

How Much OWL Do You Need to Know to Make Sense of Building Ontologies?

María Poveda-Villalón^{1,*}, Sergio Carulli-Pérez¹ and Raúl García-Castro¹

¹Universidad Politécnica de Madrid, Madrid, Spain

Abstract

The fact that ontologies are considered shareable and reusable components has always been claimed as one of their main advantages. While reusing ontologies is promised to reduce time and resources invested during ontology development, the experience shows that reusing ontologies also involves a costly effort. In this work, we first analyse how ontologies representing the building domain currently reuse each other. Finally, we detect existing Web Ontology Language (OWL) modelling patterns in the ontologies analysed and generate a graphical pattern library to facilitate reuse of the observed patterns.

Keywords

Ontology, Ontology reuse, Ontology design pattern

1. Introduction

The reusable characteristic of ontological resources, such as ontologies or ontology design patterns, has been claimed to be one of the main benefits of using and developing ontologies. Ontology reuse is usually done by hard reuse, that is, by means of using the `owl:imports` statement, or by soft reuse, that is, by referencing the reused ontology element Uniform Resource Identifiers (URIs) [1].

However, the ontology reuse activity is costly and can be hampered by several factors such as lack of licence, failures in ontology availability, lack of good quality documentation, etc. as reported by [1].

Previous studies have analysed how ontologies are reused independently of the domain [2] and also focus on particular domains such as biology [3]. In this work, our aim is not only to observe how ontologies are reusing each other, but also to extract common patterns repeated along ontologies. More precisely, in this work we aim to answer the following questions:

- How and to what extent are ontologies being reused in the building domain?
- What are the ontology design patterns used within the ontologies in the building domain?

To do so, we analysed 18 available ontologies about buildings following the process described in Section 3 and observed that there is a low rate of reuse between ontologies (Section 4).


One of the main difficulties to reuse ontologies is the effort needed to understand them, and when reusing submodules there is an additional effort to prune the model. In this sense, it has been claimed that having ontology design patterns commonly used in domain ontologies could alleviate the ontology reuse activity [4]. For this reason, our second question focuses on the specific patterns (those implemented using exact URIs in different ontologies) that appear in the building ontologies. In principle, once these patterns are identified, they can be coded and shared with the community. Such patterns could be shared as OWL building blocks, as, for example, in the Ontology Design Patterns (ODP) Portal¹, or by diagram libraries that can be converted into OWL code to facilitate reuse in a graphical way, as in the case of the Chowlk notation libraries [5].

LDAC 2024: 12th Linked Data in Architecture and Construction Workshop, June 13–14, 2024, Bochum, Germany

*Corresponding author.

✉ m.poveda@upm.es (M. Poveda-Villalón); sergiomario.carulli.perez@upm.es (S. Carulli-Pérez); r.garcia@upm.es (R. García-Castro)

ORCID iD 0000-0003-3587-0367 (M. Poveda-Villalón); 0000-0002-0421-452X (R. García-Castro)

 © 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

¹http://ontologydesignpatterns.org/wiki/Main_Page

During this study, no specific patterns have been found; however, common OWL patterns are detected and a graphical library containing the generalised ontology design patterns has been created. This library compiles the OWL constructs combination that appears in the building domain ontologies.

In conclusion, this study is quite strict in the sense of pattern definition, as the matching is done at the URI level instead of at the concept level, for example by examining the attached labels to an URI. Although the current work has been useful to provide a library of generalised patterns, it could be extended by incorporating ontology alignment techniques to find specific patterns (Section 5).

2. Related work

Some studies have analysed how ontologies are reused in general ontologies [6] and in particular domains such as biology [3]. In this work, we focus on the available ontologies related to the buildings domain.

Regarding the construction domain, Schenider [7] presented in 2019 an approach to compare alignments between ontologies generated manually with automatic ones to test how reliable are ontology matching algorithms for this domain. Schenider's work could be considered complementary to this study as it focus on subsumption and equivalences between names entities, either classes or properties, while our patterns focus on anonymous classes representing axioms.

