Adding Context in OWL Ontologies: Visual Qualifiers in OWLGrEd

Jūlija Ovčiņņikova[1,*], Kārlis Čerāns[1,*]

[1]Institute of Mathematics and Computer Science, University of Latvia, Riga, Latvia

Abstract

We present an extension to the OWLGrEd visual ontology editor that enables visual annotation of OWL data and object properties, and instance property assertions with contextual qualifiers such as temporal scope, provenance, and confidence. Inspired by RDF 1.2 reified triples, our approach uses OWL 2 annotation mechanisms to enrich both property declarations and assertions without altering ontology semantics. This results in an intuitive visual representation of contextual data in OWL ontologies.

Keywords

OWL 2, visual ontology modelling, semantic qualifiers

1. Introduction

A wide range of ontology visualization tools provide graphical representations of OWL 2 [1] ontologies to facilitate understanding. For example, WebVOWL [2] and OntoDia [3] offer an interactive graph-based view for ontology exploration, and OWLGrEd [4, 5] supports a UML-style diagram notation for OWL ontologies. Tools like Chowlk [6], GRAPHOL [7], and G-OWL [8] provide fully diagrammatic ontology modelling environments, where properties are treated as first-class graphical elements. However, despite the richness of these visualization tools, none offers a way to attach context-rich qualifiers – such as temporal scopes, provenance information, or confidence scores – directly to properties and their assertions within the diagrams.

In this work, we present an extension to the OWLGrEd ontology editor that fills this gap by enabling visual qualifiers on relationships. Our approach enables ontology developers to attach contextual data (e.g., time intervals, source identifiers, trust/confidence values) to both property declarations (schema-level relationships) and individual property assertions (instance-level facts) within an OWL ontology. This is achieved using the OWL 2 annotation mechanisms, which enable the linking of additional information to ontology entities and axioms. By leveraging standard annotation properties in OWL 2, we ensure that qualifiers are encoded in a semantically compatible way – they do not alter the formal semantics of the ontology, but enrich it with supporting information. The result allows users to explore not just the structure of an ontology but also the context of its relationships within a single integrated diagram.

2. Background: OWLGrEd Notation

OWLGrEd [4, 5] provides a UML-style graphical notation for OWL 2. OWL classes appear as UML classes, object properties as associations, data properties as attributes, and individuals as instances. Logical axioms such as subclass, equivalence, disjointness, and restrictions are displayed textually (in Manchester syntax [9]) or with visual connectors. OWLGrEd also includes features as visual fields for expressing property characteristics such as functional or transitive properties, the use of anonymous classes to represent complex class expressions, and diagrammatic elements to represent binary and n-ary

Posters and Demos Track at SEMANTiCS'25: International Conference on Semantic Systems, September 3–5, 2025, Vienna, Austria
*Corresponding author.
julija.ovcinnikova@lumii.lv (J. Ovčiņņikova); karlis.cerans@lumii.lv (K. Čerāns)
0000-0002-5884-763X (J. Ovčiņņikova); 0000-0002-0154-5294 (K. Čerāns)

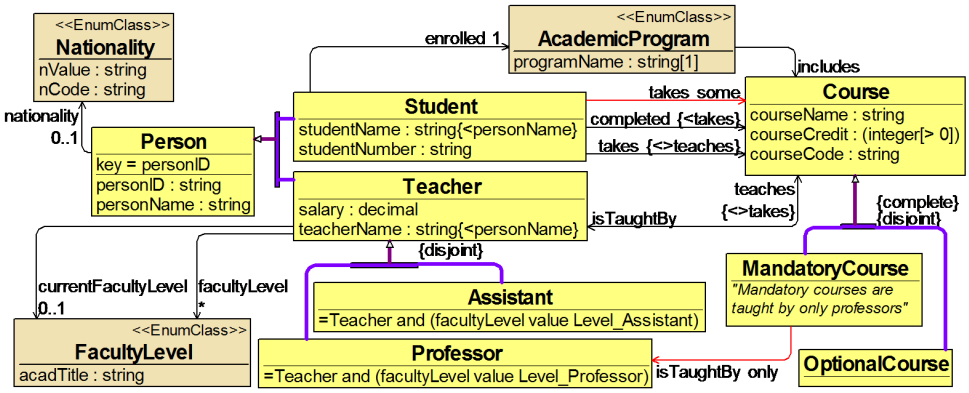

Figure 1: Mini University ontology

axioms such as disjointness or equivalence. OWLGrEd has been built in GRTP visual tool development platform [10], based on a dedicated graph diagramming engine [11].

3. Visual Notation Extensions

To introduce property qualifiers into the OWLGrEd visual notation, we provide the option to visually specify object and data properties using dedicated graphical boxes, and enable describing qualifiers within compartments of these property boxes.

