Automatically Generating Human Readable Documentation for Ontology Design Patterns

Cogan Shimizu ✉ and Pascal Hitzler

Data Semantics Lab, Kansas State University, USA
{coganmshimizu, hitzler}@ksu.edu

Abstract. Keeping documentation consistent across deliverables is frequently a hassle. This program is a method for extracting and utilizing the common documentation annotations and those more specific to ontology design patterns from OWL files and rendering via LaTeX to provide a human readable, shareable PDF.

1 Motivation

Documentation is a critically important component of any system. This especially holds true as the complexity of that system grows. As knowledge graphs (KG) grow both in size and complexity, and commensurately popularity, it is important to meet the demand for in-depth documentation of the KG's schema in order to promote its (re)use. This program is used to generate human readable documentation (in the form of a pdf) from the annotations that are traditionally used for documentation purposes within an OWL file.

Automatic documentation generators are not new (e.g., [4]). The added benefit of this particular software is a focus on pattern-based ontologies (e.g. those produced by composing ontology design patterns [3] through the Modular Ontology Modeling methodology [11]) and the specific annotations that describe their modular structure [7], and the extended annotation vocabulary that describes the intended and empirical uses of the modules [6].

2 Implementation Details

This program is implemented in Python and depends on the RDFlib library. It expects an OWL file and extracts the relevant annotations to the OWL file from the Ontology Design Pattern Representation Language (OPLa) [7,6] in order to generate the sections of the documentation. The annotations can be done manually (e.g. through Protégé[1]), but can also be generated using tools such

[1] See https://protege.stanford.edu/.

as the OPLaTab [12] or CoModIDE [10]. The output is a well-formed LaTeX document.

The current structure is the title page, populated with top level annotations such as dc:creator, a primer chapter talking about schema diagrams and axioms (authored by Pascal Hitzler and can be configured off), and then generates the following sections, when the appropriate annotations are present. Where appropriate, it will also attempt to find figures, such as Schema Diagrams and examples, via the opla-sd:hasSchemaDiagram, opla-sd:hasShortcutDiagram, opla-sd:hasExampleDiagram annotations.

- **Overview:** is generated using the opla-cp: annotation. The overview section describes the overall purpose of the ontology (or pattern).
- **Use cases:** is generated using the opla-cp:hasScenario annotation. The program currently expects the title of the use case to be separated from the text of the use case by a newline character. Use cases are the context of expected interactions between agents (e.g., users or programs) and the ontology.
- **Competency Questions:** is generated using the opla-cp:hasCompetencyQuestion annotation. The program will create an enumerated (alphabetized) list of the competency questions for the ontology. Competency questions are the fundamental interactions that constitute a use case.
- **Formalization:** is generated using the opla-core:Axiom opla-core:isNativeTo owl:Ontology annotation pattern. Currently we support the formalization and explanation of 17 the axiom patterns that can be found in [9,2].
- **Submodules:** is generated using the opla-core:isNativeTo annotation. The program identifies any submodules and links to the appropriate section in the documentation. A submodule can be considered to be a fully defined and described component of the ontology.
- **Views:** is generated using the opla-core:hasShortcut annotation. Shortcuts are "optional" role chains throughout the ontology that may help for alignments or data publishers. These are described in the same manner as the Formalization section.
- **Entanglements:** is generated using the opla-cp:hasEntanglement annotation. Entanglements may be viewed as alignments or implementations. This section is generally more useful for patterns where multiple implementations may exist (e.g., SOSA/SSN [1] vs. RealEstateCore [5]).
- **Examples:** is generated using the opla-cp:hasExample annotation. These are just example triples encoded into a literal and represented in a Verbatim environment.
- **Remarks, Questions, Issues, Known Problems:** is generated using the opla-cp:hasConsequence, opla-cp:hasRemark, or rdfs:Comment annotations. This a catchall section.

Additional information on the implementation, installation, and use of these tools (including CoModIDE) can be found in our online portal.[2] The source

[2] See `http://daselab.org/content/modular-ontology-engineering-portal`.

