is extended with all the logically relevant axioms for these entities. Finally all the fragments are rendered graphically in the OWLGrEd editor.

The demonstration shows (i) working with OWLGrEd tool to render and author OWL ontologies (ii) OWLGrEd extension to automatically partition ontology into logical overlapping fragments based on fragment signatures.

2 OWLGrEd Notation and Editor

OWLGrEd[1] provides a complete graphical notation for OWL 2 [1], based on UML class diagrams. We visualize OWL classes as UML classes, data properties as class attributes, object properties as associations, individuals as objects and cardinality restrictions on association domain class as UML cardinalities. It is easy to visualize also subclass and inverse properties notations. For the full OWL 2 construct coverage we enrich the UML class diagrams with the new extension notations, e.g. (cf. [7,8]):

- fields in classes for *equivalent class*, *superclass* and *disjoint class* expressions written in Manchester OWL syntax [10];
- fields in associations and attributes for *equivalent*, *disjoint* and *super* properties and fields for property characteristics, e.g.,*functional*, *transitive*, etc.;
- anonymous classes containing *equivalent class expression* but no name (we show graphically only anonymous classes that need to have graphic representation in order to be able to describe other ontology concepts in the diagram);
- connectors (as lines) for visualizing binary *disjoint*, *equivalent*, etc. axioms;
- boxes with connectors for n-ary *disjoint*, *equivalent*, etc. axioms;
- connectors (lines) for visualizing object property restrictions *some*, *only*, *exactly*, as well as cardinality restrictions.

OWLGrEd provides option to specify class expressions in compact textual form rather than using separate graphical element for each logical item within class expression. If an expression is referenced in multiple places, it can optionally be shown as an anonymous class. An anonymous class is also used as a base for property domain/range specification, if this domain/range is not a named class.

Figure 1 illustrates some basic OWLGrEd constructs on a simple mini-University ontology, including different notation options for *EquivalentClasses* assertion, object property restriction and a comment. The notation is explained in more detail in [7].

[1] http://owlapi.sourceforge.net
[2] http://owlgred.lumii.lv/

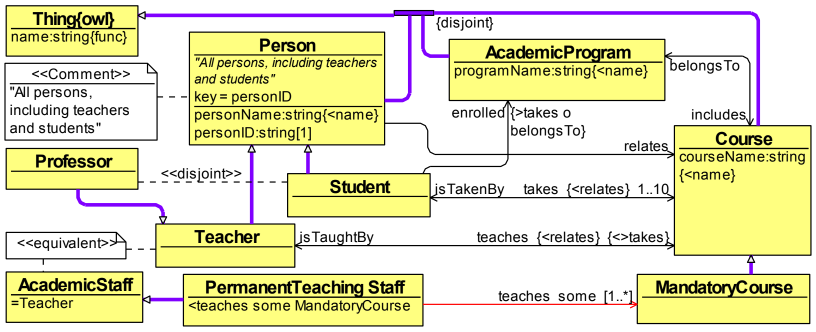

Fig. 1. Example: OWLGrEd notation for a mini-University ontology

The OWGrEd editor offers ontology interoperability (import/export) functionality with the Protégé 4.2 ontology editor [5]. The principal OWLGrEd usage ways are:

- ontology authoring (create and edit an ontology in OWLGrEd, then export it to Protégé to analyze and possibly submit it to other ontology processing tools)
- ontology visualization (an ontology that is imported from Protégé is displayed graphically to obtain a comprehensible visual view on it).

3 Visualizing Fragments of Ontology

The graphical form is ideal for understanding small ontologies, but for large ontologies it fast becomes overwhelming because of too many line crossings. For visualization of the ontology in the form of fragments an issue is to describe the fragments to be visualized since manual enumeration of all axioms to be included into a fragment would clearly be infeasible. Recently there has been work on signature-based automatic decomposition [11] of ontologies that allows specify ontology modules just in terms of their “core” terms/entities. The decomposition then finds all the axioms that are logically relevant for the given entities.

We have extended OWLGrEd editor with the Automatic Decomposition feature. A user can specify either a single ontology fragment, or a list of fragments covering the whole ontology that is to be visualized. The automatic decomposition then finds all the relevant axioms for each specified fragment thus allowing OWLGrEd showing the fragments visually in a graphical form.