At the instance level, we provide a visual notation for recording any annotations on data and object property assertions, with the intention to use them with the annotation properties introduced as data and object property qualifiers on the schema level.

We use color-coding to distinguish property types: object properties and object property assertions are depicted in blue hexagonal boxes, while data properties and data property assertions use green hexagonal boxes (see Figure 2). Each box can graphically connect to relevant ontology elements: classes in the case of property declarations, or individuals in the case of assertions. The graphical connections follow the logical structure of RDF triples: the subject (either a class or an individual) links to the property box with an outgoing arrow, and the object (class or individual) connects via an incoming arrow. This left-to-right flow visually mirrors RDF's subject–predicate–object structure and supports intuitive reading of both ontology declarations and instance-level assertions. Alternatively, information about domain and range (in the case of declarations) or subject and object/value (in the case of assertions) can be displayed textually within the box itself using labelled fields such as "*Domain*", "*Range*" (see the third row of Figure 2), and "*Value*" (fourth row of Figure 2). This dual representation supports both compact and detailed modelling preferences.

Additionally, generalization relationships can be used with object and data properties to visually encode sub-property axioms, clearly representing hierarchical relationships among properties within the ontology (as shown in the first row of Figure 2).

Each of the property and assertion visual elements can be enriched with qualifiers – fields that provide contextual information about the axiom, such as temporal validity, certainty, or provenance. Qualifiers like *datePaid*, *mark*, and *tutor* are displayed as labelled fields within the hexagonal box.

4. Qualifier annotations

OWL 2 does not have a native notation specifically dedicated for expressing property qualifiers. However, it provides mechanisms, such as annotation properties, that can effectively represent this information. In this work, we use the OWL 2 annotation mechanism to introduce property qualifiers such as temporal scopes, provenance information, and confidence levels. These annotations enrich both

