Databases and Information Systems IX
G. Arnicans et al. (Eds.)

doi:10.3233/978-1-61499-714-6-17

Adding Verbalization to Graphical Ontology Editor OWLGrEd

Renārs LIEPIŅŠ[a], Normunds GRŪZĪTIS[a,b], Kārlis ČERĀNS[a,b], Jūlija OVČIŅŅIKOVA[a,b], Uldis BOJĀRS[a,b] and Edgars CELMS[a,b]

[a]*Institute of Mathematics and Computer Science, University of Latvia*
Raina bulvaris 29, Riga, LV-1459, Latvia
[b]*Faculty of Computing, University of Latvia*
Raina bulvaris 19, Riga, LV-1536, Latvia
{renars.liepins, normunds.gruzitis, karlis.cerans, julija.ovcinnikova, uldis.bojars, edgars.celms}@lumii.lv

Abstract. To participate in Semantic Web projects, domain experts need to be able to understand the ontologies involved. Visual notations can provide an overview of the ontology and help users to understand the connections among entities. However, users first need to learn the visual notation before they can interpret it correctly. Controlled natural language representation would be readable right away and might be preferred in case of complex axioms, however, the structure of the ontology would remain less apparent. To achieve the combination of the graphical and Controlled natural language approaches (CNL), we describe the possibility of adding CNL information into graphical OWL ontology editor OWLGrEd.

Keywords. OWL, Ontology visualization, Contextual verbalization, Controlled natural language

1. Introduction

Semantic Web technologies have been successfully applied in pilot projects and are transitioning toward mainstream adoption in the industry. In order for this transition to be successful, there are still barriers to be overcome. One of them are the difficulties that domain experts have in understanding mathematical formalisms and notations that are used in ontology engineering.

Visual notations have been proposed as a way to help domain experts to work with ontologies. Indeed, when domain experts collaborate with ontology experts in designing an ontology "they very quickly move to sketching 2D images to communicate their thoughts" [1]. The use of diagrams has also been supported by an empirical study done by Warren et al. where they reported that "one-third [of participants] commented on the value of drawing a diagram" to understand what is going on in the ontology [2].

Despite the apparent success of the graphical approaches, their unconditional use is still somewhat problematic. When a novice user wants to understand a particular ontology, he or she cannot just look at the diagram and know what it means. The user first needs to learn the syntax and semantics of the notation – its mapping to the underlying formalism. This limitation has long been noticed in software engineering [3] and, for this reason, formal models in software engineering are often translated into

informal textual documentation by systems analysts, so that they can be validated by domain experts [4].

A similar idea of automatic conversion of ontologies into seemingly informal controlled natural language (CNL) texts and presenting the texts to domain experts has been investigated by multiple groups [5,6,7]. CNL is more understandable to domain experts and end-users than the alternative representations because the notation itself does not have to be learned, or the learning time is very short. However, the comparative studies of textual and graphical notations have shown that while domain experts that are new to graphical notations better understand the natural language text, they still prefer the graphical notations in the long run [8,9]. It leads to a dilemma of how to introduce domain experts to ontologies. The CNL representation shall be readable right away and might be preferred in case of complex axioms (restrictions) while the graphical notation makes the overall structure and the connections more comprehensible.

We elaborate here on an approach that combines the benefits of both graphical notations and CNL verbalizations. We show, how to extend the graphical notation with interactive contextual verbalizations of the axioms that are represented by the selected graphical element. The graphical representation gives the users an overview of the ontology while the contextual verbalizations can explain what a particular graphical element means. Thus, domain experts that are novices in ontology engineering shall be able to learn and use the graphical notation rapidly and independently without special training.

Throughout this paper we refer to the OWLGrEd visual notation [10] and ontology editing and visualization tools that are using it. The OWLGrEd notation is a compact and complete UML-style notation for OWL 2 ontologies [11]. It relies on Manchester OWL Syntax [12] for certain class expressions. This notation is implemented in the OWLGrEd ontology editor[1] and its online ontology visualization tool[2] [13]. We describe in this paper the principles and architecture for introducing the natural language information into the OWLGrEd ontology editor, as well as architecture of web-based ontology verbalization environment creation. The OWLGrEd editor extension by means of CNL support features relies on a plugin mechanism for the OWLGrEd editor [14].

In Section 2, we present the general principles of extending graphical ontology notations with contextual natural language verbalizations. In Section 3, we overview the OWLGrEd ontology editor notation, present on example of ontology online verbalization on the basis of the OWLGrEd graphical ontology notation, as well as describe implementation of the CNL support within the OWLGrEd editor. Section 4 then summarizes the article.

2. Contextual Verbalization of Graphical Ontology Notations: the Vision

This section outlines the approach for contextual verbalization of graphical elements in ontology diagrams [15]. We start with a motivating example. During the outline of the contextual verbalization vision focusing particularly on OWL ontologies, we shall assume that the ontology and its graphical presentation have already been created and that the ontology symbols (names) are lexically motivated and consistent.