As an example consider the schema.org ontology. It consists of about 300 classes, 110 object properties, 70 data properties and 310 subclass assertions. The ontology is clearly too large to be easily perceived as a single diagram. However, it would be feasible as well as meaningful to visualize fragments of the. For example, in the Figure 2 is shown a fragment that is centered on entities “Event”, “Product” and “Person”. Once the user has specified such an entity list, the tool automatically finds the relevant axioms for these entities and then shows this fragment graphically. It is possible to specify any number of such fragment signatures at a time and the tool will create visualization for each of them.

The experiments we have performed allows us to judge that the offered approach of combining of the traditional OWLGrEd ontology visualization means with ontology decomposition techniques would be a useful tool for the semantic technology community in ontology schema structure representation.

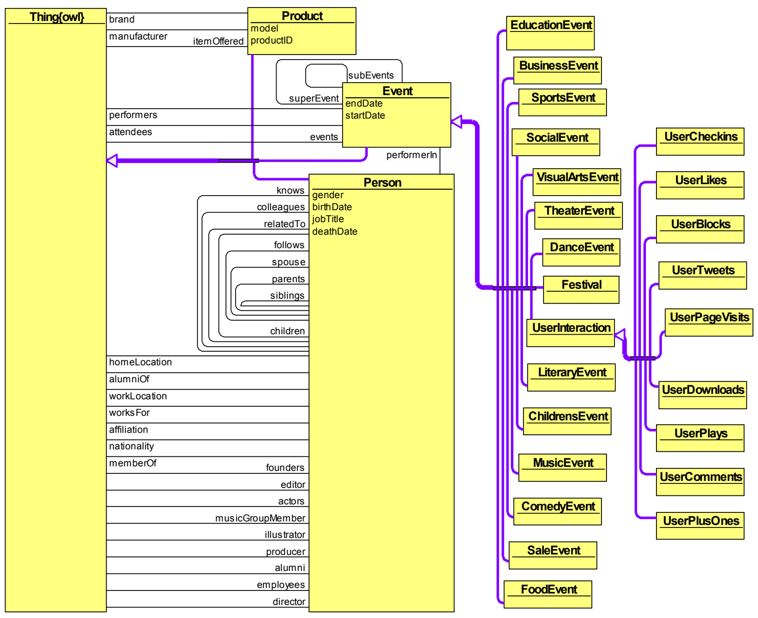

Fig. 2. Automatically extracted fragment of schema.org ontology based on a signature "Event, Product, Person".

References

1. Motik, B; Patel-Schneider P.F; Parsia B.: OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax, 2009
2. Brockmans, S., Volz, R., Eberhart, A., Löffler, P. Visual Modeling of OWL DL Ontologies Using UML, Proc. of ISWC 2004, LNCS 3298, pp. 198-213, 2004.
3. ODM UML profile for OWL, http://www.omg.org/spec/ODM/1.0/PDF/
4. TopBraid Composer, http://www.topquadrant.com/products/TB_Composer.html.
5. Protégé 4, http://protege.stanford.edu/
6. OWL Viz, http://www.co-ode.org/downloads/owlviz/
7. Barzdins, J.; Barzdins, G.; Cerans, K.; Liepins, R.; Sprogis, A.: OWLGrEd: a UML Style Graphical Notation and Editor for OWL 2. In Proc. of OWLED 2010, 2010.
8. Barzdins, J.; Cerans, K.; Liepins, R.; Sprogis, A.: UML Style Graphical Notation and Editor for OWL 2. In Proc. of BIR'2010, LNBIP, Springer 2010, vol. 64, p. 102-113, 2010.
9. Unified Modeling Language: Infrastructure, version 2.1. OMG Specification ptc/06-04-03, http://www.omg.org/docs/ptc/06-04-03.pdf
10. OWL 2 Manchester Syntax, http://www.w3.org/TR/owl2-manchester-syntax/
11. Klinov, P.; Vescovo, C.; Schneider, T.: Incrementally Updateable and Persistent Decomposition of OWL Ontologies. In Proc of OWLED 2012.