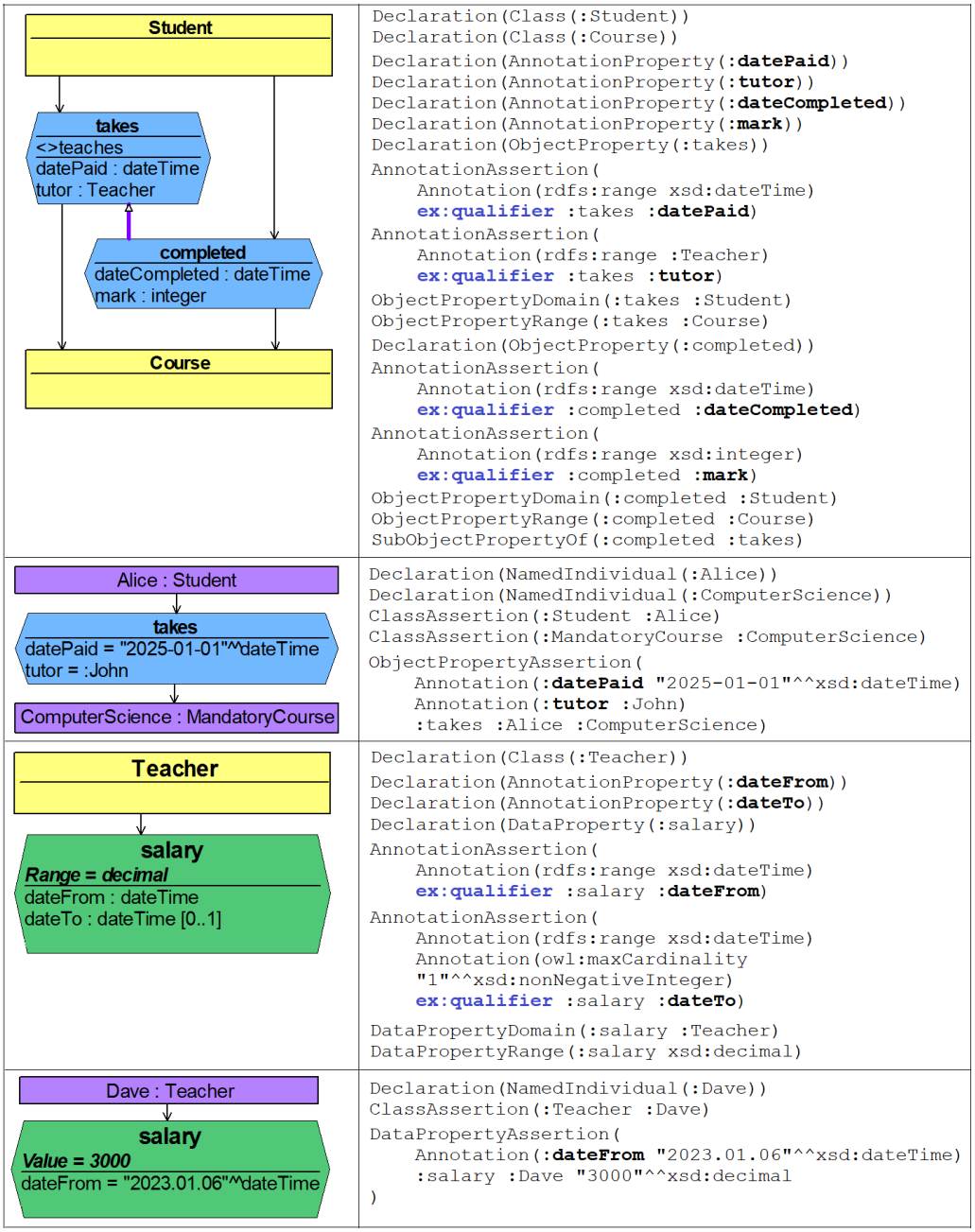

Figure 2: Object/Data property and Object/Data property assertion with qualifiers

property declarations (schema-level relationships) and property assertions (instance-level facts) without modifying the formal semantics of the ontology.

The property qualifiers (e.g., *dateTo*, *tutor*) are encoded in the ontology as annotation properties. These qualifiers are attached to the corresponding data or object properties using a dedicated annotation property *ex:qualifier*[1]. The expected type of qualifier values is specified using a nested annotation with annotation property *rdfs:range*, which may refer to a datatype or a class IRI, depending on whether the qualifier values are literals or resources (see the example of the property *:salary* below). Additional information, such as cardinality constraints (e.g., *owl:maxCardinality* annotation property) indicating how many times a qualifier may or must appear, can also be included. For example, the following axiom indicates that the data property *:salary* is qualified by *:dateTo*, which is expected to be a datetime and not occur more than once per salary assertion:

[1] `ex:<http://owlgred.lumii.lv/2011/1.1/extensions#>`

```
AnnotationAssertion(
  Annotation(rdfs:range xsd:dateTime)
  Annotation(owl:maxCardinality "1"^^xsd:nonNegativeInteger)
  ex:qualifier :salary :dateTo)
```

Once defined and annotated in this way, the same annotation properties are consistently applied to annotate the corresponding property assertions on the instance level, ensuring uniform usage across the ontology. For example, a data property assertion like *:Dave :salary "3000"* can be annotated with qualifier *dateFrom* and value "*2023-01-06*".

We have chosen to use the local semantics for the qualifier visualizations as the qualifier range (and cardinality) assertions are added directly into the axioms asserting a particular qualifier for a property (e.g., *AnnotationAssertion(ex:qualifier :salary :dateTo)* in Figure 2). Another option would have been to introduce the qualifier property characteristics globally, as follows:

```
AnnotationAssertion(ex:qualifier :salary :dateTo)
AnnotationPropertyRange(:dateTo xsd:dateTime)
AnnotationAssertion(owl:maxCardinality :dateTo 1)
```

The local option has been chosen since it can be expected that the same qualifiers could be applied in different places in the ontology, together with a different characteristics.

Although these qualifier annotations do not affect the ontology's formal semantics from an OWL reasoning perspective, they enrich axioms with contextual information. This enhances visualization and better readability for users while maintaining full compatibility with OWL 2 standards.

This approach is conceptually inspired by the RDF 1.2 notion of reified triples [12], which will offer a mechanism for attaching data directly to RDF statements. We observe that the annotation construct in OWL 2 is fully sufficient to record the qualifier information built into the RDF 1.2 architecture. We note that our approach allows to visualize the situations when the qualifier values are (expected to be) literals and when they are resources (denoted by IRIs).

We have implemented a prototype extension of the OWLGrEd editor to support visual annotation of OWL axioms.[2] The prototype includes a sample mini-University project using the extra annotation notation described in this paper.

5. Conclusions

We have extended the UML-style OWL diagramming tool OWLGrEd with the ability to represent qualifiers on both schema and instance relationships, using a clear syntax that extends the graphical notation without overcomplicating it. This extension introduces new diagram elements and syntax for qualifiers while remaining fully compliant with OWL 2 standards (qualifiers are stored as OWL annotations, and thus preserve semantic compatibility).

We expect that our contributions would enhance the usability of the tool - ontology engineers and domain experts can now capture and view contextual data directly in the ontology diagram, without needing external documentation. The integrated qualifiers not only improve the expressiveness and understandability of ontology visualizations, but also open the door for some forms of enhanced reasoning and analysis: since the context qualifiers are formally represented, they can be used by reasoning tools or queries (e.g., temporal reasoner) to provide richer answers and explanations.

