Domain-Specific OWL Ontology Visualization with OWLGrEd

Karlis Cerans[*], Renars Liepins[**], Arturs Sprogis[**], Julija Ovcinnikova[*], Guntis Barzdins[*]

Institute of Mathematics and Computer Science, University of Latvia
{Karlis.Cerans, Renars.Liepins, Arturs.Sprogis, Julija.Ovcinnikova, Guntis.Barzdins}@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 expressions. We present an extension mechanism for OWLGrEd that allows adding custom information areas, rules and visual effects to the ontology presentation thus enabling domain specific OWL ontology visualizations. The usage of OWLGrEd and its extensions is demonstrated on ontology engineering examples involving custom annotation visualizations, advanced UML class diagram constructs and integrity constraints in semantic database schema design.

Keywords: OWL, UML/OWL profile, OWLGrEd, domain-specific ontology visualization, semantic databases, integrity constraints

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 [1,2] ontologies in a graphical form, including UML Profile for OWL DL [3], ODM [4], TopBraid Composer [5], Protégé [6] plug-in OWLViz [7], OWLGrEd [8,9]. The approaches of [3,4,8,9] use UML [10,11] class diagrams to visualize OWL ontologies. A core principle here is to visualize an independent hierarchy of ontology classes and then structure the data and object property visualizations along the property domain and range classes. Depicting OWL classes as UML classes, OWL object properties as association roles and OWL data properties as attributes allows for easy graphical visualization also of subclass assertions, simple cardinality constraints and inverse-of relations. Further OWL ontology constructions (e.g. class expressions, properties with more than one domain assertion, sub-property relations etc.) are then handled by some auxiliary means in the notation and the editor. The design choice for OWLGrEd is to use textual OWL Manchester syntax [12] for class expressions where the graphical notation is not

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

available or is not desired thus allowing compact and comprehensible presentation of up to medium-sized ontologies[1] within a single diagram.

Although UML-style class diagram notation for basic OWL constructs can be successfully used in ontology rendering and authoring, there are further features that would be welcome in a graphical ontology editor. Since annotations in OWL 2.0 [2] may carry substantial model information that just does not fit into the "logical" part of the ontology, it would be important to offer means for domain-specific visualization of annotation assertions via specific textual presentation or graphical effects, e.g. as outlined in [13]. As a special case, a UML-style modeling in OWL would benefit from graphical composition or property derived union notation (modeled semantically as annotation assertions to the respective properties).

With the advent of semantic OWL-based databases, such as StarDog [14], an important issue is rising about incorporating integrity constraints [15,16], also expressed in OWL syntax, in graphical database schema design. As an example of our technology application we provide a domain-specific ontology visualization profile for axiom-level annotations that separate "proper" (i.e. open-world) OWL axioms from integrity constraints, depicted within the same graphical ontology diagram.

The demonstration shows (i) working with OWLGrEd tool to render and author OWL ontologies (ii) OWLGrEd extension mechanism for creating domain-specific ontology visualization tools and (iii) created domain-specific tools, including OWLGrEd/S for integrity constraint specification, at work.

2 OWLGrEd Notation and Editor

OWLGrEd[2] provides a complete graphical notation for OWL 2 [2], based on UML class diagrams. We visualize OWL classes as UML classes, data properties as class attributes, object properties as associations, individuals as objects, cardinality restrictions on association domain class as UML cardinalities, etc. We enrich the UML class diagrams with the new extension notations, e.g. (cf. [8,9]):

- fields in classes for *equivalent class*, *superclass* and *disjoint class* expressions written in Manchester OWL syntax [12];
- 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 those 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

[1] Please see http://owlgred.lumii.lv/examples for some ontology presentations
[2] http://owlgred.lumii.lv/

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 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 [8].