[1] http://owlgred.lumii.lv

[2] http://owlgred.lumii.lv/online_visualization/

2.1. Motivating Example

In most diagrammatic OWL ontology notations, object property declarations are shown either as boxes (for example in VOWL [16]) or as labeled links connecting the property domain and range classes as in OWLGrEd [10]. Figure 1 illustrates a simplified ontology fragment that includes classes *Person* and *Thing*, an object property *likes* and a data property *hasAge*. This fragment is represented in a (controlled) natural language, as well as using three alternative formal notations: Manchester OWL Syntax [12], VOWL and OWLGrEd. As can be seen, the visualizations are tiny and may already seem self-explanatory. Nevertheless, even in this simple case, the notation for domain experts may be far from obvious. For example, the Manchester OWL Syntax uses the terms *domain* and *range* when defining a property, and these terms may not be familiar to a domain expert. In the graphical notations, the situation is even worse because the user may not even suspect that the edges represent more than one assertion and that the assertions are far-reaching. In the case of *likes*, it means that everyone that likes something is *necessarily* a person, and vice versa.

We have encountered such problems in practice when introducing ontologies in the OWLGrEd notation to users familiar with the UML notation. Initially, it turned out that they are misunderstanding the meaning of the association edges. For example, they would interpret that the edge *likes* in Figure 1 means "persons *may* like persons", which is true, however, they would also assume that other disjoint classes could also have this property, which is false in OWL because multiple domain/range axioms of the same property are combined to form an intersection. Thus, even having a very simple ontology, there is a potential for misunderstanding the meaning of both the formal textual notation (e.g., Manchester OWL Syntax) and the graphical notations.

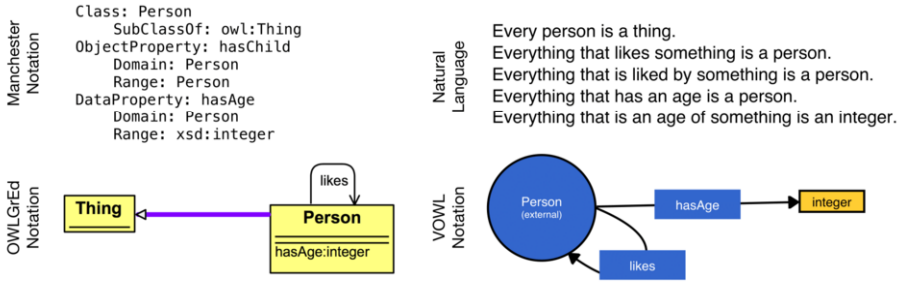

Figure 1. A simplified ontology fragment alternatively represented by using Manchester OWL Syntax, VOWL and OWLGrEd, and an explanation in a controlled natural language [15]

2.2. Proposed Approach

We propose to extend graphical ontology diagrams with contextual on-demand verbalizations of OWL axioms related to the selected diagram elements, with the goal to help users to better understand their ontologies and to learn the graphical notations based on their own and/or real-world examples.

The contextual verbalization of ontology diagrams relies on the assumption that every diagram element represents a set of ontology axioms, i.e., the ontology axioms are generally presented locally in the diagram, although possibly a single ontology axiom can be related to several elements of the diagram.

The same verbalization can be applied to all the different OWL visual notations, i.e., we do not have to design a new verbalization (explanation) grammar for each new visual notation, because they all are mapped to the same underlying OWL axioms. Thus, the OWL visualizers can reuse the same OWL verbalizers to provide contextual explanations of any graphical OWL notation.

By reusing ontology verbalizers, existing ontology visualization systems can be easily extended with a verbalization service. Figure 2 illustrates how the proposed approach might work in practice:

1. *Visualizer* is the existing visualization component that transforms an OWL ontology into its graphical representation.
2. The system is extended by a *User Selection* mechanism that allows users to select the graphical element that they want to verbalize.
3. *Collector* gathers a subset of the ontology axioms that correspond to the selected graphical element.
4. The relevant axioms are passed to *Verbalizer* that produces CNL statements – a textual explanation that is shown to the user.

The actual implementation would depend on the components used and on how the output of the verbalization component can be integrated into the resulting visualization.

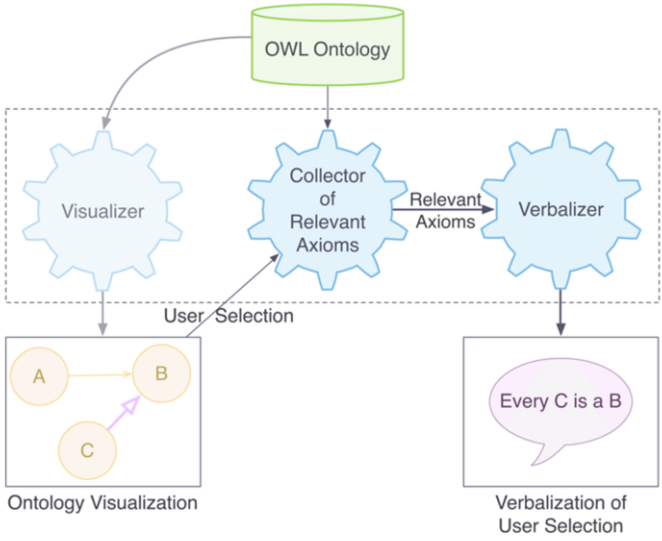

Figure 2. Architecture of a contextual ontology verbalizer [15]

In Section 3 we shall demonstrate the implementation of the architecture in the context of the OWLGrEd ontology editor.