(documentation-generator/), licensing (license.txt), and example execution(documentation/), containing two working patterns, for this demonstration can be found on GitHub.[3]

3 Demonstration

We expect to provide a live demonstration of the program's functionality in a tutorial-like format. We will provide several additional pre-annotated patterns beyond what is currently included in the online repository for this demonstration in order to show how different annotations result in changed documentation. In particular, we will walk potential users through the execution process of this program, customize the final output, and illustrate integration with the surrounding tool-suite.

4 Next Steps

There are many additional directions that we may take this work. First and foremost, we intend to incorporate additional insights from [8] on how to better document ontology design patterns to potential users. This includes a more comprehensive modeling of use-cases and competency questions and the provenance and lineage thereof. We also wish to see how it may be used to document so-called Modular Ontology Design Libraries [13] where many patterns may interrelate (e.g., through opla-core:hasRelatedPattern).

While this program represents a prototypical demonstration of the ability to generate useful, human-readable documentation from annotations, there are several ways to improve this process: for better code quality and improved, resultant machine-readable (and human-readable) documentation. For the former, by separating static portions of the documentation (e.g., the primer chapter or LaTeX specific formatting) For the latter, by outputting to HTML. Finally, we expect to support additional documentation annotations.

Acknowledgement. The authors acknowledge support by the National Science Foundation under Grant 2033521 A1: KnowWhereGraph: Enriching and Linking Cross-Domain Knowledge Graphs using Spatially-Explicit AI Technologies.

References

1. Compton, M., Barnaghi, P., Bermudez, L., García-Castro, R., Corcho, O., Cox, S., Graybeal, J., Hauswirth, M., Henson, C., Herzog, A., Huang, V., Janowicz, K., et al.: The ssn ontology of the w3c semantic sensor network incubator group. Web Semant. **17**(C), 25–32 (Dec 2012)

[3] See `https://github.com/KnowWhereGraph/documentation-generator`.