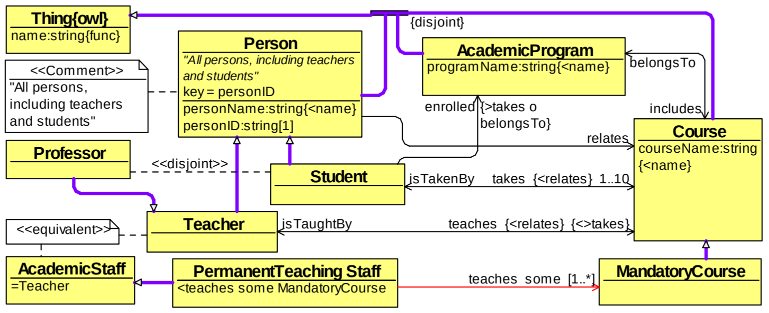

Fig. 1. Example: OWLGrEd notation for a mini-University ontology

The OWGrEd editor offers ontology interoperability (import/export) functionality with Protégé 4.1. ontology editor [6]. The principal OWLGrEd usage tool chains 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).

Any combination of these two OWLGrEd usage patterns, including ontology round-trip engineering between OWLGrEd and Protégé are possible, as well.

3 Creating Domain-Specific Ontology Visualizations

Domain-specific ontology visualizations in OWLGrEd ontology editor are defined by means of ontology visualization profiles. Each ontology visualization profile consists of a set of visual item (= abstract field) specifications, where each field comprises:

(i) field type (e.g. textual/boolean(= check box)/combo box field)

(ii) field appearance (e.g. visibility and text font style)

(iii) visual effects on ontology diagram symbols and other fields (e.g. symbol color and shape)

(iv) field semantics (what OWL axioms or axiom annotations a value in the field corresponds to).

For an ontology to be visualized in OWLGrEd in a domain-specific way, the corresponding ontology visualization profile has to be created or imported using OWLGrEd visualization profile plug-in. When the ontology created in such domain-specific extension of OWLGrEd is exported to Protégé ontology editor, the ontology diagram node and edge fields that correspond to profile visual items generate the OWL axioms or axiom annotations, as specified in field semantics description.

Consider an ontology *A* fragment visualized in a domain-specific way, as in Fig.2. The graphical notation has a new class field “DB” rendered textually with prefix “*{DB:*” and suffix “*}*”, a class field “isImportant” whose value “true” is rendered as orange background and 3D shape of the class symbol, and association role sub-field “isComposition” whose value “true” is rendered as diamond symbol on opposite association end. We desire to have these fields correspond to the following axioms:

AnnotationAssertion(A:DBExpr A:AcademicProgram "XProgram")
AnnotationAssertion(A:DBExpr A:Course "XCourse")
AnnotationAssertion(A:isImportant A:Teacher "true")
AnnotationAssertion(A:isComposition A:includes "true")

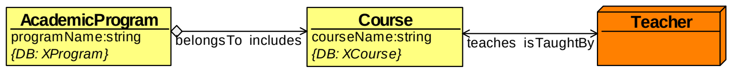

Fig. 2. Simple domain-specific ontology annotation visualization

This is achieved by semantics declarations: *AnnotationAssertion(:DBExpr $subject $value)* for the field “DB”, *AnnotationAssertion(:isImportant $subject "true")* for the value “true” in the boolean-typed field “isImportant”, and *AnnotationAssertion(:isComposition $subject "true")* for the value “true” in “isComposition”.

When an ontology that uses the *A:isImportant*, *A:DBExpr* and *A:isComposition* annotations (or other OWL built-in or user defined annotations whose visual image is foreseen in a loaded ontology visualization profile) is imported into OWLGrEd, the editor is able to create the domain-specific visualization (like Fig.2) automatically.