Previous works have analysed the use of OWL construct in the Semantic Web [8] [9], however they did not consider the combination of constructs to identified repeated patterns. A recent study [10] focuses on the extraction of "Conceptual Components" that are implemented in different ways by different ontologies. This approach differs from ours in the sense that the Conceptual Components represent a cluster of several implementations (called "Observed Ontology Design Patterns", however, this notion of patterns does not imply repetition of the implementation, therefore it can also be considered sub-modules in some cases) in particular ontologies. In this sense, the Conceptual Components do not provide the corresponding code, but link to the code to the 'Observed Ontology Design Patterns'.

3. Methodology

To carry out this study, the workflow followed in Figure 1 has been followed. As can be observed, the first step is to collect the ontologies to be analysed. In this case, we have compiled ontologies about buildings from Linked Open Vocabularies [11] and the Linked Building Data Community Group GitHub repository about ontologies². After removing duplicates and discarding unavailable ontologies, the list of 18 ontologies considered is presented in Table 1.

Once the ontologies have been selected and downloaded, several processing steps have been carried out. First, in order to know how building ontologies reuse each other, it has been observed in which ontologies hard reuse is present, by means of using `owl:imports`, and which ontologies implement soft reuse by referencing other ontology elements URIs (note that metadata references are ignored). Second, all ontologies are inspected to identify repeating patterns. Prior to pattern detection, the system identifies structures in the ontologies code stemming from the classes' `rdfs:subClassOf`, `owl:equivalentClass` and `owl:disjointWith` statements in which the object is a restriction, that is, the object is not a named class. These structures can be specific, considering the exact URIs for the ontology elements (see Figure 2(a)) or general structures defined by generalising the particular URIs to the OWL or RDF(S) type of elements (see Figure 2(b)).

To identify patterns, our system counts how many times the exact structure appears in all the ontologies. This is done for the specific and general structures to identify specific and general patterns, respectively.

Finally, taking as input the general patterns identified, a Chowlk library has been created to ease the