With the proposed approach, when domain experts encounter the example ontology in Figure 1, they would not have to guess what the elements of this graphical notation mean. Instead, they can just ask the system to explain the notation using the ontology that they are exploring. When the user clicks on the edge likes in Figure 1 (in either visual notation), the system would show the verbalization that unambiguously explains the complete meaning of this graphical element:

Everything that likes something is a person. Everything that is liked by something is a person.

By applying the proposed approach and by using natural language to interactively explain what the graphical notation means, developers of graphical OWL editors and viewers can enable users (domain experts in particular) to avoid misinterpretations of ontology elements and their underlying axioms, resulting in a better understanding of both the ontology and the notation.

The verbalization can help users even in relatively simple cases, such as object property declarations where user's intuitive understanding of the domain and range of the property might not match what is asserted in the ontology. The verbalization of OWL axioms makes this information explicit while not requiring users to be ontology experts. The value of contextual ontology verbalization is even more apparent for elements whose semantics might be somewhat tricky even for more experienced users (e.g., *some*, *only* and cardinality constraints on properties, or OWLGrEd generalization forks with *disjoint* and *complete* constraints).

The verbalization of ontology axioms has been shown to be helpful in teaching OWL to newcomers both in practical experience reports [17] as well as in statistical evaluations [7].

3. Ontology Verbalization in OWLGrEd

3.1. Overview of the OWLGrEd Notation

OWLGrEd provides a graphical notation for OWL 2, based on UML class diagrams. OWL classes are typically visualized as UML classes, data properties as class attributes, object properties as association roles, individuals as objects, cardinality restrictions on association domain class as UML cardinalities, etc. The UML class diagrams are enriched with new extension notations, e.g. (cf. [10,18]):

- fields in classes for equivalent class, superclass and disjoint class expressions written in the Manchester OWL syntax [12];
- fields in association roles and attributes for equivalent, disjoint and super properties and fields for property characteristics, e.g., functional, transitive, etc.;
- anonymous classes containing an equivalent class expression but no name;
- 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.

Figure 3 contains a demonstration fragment of a Mini University ontology in the OWLGrEd notation, illustrating the class and subclass notation, data and object property notation, subproperty and object property restrictions, different ways of representing individuals, disjoint classes, class assertions and object property assertions. For instance, classes *Student* and *Teacher* are *Person* class subclasses. The subclass relation is represented as a generalization set element (a purple horizontal fork, an arrowed line to the super class and simple lines to subclasses). Class *Person* is disjoint with *AcademicProgram* and *Course* classes; this is represented textually in the class box using a prefix "<>" (<>*AcademicProgram,* <>*Course*). In a similar way, textually can be represented superclass (prefix "<") and equivalent class (prefix "=") axioms. Another

way for representing disjoint classes is using *{disjoint}* mark at the generalization set, as in case of *Assistant*, *Docent* and *Professor* classes. An association line between two classes, corresponding to an object property, may contain cardinality constraints (e.g. the association role *teaches* between *Teacher* and *Course*, with cardinality *0..2*). An object property restriction can be represented graphically with a red line, marked with an expression describing the restriction. For example, the arrow marked with *inverse(teaches) only* between classes *MandatoryCourse* and *Professor* corresponds to the following restriction in the Manchester notation:

MandatoryCourse SubClassOf inverse teaches only Professor

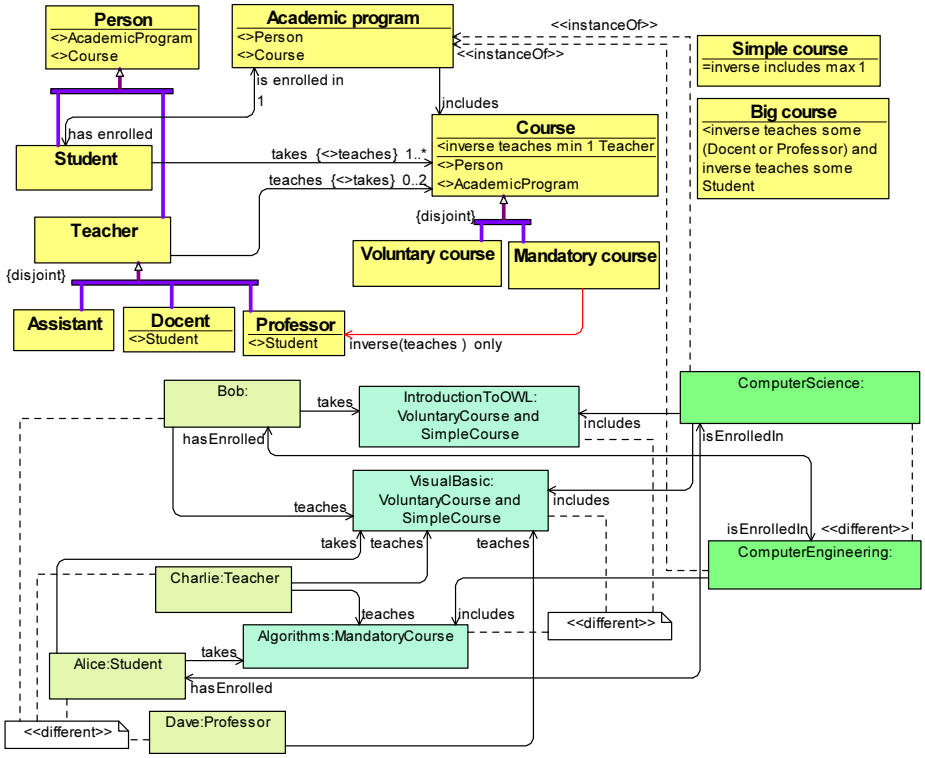

Figure 3. Demo fragment of Mini University Ontology

Class assertion axioms can be represented through the *<<instanceOf>>* connector, connecting a class and an object (e.g. the *<<instanceOf>>* link between the *Academic program* class and the *ComputerScience* object), or textually, by adding a class name after the object name separated with a ":" symbol (e.g. for the object *Dave:Professor*, "Dave" is the object name and "Professor" – the class name which the object belongs to).

The OWLGrEd tool allows both for ontology authoring (with an option to save the ontology in a standard textual format, e.g. RDF/XML or OWL Functional Syntax) and for ontology visualization that includes automated ontology diagram formation and a layouting step, followed by optional, manual diagram fine tuning.

An important feature of the OWLGrEd ontology editor is its plugin mechanism. The ontology editor plugins enable means for extending the editor notation and environment,

in a similar manner as profiles do for UML class diagrams [19,20]. A plugin to the ontology editor may include structural editor symbol extensions with fields and visual effects, as well as editor behavior extensions.

The OWLGrEd plugin mechanism shall be used to support the editor's extension which provides CNL verbalizations for user-selected ontology elements.

3.2. Ontology Online Verbalization Example

To show the power of adding lexical information to OWLGrEd, we start with ontology online visualization and verbalization demonstration. The ontology visualization and verbalization environment is obtained by:

- adding lexical information to the ontology elements, as they appear in the OWLGrEd editor;
- generating the visualization environment from an enriched ontology project.

Using Attempto Controlled English (ACE) [21] as a pivot CNL, the ontology visualization environment shall offer on-demand contextual multilingual verbalizations of OWL axioms corresponding to different visual elements in the ontology diagram.

The interactive ontology verbalization layer allows users to inspect a particular element of the presented ontology diagram and receive a verbal explanation of the ontology axioms that are related to this ontology element. By clicking a mouse pointer on an element, a pop-up widget is displayed, containing a CNL verbalization of the corresponding axioms. By default, the OWLGrEd visualizer minimizes the number of verbalization widgets shown simultaneously by hiding them after a certain timeout. For users to simultaneously see the verbalizations for multiple graphical elements, there is an option to "freeze" the widgets and prevent them from disappearing.

A demonstration of our approach, based on the example mini-university ontology, is available online[3]. Figure 4 shows a screenshot of the OWLGrEd visualization of this ontology containing a number of verbalizations.

These verbalizations describe the ontology elements that represent the class *Course*, the object property *teaches*, the individual *Alice* and the restriction on the class *MandatoryCourse*. Verbalizations are implicitly linked to the corresponding elements using the element labels when possible. While it might be less convenient to identify the implicit links in a static image, the interactive nature of the combined ontology visualization and verbalization tool makes it easier for users to keep track of relations between diagram elements and their verbalizations.

To illustrate the verbalization functionality, let us look at the object property *teaches*, represented in the diagram by an edge connecting the class *Teacher* to the class *Course*. It leads to the following ACE verbalization of four axioms:

Every teacher teaches at most 2 courses.
Everything that is taught by something is a course.
Everything that teaches something is a teacher.
If X takes Y then it is false that X teaches Y.