As future work, we plan to implement the described graphical notation in a web-based environment that supports the full lifecycle of visual ontology development. We also intend to conduct a detailed evaluation of the notation within the context of the upcoming ontology development tool.

Acknowledgments

This work was supported by activity 1.1.1.9 Research application No 1.1.1.9/LZP/1/24/037 of the Activity

[2] https://owlgred.lumii.lv/downloads/OWLGrEd_extended.zip

"Post-doctoral Research" "*Visual methods and tools for ontology management*", and the Latvian Council of Science, project "*What's in Your Knowledge Graph?*" (Project No. lzp-2024/1-0665).

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT (GPT-4) in order to: Improve writing style; Grammar and spelling check. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] P. Hitzler, M. Krötzsch, B. Parsia, P. F. Patel-Schneider, S. Rudolph, Owl 2 web ontology language primer (second edition), https://www.w3.org/TR/owl2-primer/, 2012. W3C Recommendation, 11 December 2012.

[2] V. Wiens, S. Lohmann, S. Auer, Webvowl editor: Device-independent visual ontology modeling, in: Proceedings of ISWC 2018 Posters & Demonstrations, Industry and Blue Sky Ideas Tracks, volume 2180 of *CEUR Workshop Proceedings*, 2018. URL: https://ceur-ws.org/Vol-2180/paper-75.pdf.

[3] D. Mouromtsev, D. Pavlov, Y. Emelyanov, A. Morozov, D. Razdyakonov, M. Galkin, The simple, web-based tool for visualization and sharing of semantic data and ontologies, in: Proceedings of the ISWC 2015 Posters & Demonstrations Track, volume 1486 of *CEUR Workshop Proceedings*, 2015. URL: https://ceur-ws.org/Vol-1486/paper_77.pdf.

[4] J. Bārzdiņš, G. Bārzdiņš, K. Čerāns, R. Liepiņš, A. Sproģis, Uml style graphical notation and editor for owl 2, in: P. Forbrig, H. Günther (Eds.), Perspectives in Business Informatics Research, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 102–114.

[5] K. Čerāns, J. Ovčiņņikova, R. Liepiņš, M. Grasmanis, Extensible visualizations of ontologies in owlgred, in: P. Hitzler, S. Kirrane, O. Hartig, V. de Boer, M.-E. Vidal, M. Maleshkova, S. Schlobach, K. Hammar, N. Lasierra, S. Stadtmüller, K. Hose, R. Verborgh (Eds.), The Semantic Web: ESWC 2019 Satellite Events, Springer International Publishing, Cham, 2019, pp. 191–196.

[6] S. Chávez-Feria, R. García-Castro, M. Poveda-Villalón, Chowlk: from uml-based ontology conceptualizations to owl, in: P. Groth, M.-E. Vidal, F. Suchanek, P. Szekley, P. Kapanipathi, C. Pesquita, H. Skaf-Molli, M. Tamper (Eds.), The Semantic Web, Springer International Publishing, Cham, 2022, pp. 338–352.

[7] D. Lembo, V. Santarelli, D. F. Savo, G. D. Giacomo, Graphol : A Graphical Language for Ontology Modeling Equivalent to OWL 2, Future Internet 14 (2022) 1–29. URL: https://ideas.repec.org/a/gam/jftint/v14y2022i3p78-d760974.html.

[8] M. Héon, G. Paquette, G-owl: A complete visual syntax for owl 2 ontology modeling and communication, Semantic Web (2020). doi:10.3233/SW-200390.

[9] M. Horridge, P. F. Patel-Schneider, Owl 2 web ontology language: Manchester syntax (second edition), https://www.w3.org/TR/owl2-manchester-syntax/, 2012. W3C Working Group Note, 11 December 2012.

[10] J. Bārzdiņš, A. Zariņš, K. Čerāns, A. Kalniņš, E. Rencis, L. Lāce, R. Liepiņš, A. Sproģis, Grtp: Transformation based graphical tool building platform, volume 297, 2007. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84877312487&partnerID=40&md5=ed2bb5c3aa099ab8373880519d7491c6.

[11] J. Bārzdiņš, K. Cerans, S. Kozlovics, E. Rencis, A. Zarins, A graph diagram engine for the transformation-driven architecture, volume 439, 2009. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84877264914&partnerID=40&md5=d0ae5d9728018b2162a0ffc4f9febd46.

[12] O. Hartig, P.-A. Champin, G. Kellogg, A. Seaborne, Rdf 1.2 concepts and abstract syntax, https://www.w3.org/TR/rdf12-concepts/, 2024. W3C Recommendation, 28 March 2024.