²<https://github.com/w3c-lbd-cg/ontologies>

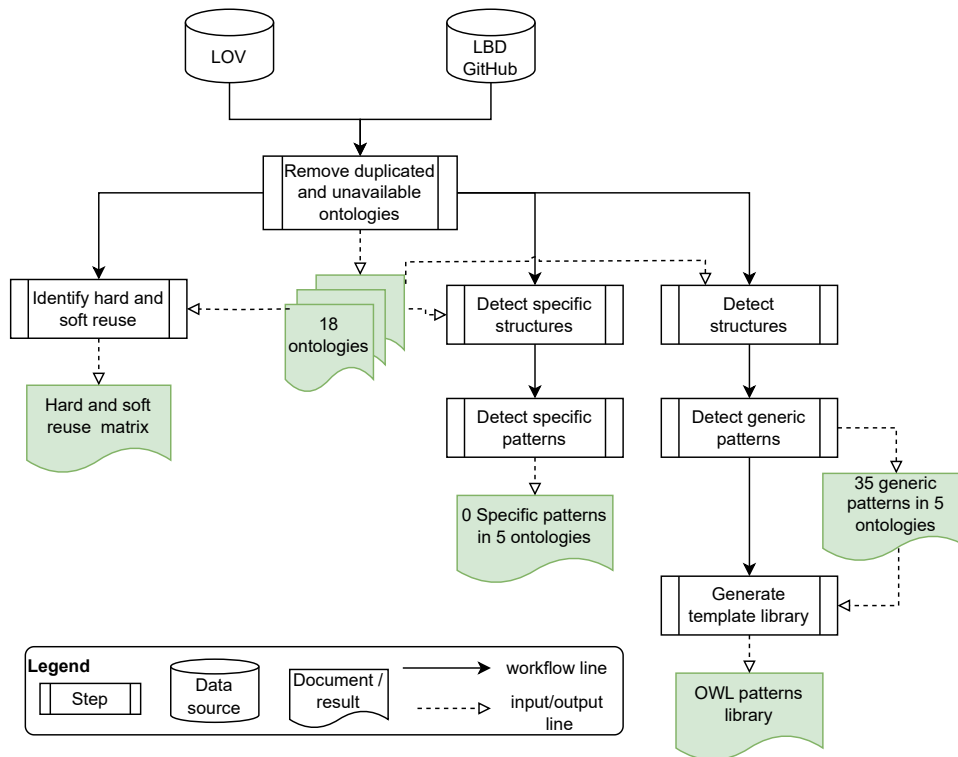


Figure 1: Workflow followed.

Table 1
Ontologies analyzed (alphabetical order)

Prefix	Ontology Title	Ontology URI
bcom	Building Concrete Monitoring Ontology	https://w3id.org/bcom
beo	Building Element Ontology	https://pi.pauwel.be/voc/buildingelement
bimerr-op	Occupancy Profile ontology	http://bimerr.iot.linkeddata.es/def/occupancy-profile#
bpo	Building Product Ontology	https://w3id.org/bpo
bot	Building Topology Ontology	https://w3id.org/bot#
brick	Brick	https://brickschema.org/schema/Brick#
fog	File Ontology for Geometry formats	https://w3id.org/fog
ifc	ifcOWL ontology (IFC4_ADD1)	https://w3id.org/ifc/IFC4_ADD1
IFC4	list of properties extracted from IFC4 psets	https://w3id.org/product/props/
jup	Ontology of Building Accessibility	http://w3id.org/charta77/jup
mep	Distribution Element Ontology	https://pi.pauwel.be/voc/distributionelement
omg	Ontology for Managing Geometry	https://w3id.org/omg#
rami	Reference Architecture Model	http://iais.fraunhofer.de/vocabs/rami#
rec	RealEstateCore	https://w3id.org/rec
rooms	Buildings and Rooms Vocabulary	http://vocab.deri.ie/rooms
s4bldg	SAREF extension for building	https://saref.etsi.org/saref4bldg/
sbeo	Smart Building Evacuation Ontology	https://w3id.org/sbeo
seasbo	The SEAS Building Ontology	https://w3id.org/seas/BuildingOntology

use of common OWL structures in ontologies.³ It should be noted that the generation of this library has been done semi-automatically.

4. Results

Regarding the hard and soft reuse observed in the analysed ontologies, it can be observed that the analyzed ontologies do not reuse much each other and the most common reuse technique is soft reuse.

³Exception: pattern 32 is not included in the library as it is more expressive than the Chowlk notation at the time of writing this paper.

```

Ontology: s4bldg.rdf
Structure: s4bldg.rdf-1
s4bldg:Actuator
  | rdfs:subClassOf
  | | owl:Restriction
  | | | owl:allValuesFrom
  | | | | xsd:string
  | | | owl:onProperty
  | | | | s4bldg:failPosition

```

(a) Example of an specific structure

```

Ontology: s4bldg.rdf
Structure: s4bldg.rdf-1
owl:Class
  | rdfs:subClassOf
  | | owl:Restriction
  | | | owl:allValuesFrom
  | | | | rdfs:Datatype
  | | | owl:onProperty
  | | | | owl:DatatypeProperty

```

(b) Example of a general structure

Figure 2: Structure examples

The following list shows for each analysed ontology which ontologies it imports (hard reuse) and which ontologies are referenced (soft reuse). In the case of soft reuse, the number of ontology elements referenced are included between “()”. The ontologies analyzed in this study are identified by their prefix while ontologies not included in the analysis are identified by their namespaces.

With regard to the identified patterns, we can observe that no specific patterns are repeated. This means that the specific structures identified in the ontologies are not repeated in any other ontology. This result was expected after analysing the soft reuse, as this study seeks the repeated appearance of same URIs in equal structures. For this reason, that exact URIs are matched, we cannot claim that similar concepts are not defined in more than one ontology in a similar way.