[3] http://owlgred.lumii.lv/cnl-demo

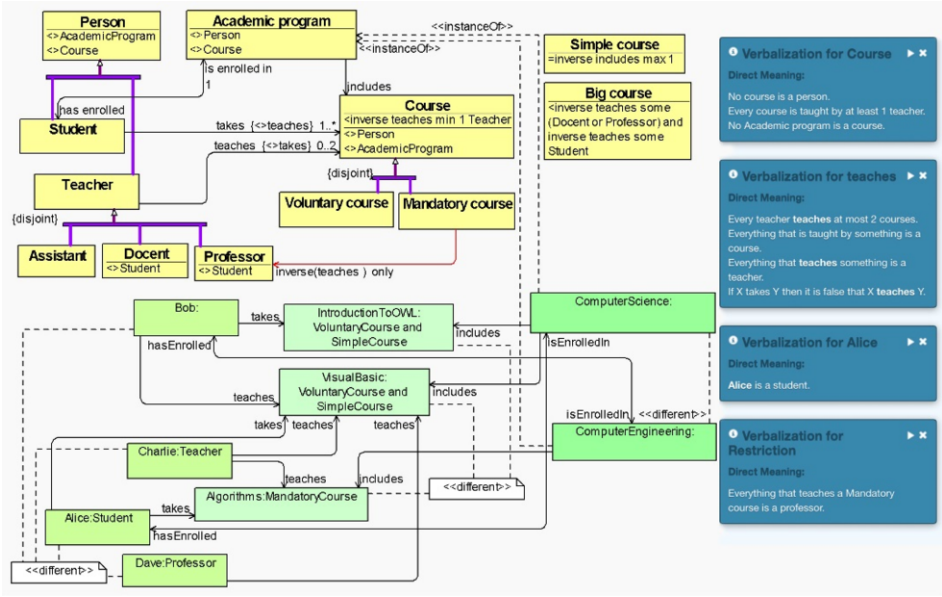

Figure 4. The example ontology in the OWLGrEd notation with CNL verbalizations (explanations) of selected diagram elements.

Note that the specific OWL terms, like *disjoint*, *subclass* and *inverse*, are not used in the ACE statements. The same meaning is expressed implicitly via paraphrasing – using more general common sense constructions and terms.

In this case, the edge represents not only the domain and range axioms of the property but also the cardinality of the range and the restriction that *teaches* is disjoint with *takes* (expressed by the if-then statement).

Further information about combining interactive contextual CNL verbalization with OWLGrEd ontology visualization is available in [15].

3.3. Implementation

To support the ontology verbalization, the lexical information about ontology entities needs to be introduced, typically using the ontology entity annotation mechanism. The lexical annotation can be added to an ontology by any ontology editor, however, the OWLGrEd editor equipped with its CNL plugin offers explicit services for convenient lexical information entry, including the simple lexical form information, as well as user-friendly services for much more involved entry of object property syntactic valence information. The CNL plugin of the OWLGrEd editor allows:

- Entry of entity lexical information;
- Computational lexicon generation;
- Ontology verbalization.

The first step towards the ontology verbalization is the lexical information entry. Technically, specific lexical information fields for classes, association roles, attributes, objects as well as the ontology header information (the ontology reference symbol in the project diagram) are added into the OWLGrEd editor. The current implementation

supports the lexical information entry for English which is an analytic language and for Latvian which is a highly inflected language.

Figure 5 illustrates the bilingual lexical information fields for classes. Information that needs to be entered is English and Latvian labels of the respective class. The class name is automatically generated from the English or Latvian label (English by default). Additionally, a more detailed lexical information is generated for each language and stored in a JSON format into the repository.

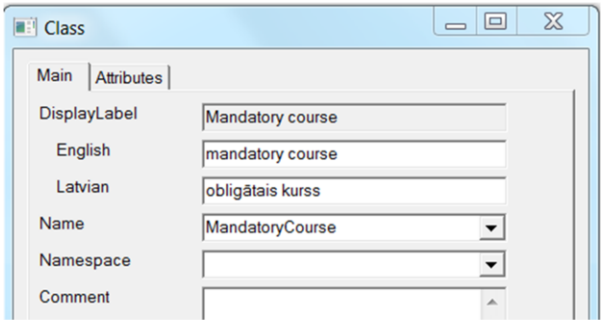

Figure 5. A class declaration dialog that includes input fields for the lexical information

Figure 6 illustrates the detailed lexical information generated for the class *Mandatory course* from the English and Latvian labels given in Figure 5. In English it is basically sufficient to infer the plural form of a class label, which is used for verbalizing specific axioms, while in Latvian it is necessary to infer also the grammatical gender and the internal structure of multi-word labels to ensure correct syntactic agreement in a sentence.


```
{
 "URI_gen":"MandatoryCourse",
 "entry.plural":"mandatory courses",
 "entry.gender":"neuter",
 "entry.singular":"mandatory course",
 "label_gen":"Mandatory course",
 "language":"en",
 "element":"class"
}
```

```
{
 "URI_gen":"ObligātaisKurss",
 "entry.components":"[obligātais][kurss]",
 "entry.number":"regular",
 "entry.gender":"masculine",
 "entry.pattern":"[A][N]",
 "label_gen":"Obligātais kurss",
 "language":"lv",
 "element":"class"
}
```


Figure 6. English and Latvian lexical information for the class *Mandatory course*

Lexical information input fields for association roles are shown in Figure 7. For each language, there are three fields present. Apart from the field *Predicate* where the lexical label of the property has to be entered, there are two contextual helper fields: *Subject* – provides the lexical label of the domain class; *Object* – provides the lexical label of the range class. Subject and object field values are filled automatically based on the domain and range classes. Additionally, there is a shared input field *Type of predicate* which determines whether the predicate is expressed as a verb or a noun and, thus, what grammatical constructions the OWL verbalizer should use.