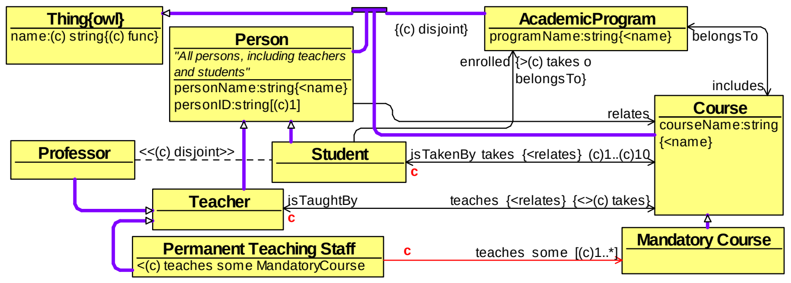

Fig. 3. Integrity constraint specification for mini-University ontology

4 Integrity Constraints in Semantic Database Schema Design

Using the ontology of Figure 1 as a schema for semantic database would be problematic due to the standard OWL axiom interpretation in “open-world” sense[3]. The

[3] This interpretation would allow to infer e.g. that a person is a student, if he/she has been entered into the database as taking (instead of teaching) a course, or that a student is enrolled in two academic programs just because of taking courses that belong to both of them.

solution we are offering is to mark explicitly the axioms whose interpretation in the open-world sense is undesirable, as integrity constraints[4].

The OWLGrEd editor is extended by “integrity constraint” visualization profile[5] that foresees a possibility to attach a (c)-mark (“c” for constraint) to visual places that can be identified as “holding” the concrete axioms, as in Fig. 3 for mini-University.

In the example, for instance, the axiom *ObjectPropertyDomain(A:takes A:Student)* is annotated to become *ObjectPropertyDomain(Annotation(C:isConstraint “true”) A:takes A:Student)* for a suitable namespace *C* holding the *isConstraint* annotation property. The visual **c**-notation placed at the beginning of *takes*-role link is obtained from a “DomainMode” field under the association role *takes*. The corresponding semantics specification for the “DomainMode” field causing the considered *ObjectPropertyDomain*-axiom annotation is *Annotation(C:isConstraint “true”)*.

The considered examples outline the potential of domain-specific ontology visualization using OWLGrEd and invite the reader either to apply the demonstrated ontology visualization profiles, or design his/her own ontology visualization tools.

References

1. Smith, M. K.; Welty, C.; and McGuiness, D.: OWL Web Ontology Language Guide, 2004
2. Motik, B; Patel-Schneider P.F; Parsia B.: OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax, 2009
3. 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.
4. ODM UML profile for OWL, http://www.omg.org/spec/ODM/1.0/PDF/
5. TopBraid Composer, http://www.topquadrant.com/products/TB_Composer.html.
6. Protégé 4, http://protege.stanford.edu/
7. OWL Viz, http://www.co-ode.org/downloads/owlviz/
8. 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.
9. 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.
10. Unified Modeling Language: Infrastructure, version 2.1. OMG Specification ptc/06-04-03, http://www.omg.org/docs/ptc/06-04-03.pdf
11. Unified Modeling Language: Superstructure, version 2.1. OMG Specification ptc/06-04-02, http://www.omg.org/docs/ptc/06-04-02.pdf
12. OWL 2 Manchester Syntax, http://www.w3.org/TR/owl2-manchester-syntax/
13. Barzdins, J.; Cerans, K.; Liepins, R.; Sprogis, A.: Advanced ontology visualization with OWLGrEd. In Proc. of OWLED 2011, 2011.
14. Stardog, http://stardog.com/
15. Tao, J.; Sirin, E.; Bao J; McGuinness, D.: Integrity Constraints in OWL. In Proc. of AAAI 2010, 2010.
16. Sirin, E; Smith, M; Vallace, E: Opening, Closing Worlds – On Integrity Constraints. In Proc. of OWLED 2008, 2008.

[4] We refer to [15, 16] for integrity constraint discussion in the context of StarDog databases, noting that our integrity constraint encoding is easily interconvertible with that of StarDog’s.

[5] The extended editor is available as OWLGrEd/S from http://owlgred.lumii.lv/s