Looking now at the general patterns extracted⁴ we can see that 35 patterns are identified among 5 ontologies. These patterns are only present in 5 ontologies because the rest of ontologies did not include class descriptions using OWL restrictions, boolean combinations or cardinalities. That is, there was no classes’ `rdfs:subClassOf`, `owl:equivalentClass` and `owl:disjointWith` statements whose object was a restriction or a blank node instead of a named class in 13 ontologies.

The identified general patterns are depicted in Figure 3, Figure 4 and Figure 5 following the Chowlk notation. In such figures there are three numbers on the left of each pattern; the top number represents the pattern identifier, according to the numeration in the supplementary material; the middle number indicates in how many ontologies appears the pattern; and the bottom number indicates how many repetitions of the pattern have been observed considering all ontologies.

⁴The data resulting from the study is available at <https://doi.org/10.5281/zenodo.10997320>.

Table 2
Ontologies reused

Ontology	Reuse	Reused Ontologies
mep	Hard	-
	Soft	-
bimerr-op	Hard	-
	Soft	http://www.w3.org/2004/02/skos/core# (6) https://w3id.org/def/saref4building# (6) https://w3id.org/saref# (4) http://bimerr.iot.linkeddata.es/def/building# (2) http://www.w3.org/2006/time# (6) http://xmlns.com/foaf/0.1# (1)
brick	Hard	-
	Soft	http://qudt.org/schema/qudt/ (9) http://www.w3.org/2004/02/skos/core# (5) http://schema.org/ (2) http://data.ashrae.org/bacnet/2020# (25) http://data.ashrae.org/standard223# (2) http://www.w3.org/ns/sosa/ (2) http://www.w3.org/2006/vcard/ns# (2)
fog	Hard	-
	Soft	omg (3) http://xmlns.com/foaf/0.1/ (1) http://www.opengis.net/ont/geosparql# (2)
IFC4	Hard	-
	Soft	http://xmlns.com/foaf/0.1/ (1) https://w3id.org/product/Actuator# (3) https://w3id.org/product/Controller# (5) https://w3id.org/product/CableCarrierSegment# (4) https://w3id.org/product/Damper# (4) https://w3id.org/product/BuildingElementProxy# (1) https://w3id.org/product/AudioVisualAppliance# (8) https://w3id.org/product/Annotation# (3) https://w3id.org/product/CableSegment# (1) https://w3id.org/product/Boiler# (2) https://w3id.org/product/Covering# (2) https://w3id.org/product/CooledBeam# (2)
bcom	Hard	-
	Soft	http://www.w3.org/2006/vcard/ns# (2) http://xmlns.com/foaf/0.1/ (2)
rami	Hard	-
	Soft	http://purl.oclc.org/NET/ssnx/ssn# (2) http://www.wurvoc.org/vocabularies/om-1.8/ (2) http://www.w3.org/ns/prov# (1) http://xmlns.com/foaf/spec/ (1)
s4bldg	Hard	-
	Soft	ifc (5) https://saref.etsi.org/core/ (11) http://www.w3.org/2003/01/geo/wgs84_pos# (1)
jup	Hard	-
	Soft	http://www.w3.org/2004/02/skos/core# (5) http://www.w3.org/ns/org# (4) http://purl.org/dc/terms/ (4) http://xmlns.com/foaf/0.1/ (5) http://schema.org/ (3) http://www.w3.org/ns/regorg# (1) http://dbpedia.org/ontology/ (1) http://rdfs.org/sioc/ns# (1) http://www.w3.org/ns/adms (1)
beo	Hard	-
	Soft	-
bpo	Hard	-
	Soft	http://xmlns.com/foaf/0.1/ (2) http://schema.org/ (8) http://qudt.org/schema/qudt# (1) https://w3id.org/seas/ (2) http://purl.org/goodrelations/v1# (1)
bto	Hard	-
	Soft	https://schema.org/ (1) http://purl.org/vocommons/voaf# (1) http://www.w3.org/2006/vcard/ns# (1)
ifc	Hard	https://w3id.org/express
	Soft	https://w3id.org/list# (5)
sbeo	Hard	-
	Soft	seasbo (35) http://www.w3.org/ns/sosa/ (8) https://w3id.org/seas/ (36) http://purl.org/ontology/olo/core# (10) http://xmlns.com/foaf/0.1/ (5)
omg	Hard	-
	Soft	https://w3id.org/seas/# (1) https://w3id.org/opm# (2) http://www.w3.org/ns/prov# (1) https://w3id.org/seas/ (1) http://xmlns.com/foaf/0.1/ (1)
rec	Hard	http://datashapes.org/dash https://brickschema.org/schema/1.3/Brick https://w3id.org/rec/brickpatches
	Soft	-
rooms	Hard	-
	Soft	http://www.w3.org/ns/adms# (3) http://xmlns.com/foaf/0.1/ (7)
seasbo	Hard	-
	Soft	http://xmlns.com/foaf/0.1/ (2) http://purl.org/vocommons/voaf# (1)