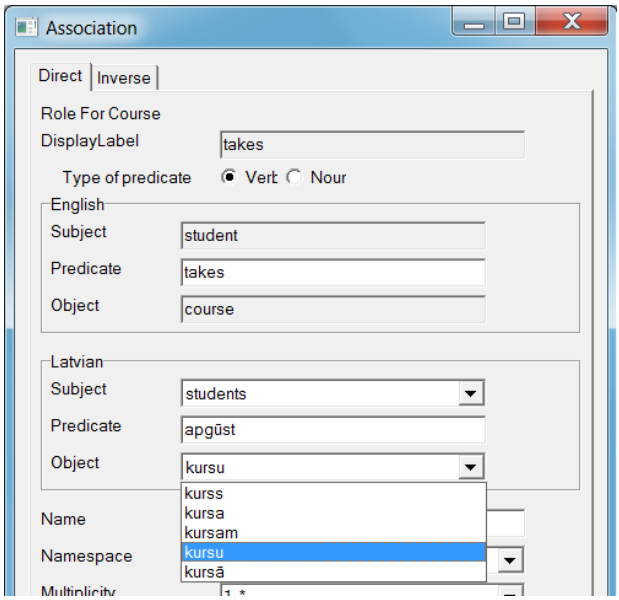

Figure 7. Lexical information fields for an association role

The role of the contextual subject and object fields is twofold. First, they help to specify the property label consistently, including the use of auxiliary verbs and prepositions, so that the predicate makes a grammatically and lexically valid clause. Consequently, the automatically generated property names are more consistent and readable as well. Second, this is an implicit and intuitive means how the user specifies the syntactic valence of the predicate – the grammatical agreement between the predicate and its subject and object. While it is not an issue in highly analytical languages like English, the grammatical case of the subject and object often depends on the particular verb in inflected languages. The CNL plugin suggests the valence pattern by automatically selecting the most likely inflectional forms of the subject and object labels instead of explicitly exposing the user to the grammatical cases (accusative, locative, dative etc.). If the automatic suggestion is incorrect, the user implicitly corrects the grammatical case by selecting the right inflectional form of the label.

A somewhat similar approach is used in LEMON Assistant [22], a web application that allows to equip existing ontologies with lexicalization patterns according to the LEMON model [23], an RDF model for representing lexical information relative to ontology entities. In order to help the user in checking the correctness of a lexicalization pattern, LEMON Assistant generates a sentence in natural language based on the subject-predicate-object labels provided by the user.

Based on *Subject*, *Object* and *Predicate* values, detailed lexical information for the association role is generated as illustrated in Figure 8: the user has formed a grammatically correct clause *"[ikviens] students apgūst [kādu] kursu"* (*"[every] student takes [a] course"*) from which the CNL plugin is able to generate the entire lexical and grammatical structure characterizing the predicate.

```
{
 "URI_gen":"takes",
 "predicate.entry.tense":"simple",
 "predicate.entry.present":"takes",
 "predicate.entry.participle":"taken",
 "predicate.entry.infinitive":"take",
 "predicate.entry.voice":"active",
 "predicate.entry.preposition":"",
 "POS":"verb",
 "label_gen":"takes",
 "language":"en",
 "element":"objectProperty",
 "subject":"student",
 "object":"course"
}
```

```
{
 "URI_gen":"apgūst",
 "predicate.entry.tense":"simple",
 "predicate.entry.present":"apgūst",
 "predicate.entry.past":"apguva",
 "predicate.entry.infinitive":"apgūt",
 "predicate.entry.voice":"active",
 "predicate.entry.preposition":"",
 "predicate.entry.object":"accusative",
 "predicate.entry.subject":"nominative",
 "POS":"verb",
 "subject.entry.gender":"masculine",
 "subject.entry.number":"regular",
 "subject.entry.pattern":"[N]",
 "subject.entry.components":"[students]",
 "object.entry.gender":"masculine",
 "object.entry.number":"regular",
 "object.entry.pattern":"[N]",
 "object.entry.components":"[kurss]",
 "label_gen":"apgūst",
 "language":"lv",
 "element":"objectProperty"
}
```


Figure 8. English and Latvian lexical information for an association role

The acquired lexical information is used in the on-demand verbalization of ontology axioms that underlie graphical elements in an ontology diagram as exemplified in Figure 4. The lexical labels of ontology entities are used also in the presentation of ontology diagrams, depending on the rendering language (e.g. English or Latvian) which the user has chosen. Figure 9 illustrates a fragment of the Mini University ontology diagram rendered alternatively in English and Latvian.