2. Eberhart, A., Shimizu, C., Chowdhury, S., Sarker, M.K., Hitzler, P.: Expressibility of OWL axioms with patterns. In: Verborgh, R., Hose, K., Paulheim, H., Champin, P., Maleshkova, M., Corcho, Ó., Ristoski, P., Alam, M. (eds.) The Semantic Web - 18th International Conference, ESWC 2021, Virtual Event, June 6-10, 2021, Proceedings. Lecture Notes in Computer Science, vol. 12731, pp. 230–245. Springer (2021). https://doi.org/10.1007/978-3-030-77385-4_14, `https://doi.org/10.1007/978-3-030-77385-4_14`
3. Gangemi, A., Presutti, V.: Ontology design patterns. In: Staab, S., Studer, R. (eds.) Handbook on Ontologies, pp. 221–243. International Handbooks on Information Systems, Springer (2009)
4. Garijo, D.: WIDOCO: A wizard for documenting ontologies. In: d'Amato, C., Fernández, M., Tamma, V.A.M., Lécué, F., Cudré-Mauroux, P., Sequeda, J.F., Lange, C., Heflin, J. (eds.) The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part II. Lecture Notes in Computer Science, vol. 10588, pp. 94–102. Springer (2017). https://doi.org/10.1007/978-3-319-68204-4_9, `https://doi.org/10.1007/978-3-319-68204-4_9`
5. Hammar, K., Wallin, E.O., Karlberg, P., Hälleberg, D.: The realestatecore ontology. In: Ghidini, C., Hartig, O., Maleshkova, M., Svátek, V., Cruz, I.F., Hogan, A., Song, J., Lefrançois, M., Gandon, F. (eds.) The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part II. Lecture Notes in Computer Science, vol. 11779, pp. 130–145. Springer (2019). https://doi.org/10.1007/978-3-030-30796-7_9, `https://doi.org/10.1007/978-3-030-30796-7_9`
6. Hirt, Q., Shimizu, C., Hitzler, P.: Extensions to the ontology design pattern representation language. In: Janowicz, K., Krisnadhi, A.A., Poveda-Villalón, M., Hammar, K., Shimizu, C. (eds.) Proceedings of the 10th Workshop on Ontology Design and Patterns (WOP 2019) co-located with 18th International Semantic Web Conference (ISWC 2019), Auckland, New Zealand, October 27, 2019. CEUR Workshop Proceedings, vol. 2459, pp. 76–75. CEUR-WS.org (2019), `http://ceur-ws.org/Vol-2459/short2.pdf`
7. Hitzler, P., Gangemi, A., Janowicz, K., Krisnadhi, A.A., Presutti, V.: Towards a simple but useful ontology design pattern representation language. In: Blomqvist, E., Corcho, Ó., Horridge, M., Carral, D., Hoekstra, R. (eds.) Proceedings of the 8th Workshop on Ontology Design and Patterns (WOP 2017) co-located with the 16th International Semantic Web Conference (ISWC 2017), Vienna, Austria, October 21, 2017. No. 2043 in CEUR Workshop Proceedings (2017)
8. Karima, N., Hammar, K., Hitzler, P.: How to document ontology design patterns. In: Hammar, K., Hitzler, P., Lawrynowicz, A., Krisnadhi, A., Nuzzolese, A., Solanki, M. (eds.) Advances in Ontology Design and Patterns, Studies on the Semantic Web, vol. 32, pp. 15–28. IOS Press, Amsterdam (2017)
9. Sarker, M.K., Krisnadhi, A., Carral, D., Hitzler, P.: Rule-based OWL modeling with rowltab protégé plugin. In: Blomqvist, E., Maynard, D., Gangemi, A., Hoekstra, R., Hitzler, P., Hartig, O. (eds.) The Semantic Web - 14th International Conference, ESWC 2017, Portorož, Slovenia, May 28 - June 1, 2017, Proceedings, Part I. Lecture Notes in Computer Science, vol. 10249, pp. 419–433 (2017). https://doi.org/10.1007/978-3-319-58068-5_26, `https://doi.org/10.1007/978-3-319-58068-5_26`
10. Shimizu, C., Hammar, K., Hitzler, P.: Modular graphical ontology engineering evaluated. In: Harth, A., Kirrane, S., Ngomo, A.N., Paulheim, H., Rula, A., Gen-

tile, A.L., Haase, P., Cochez, M. (eds.) The Semantic Web - 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31-June 4, 2020, Proceedings. Lecture Notes in Computer Science, vol. 12123, pp. 20–35. Springer (2020). https://doi.org/10.1007/978-3-030-49461-2_2, `https://doi.org/10.1007/978-3-030-49461-2_2`

11. Shimizu, C., Hammar, K., Hitzler, P.: Modular ontology modeling. Tech. rep. (2021), `https://daselab.cs.ksu.edu/publications/modular-ontology-modeling`, under Review.
12. Shimizu, C., Hirt, Q., Hitzler, P.: A protégé plug-in for annotating OWL ontologies with opla. In: Gangemi, A., Gentile, A.L., Nuzzolese, A.G., Rudolph, S., Maleshkova, M., Paulheim, H., Pan, J.Z., Alam, M. (eds.) The Semantic Web: ESWC 2018 Satellite Events - ESWC 2018 Satellite Events, Heraklion, Crete, Greece, June 3-7, 2018, Revised Selected Papers. Lecture Notes in Computer Science, vol. 11155, pp. 23–27. Springer (2018). https://doi.org/10.1007/978-3-319-98192-5_5, `https://doi.org/10.1007/978-3-319-98192-5_5`
13. Shimizu, C., Hirt, Q., Hitzler, P.: MODL: A modular ontology design library. In: Janowicz, K., Krisnadhi, A.A., Poveda-Villalón, M., Hammar, K., Shimizu, C. (eds.) Proceedings of the 10th Workshop on Ontology Design and Patterns (WOP 2019) co-located with 18th International Semantic Web Conference (ISWC 2019), Auckland, New Zealand, October 27, 2019. CEUR Workshop Proceedings, vol. 2459, pp. 47–58. CEUR-WS.org (2019), `http://ceur-ws.org/Vol-2459/paper4.pdf`