Figure 3 shows the patterns stemming from a `rdfs:subClassOf` declaration followed by universal (denoted with “(all)”) or existential (denoted with “(some)”) restrictions for object and datatype properties. For example, pattern 4 represents a `rdfs:subClassOf` restriction involving an existential constraint over a given object property. These patterns can involve property characteristics such as `owl:FunctionalProperty` (denoted with “(F)”) or `owl:TransitiveProperty` (denoted with “(T)”), as for example patterns 6, 9, 14 or 26 for object properties and 1 and 7 for datatype properties. Finally, we can observe that some patterns, 2 and 26, combine the restrictions with a union of classes instead of named classes.

Figure 4 depicts the patterns stemming from a `rdfs:subClassOf` declaration followed by non-qualified cardinality (denoted with “(X.Y)”) or qualified cardinality (denoted with “[X.Y]”) restrictions for object and datatype properties. For example, pattern 10 represents a `rdfs:subClassOf` restriction involving a minimum non-qualified cardinality constraint over a given object property and pattern 27 represents a `rdfs:subClassOf` restriction involving an exact qualified cardinality constraint over a given object property. Equally to the previous case, these patterns can involve property characteristics,

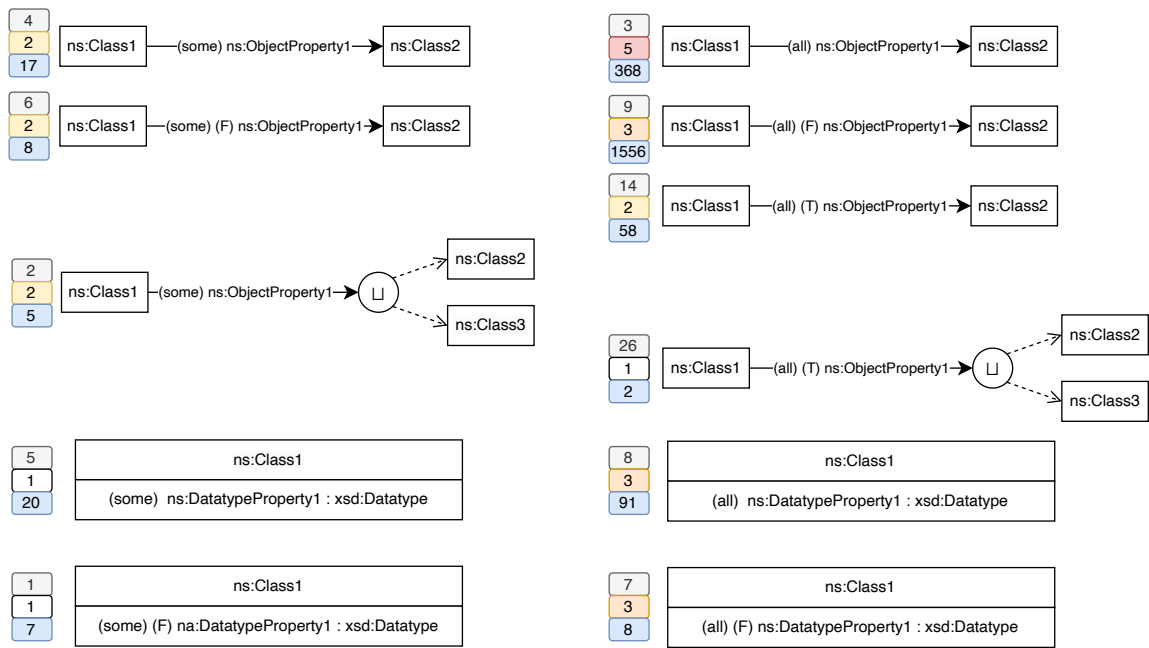


Figure 3: `rdfs:subClassOf` patterns observed involving existential and universal restrictions.

as, for example, pattern 31 that represents an exact non-qualified cardinality over a given datatype property.

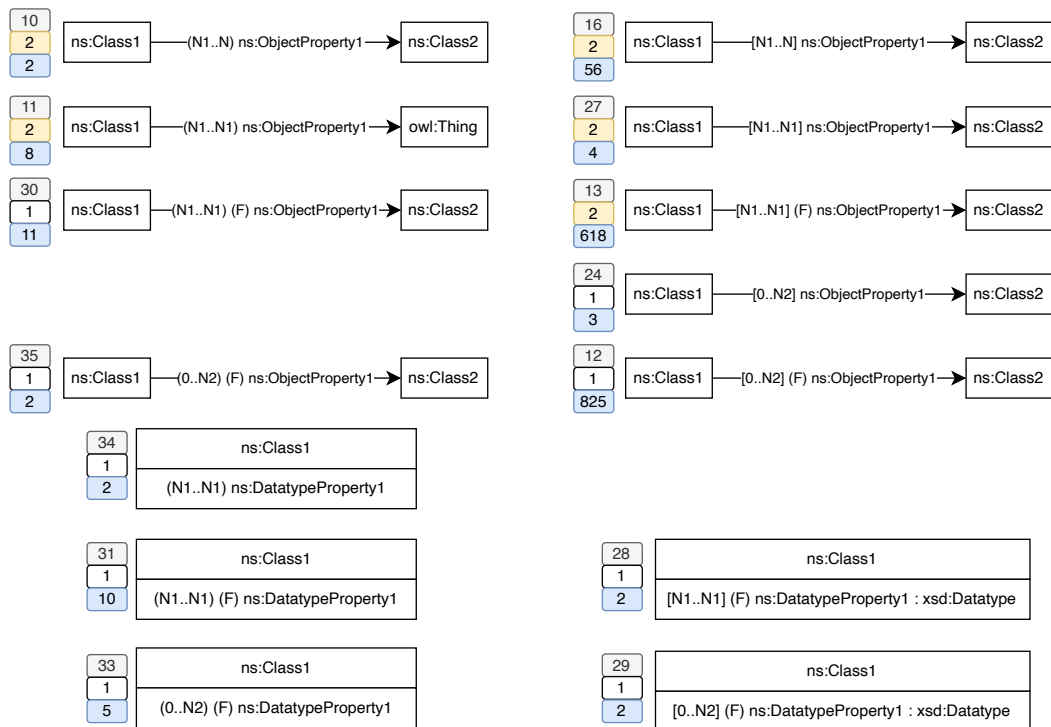


Figure 4: `rdfs:subClassOf` patterns observed involving cardinalities.

Figure 5 depicts the patterns stemming from a `rdfs:subClassOf` declaration followed by nested universal or existential restrictions for object properties. In all the cases the properties involved in the restrictions are functional properties and it can be observed that for patterns 19 and 21 the last nested

restriction involves exact qualified cardinalities instead of existential axioms. These patterns appear only in ifc.