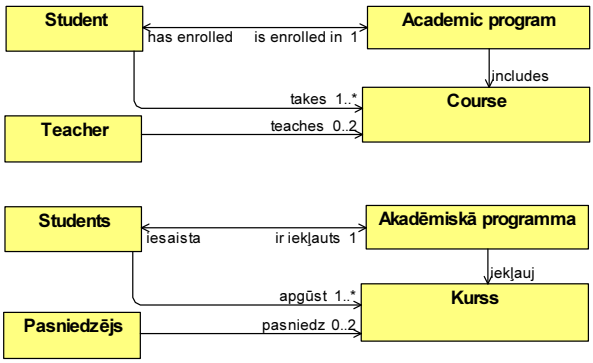

Figure 9. A fragment of an ontology diagram rendered alternatively in English and Latvian

The essential lexical information is saved during the ontology export in textual form as *AnnotationAssertion* axioms, e.g.:

AnnotationAssertion(languageFields:LabelEn :Course "Course") – for English
AnnotationAssertion(languageFields:LabelLv :Course "Kurss") – for Latvian

For association roles (object properties) the information saved into ontology consists of basic lexical form, as well as part of speech information (whether the property is expressed by a noun, or a verb). The saved information can be extended also to include the syntactic valence information to obtain a self-contained verbalizable ontology.

From the explicitly entered and implicitly inferred lexical information (Figure 6 and Figure 8), a multilingual computational lexicon is generated and compiled for each ontology. This is done by using Grammatical Framework (GF) [24] which is convenient and well-suited for the implementation of multilingual CNLs. The ontology-specific lexicon is then linked to a pre-compiled ontology-independent multilingual GF grammar for the OWL subset of ACE [25]. We follow a two-level OWL-to-CNL approach suggested in [26] in order to map the ontology symbols (entity names) to their lexical and inflectional forms in different languages and in different syntactic constructions. Although we have experimented only with English and Latvian, GF provides a reusable resource grammar library for about 30 languages, which greatly facilitates adding a new language. Note that GF is used also by the above mentioned LEMON Assistant.

The interactive ontology visualization for the web environment is generated from an ontology with verbalization information presented in OWLGrEd ontology editor the in following steps:

- Verbalization information is generated for each ontology element;
- The ontology graphical structure (boxes, lines with their compartment and style information) is coded into a JSON format;
- The acquired JSON structure is enriched with the verbalization information;
- The JSON structure is loaded into the ontology visualization web environment, where the ontology graphical structure is presented with enabled verbalization context information.

4. Conclusions

Mathematical formalisms used in ontology engineering are hard to understand for domain experts. Usually, graphical notations are suggested as a solution to this problem. However, the graphical notations, while preferred by domain experts, still have to be learned to be genuinely helpful in understanding. Until now the only way to learn these notations was by reading the documentation.

In this article, we demonstrate how to combine ontology visualizations and CNL verbalizations in order to solve the learning problem. Using this approach, the domain expert can interactively select a graphical element and receive the explanation of what the element means. The explanation is generated by passing the corresponding axioms of the element through an existing verbalization service. The service returns natural language sentences explaining the OWL axioms that correspond to the selected element, thus explaining what it means.

CNL explanations can help domain experts to rapidly and independently learn and use the graphical notation from the beginning, without extensive training, making it easier for domain experts to participate in ontology engineering, thus solving one of the problems that hinder the adoption of Semantic Web technologies in the mainstream industry.

Although the ontology has to be encoded with lexical information to enable its verbalization, our experience with OWLGrEd editor, outlined in this paper, shows that

convenient user interface for this enrichment can be offered and is able to handle also the sophisticated grammatical constructs (verb syntactic valences) in a user friendly way.

Acknowledgments

This work has been supported by the ESF project 2013/0005/1DP/1.1.1.2.0/13/APIA/VIAA/049 and the Latvian State Research program NexIT project No. 1 "Technologies of ontologies, semantic web and security" at the Institute of Mathematics and Computer Science, University of Latvia.