Figure 5 also shows two patterns, 15 and 25 that represent a subclass made of the union of classes (`owl:unionOf`) and an equivalent class defined by listing the potential individual (`owl:oneOf`) respectively.

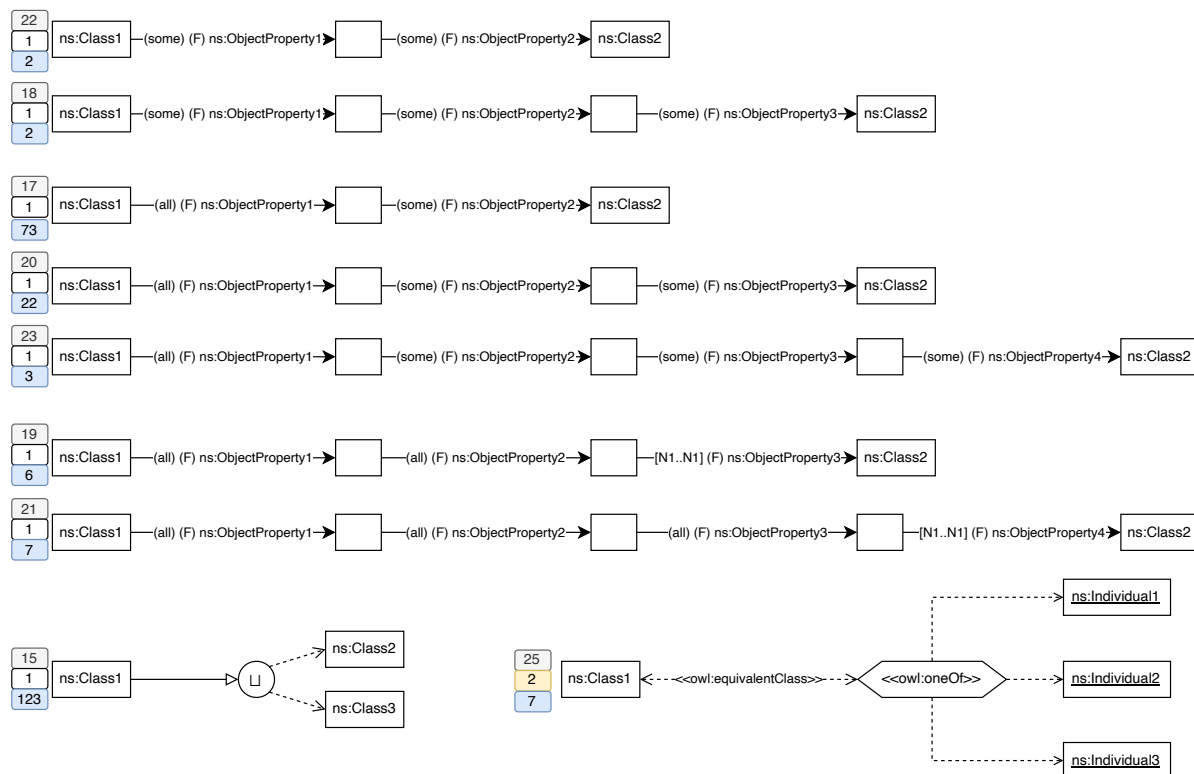


Figure 5: `rdfs:subClassOf` patterns observed involving nested existential and universal restrictions and other patterns.

5. Conclusions and future work

This work has shown the degree of reuse between ontologies in the building domain, which is quite low; however, the reasons for this are not explored in detail. In addition, regarding the observed reuse, it should be analysed whether it corresponds to references to other domains.

On the one hand, and in line with the low level of soft reuse between ontologies, we have observed that there are no specific domain patterns between the ontologies. However, this conclusion could be due to the fact that this study compares URIs rather than labels of concepts. In this sense, future lines of work include reproducing the study considering labels and annotations or even running ontology matching algorithms before searching for patterns.

On the other hand, we have extracted 35 general patterns that answer, in a graphical way, how much OWL knowledge is needed to understand the analysed ontologies. In addition, a patterns library for Chowlk has been developed to facilitate the reuse of such patterns.

During this process two valuable lessons have been learnt about pattern generation using the Chowlk notation. First, we have detected one case of an OWL structure (pattern 32) that is currently not supported by the notation, which implies an impact in the notation that should be extended. Second, we observed that the process of extracting patterns does not take into account whether the involved object and datatype properties have domain or range defined, information that is needed to generate the Chowlk visualisations. At this point, we arbitrarily decided to draw the patterns using the notation

for domain and range definition for this version; however, it is planned for the next version to keep track of that information at the structure level.

Acknowledgments

This work has been supported by Horizon 2020 research and innovation programme under grant agreements no. 958310 (COGITO) and 101016854 (AURORAL) and by the Madrid Government (Comunidad de Madrid-Spain) under the Multiannual Agreement with the Universidad Politécnica de Madrid in the Excellence Programme for University Teaching Staff, in the context of the V PRICIT (Regional Programme of Research and Technological Innovation).

References

- [1] M. Fernández-López, M. Poveda-Villalón, M. C. Suárez-Figueroa, A. Gómez-Pérez, Why are ontologies not reused across the same domain?, *Journal of Web Semantics* 57 (2019) 100492. URL: <https://www.sciencedirect.com/science/article/pii/S1570826818300726>. doi:<https://doi.org/10.1016/j.websem.2018.12.010>.
- [2] M. Poveda-Villalón, A reuse-based lightweight method for developing linked data ontologies and vocabularies, in: *Extended Semantic Web Conference*, Springer, 2012, pp. 833–837.
- [3] G.-Q. Zhang, M. R. Kamdar, T. Tudorache, M. A. Musen, A systematic analysis of term reuse and term overlap across biomedical ontologies, *Semant. Web* 8 (2017) 853–871. doi:<https://doi.org/10.3233/SW-160238>.
- [4] A. Gangemi, V. Presutti, Ontology design patterns, in: *Handbook on ontologies*, Springer, 2009, pp. 221–243.
- [5] 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. doi:https://doi.org/10.1007/978-3-031-06981-9_20.
- [6] M. Poveda-Villalón, M. C. Suárez-Figueroa, A. Gómez-Pérez, The landscape of ontology reuse in linked data, in: *Proceedings Ontology Engineering in a Data-driven World (OEDW 2012)*, 2012.
- [7] G. F. Schneider, Automated ontology matching in the architecture, engineering and construction domain—a case study, in: *Proceedings of the 7th Linked Data in Architecture and Construction Workshop (LDAC)*, volume 2389, CEUR-WS. org Lisbon, Portugal, 2019, pp. 35–49.
- [8] N. Matentzoglou, S. Bail, B. Parsia, A snapshot of the owl web, in: H. Alani, L. Kagal, A. Fokoue, P. Groth, C. Biemann, J. X. Parreira, L. Aroyo, N. Noy, C. Welty, K. Janowicz (Eds.), *The Semantic Web – ISWC 2013*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 331–346.
- [9] B. Glimm, A. Hogan, M. Krötzsch, A. Polleres, Owl: Yet to arrive on the web of data?, *arXiv preprint arXiv:1202.0984* (2012).
- [10] L. Asprino, V. A. Carriero, V. Presutti, Extraction of common conceptual components from multiple ontologies, in: *Proceedings of the 11th on Knowledge Capture Conference*, 2021, pp. 185–192.
- [11] P.-Y. Vandenbussche, G. A. Atezing, M. Poveda-Villalón, B. Vatant, Linked Open Vocabularies (LOV): a gateway to reusable semantic vocabularies on the Web, *Semantic Web* 8 (2017) 437–452.