References

[1] J. Howse, G. Stapleton, K. Taylor, P. Chapman, Visualizing ontologies: A case study. In: *The Semantic Web* – ISWC 2011, pp. 257–272. Springer, 2011.
[2] P. Warren, P. Mulholland, T. Collins, E. Motta, The usability of Description Logics. In: *The Semantic Web: Trends and Challenges*, pp. 550–564, Springer, 2014.
[3] K. Siau, Informational and computational equivalence in comparing information modeling methods. *Journal of Database Management* 15(1), 73–86, 2004.
[4] P.J. Frederiks, T.P. Van der Weide, Information modeling: The process and the required competencies of its participants, *Data & Knowledge Engineering* 58(1), 4–20, 2006.
[5] R. Power, A. Third, Expressing OWL axioms by English sentences: dubious in theory, feasible in practice. In: *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pp. 1006–1013, Association for Computational Linguistics, 2010.
[6] R. Stevens, J. Malone, S. Williams, R. Power, A. Third, Automating generation of textual class definitions from OWL to English. *J. Biomedical Semantics* 2(S-2), S5, 2011.
[7] T. Kuhn, The understandability of OWL statements in controlled English. *Semantic Web 4(1)*, 101–115, 2013.
[8] A.Ottensooser, A. Fekete, H.A. Reijers, J. Mendling, C.Menictas, Making sense of business process descriptions: An experimental comparison of graphical and textual notations. *Journal of Systems and Software* 85(3), 596–606, 2012.
[9] Z. Sharafi, A. Marchetto, A. Susi, G. Antoniol, Y.G. Gueheneuc, An empirical study on the efficiency of graphical vs. textual representations in requirements comprehension. In: *21st International Conference on Program Comprehension*, pp. 33–42, IEEE, 2013.
[10] J. Bārzdiņš, G. Bārzdiņš, K. Čerāns, R. Liepiņš, A. Sproģis, UML-style graphical notation and editor for OWL 2. In: *Perspectives in Business Informatics Research*, pp. 102–114, Springer 2010.
[11] B. Motik, P.F. Patel-Schneider, B. Parsia, OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax, 2009.
[12] M. Horridge, P.F. Patel-Schneider, OWL 2 Web Ontology Language Manchester Syntax. *W3C Working Group Not,* 2009.
[13] R. Liepins, M. Grasmanis, U. Bojars, OWLGrEd ontology visualizer. In: *ISWC Developers Workshop 2014*, CEUR, 2015.
[14] K.Čerāns, J.Ovčiņņikova, R.Liepiņš, A.Sproģis, Advanced OWL 2.0 Ontology Visualization in OWLGrEd In: A.Caplinskas, G.Dzemyda, A.Lupeikiene, O.Vasilecas (eds*.), Databases and Information Systems VII, Frontiers in Artificial Intelligence and Applications*, IOS Press, Vol 249, pp.41-54, 2013.
[15] R. Liepiņš, U. Bojārs, N. Grūzītis, K. Čerāns, E. Celms, Towards self-explanatory ontology visualization with contextual verbalization. In: *Databases and Information Systems: 12th International Baltic Conference*, DB&IS 2016, Riga, Latvia, July 4-6, 2016, Proceedings. pp. 3–17, Springer, 2016.
[16] S.Lohmann, S. Negru, F. Haag, T. Ertl, VOWL 2: User-oriented visualization of ontologies. In: *Knowledge Engineering and Knowledge Management*, pp. 266–281, Springer, 2014.
[17] A. Rector, N. Drummond, M. Horridge, J. Rogers, H. Knublauch, R. Stevens, H. Wang, C. Wroe, OWL pizzas: Practical experience of teaching OWL-DL: Common errors & common patterns. In: *Engineering Knowledge in the Age of the Semantic Web*, pp. 63–81, Springer, 2004.
[18] J. Barzdins, G. Barzdins, K. Cerans, R. Liepins, A. Sprogis, OWLGrEd: a UML Style Graphical Notation and Editor for OWL 2. In: *Proc. of OWLED 2010*, 2010.

[19] Unified Modeling Language: Infrastructure, version 2.1. OMG Specification ptc/06-04-03, http://www.omg.org/docs/ptc/06-04-03.pdf
[20] Unified Modeling Language: Superstructure, version 2.1. OMG Specification ptc/06-04-02, http://www.omg.org/docs/ptc/06-04-02.pdf
[21] N.E. Fuchs, K. Kaljurand, T.Kuhn, Attempto Controlled English for knowledge representation. In: *Reasoning Web*, pp. 104–124, Springer, 2008.
[22] R. Mariano, C. Unger, Lemonade: A Web Assistant for Creating and Debugging Ontology Lexica. In: Proceedings of the 20th International Conference on Applications of Natural Language to Information Systems (NLDB), pp. 448–452, 2015.
[23] J. McCrae, G. Aguado-de-Cea, P. Buitelaar et al., Interchanging lexical resources on the Semantic Web. *Language Resources and Evaluation* 46(4), 701–719, 2012.
[24] A. Ranta, Grammatical Framework, a type-theoretical grammar formalism. *Journal of Functional Programming* 14(2), 145–189, 2004.
[25] J. Camilleri, N. Fuchs, K. Kaljurand, ACE grammar library. Tech. Rep. MOLTO Project Deliverable D11.1 2012.
[26] N. Gruzitis, G. Barzdins, Towards a more natural multilingual controlled language interface to OWL. In: *Proceedings of the 9th International Conference on Computational Semantics*. pp. 1006–1013, 2011.