Modest Use of Ontology Design Patterns in a Repository of Biomedical Ontologies

Jonathan M. Mortensen, Matthew Horridge, Mark A. Musen, and Natalya F. Noy

Stanford Center for Biomedical Informatics Research
Stanford University, Stanford CA 94305, USA

Abstract. Ontology Design Patterns (ODPs) provide a means to capture best practice, to prevent modeling errors, and to encode formally common modeling situations for use during ontology development. Despite the popularity of ODPs and supposed positive effects from their use, there is scant empirical evidence of their level of adoption in real world ontologies or on their effectiveness. Knowing the goals of ODPs, they may assist in the development of large-scale biomedical ontologies. Before studying ODP effectiveness and applicability, we ask the following questions to understand better the landscape of ODP use: Are ODPs used in biomedical ontologies? Which patterns do the ontology developers use? In which ontologies? How frequently are patterns used? To answer these questions, we determined the adoption of ODPs from two popular ODP libraries among the ontologies in BioPortal, a large ontology repository that contains over 300 biomedical ontologies. We encoded 68 ODPs from two online libraries in the Ontology Pre-Processor Language, and, using these encodings, determined ODP prevalence in BioPortal ontologies. We found modest use of ODPs, with 33% of the ontologies containing at least one pattern. `Upper Level Ontology`, `Closure`, and `Value Partition` were the three most commonly used patterns, occurring in 20%, 9%, and 6% of the BioPortal ontologies, respectively. The low prevalence of ODPs may be due to lack of proper tooling, lack of user knowledge of and education about them, the age of the ontologies in the repository, or the specificity of some ODPs. We noted that there is a tension between the high expressivity of many ODPs and the goal of maintaining low expressivity of some biomedical ontologies. Additional tooling is necessary to make ODPs more accessible to domain experts. Furthermore, we suggest that ODPs may be developed in a bottom-up fashion, much like software-design patterns. [1]

Keywords: OWL, biomedical ontologies, BioPortal, Ontology Design Pattern, Ontology Pre-Processor Language

1 Ontology Design Patterns

There is a large body of research establishing and creating Ontology Design Patterns (ODPs) [11, 5]. Yet, there is little work to determine their use or effectiveness. In biomedicine, the development and use of ontologies are growing rapidly. This

[1] Accompanying online resources at http://www.stanford.edu/people/mortensen/odp

development process can be difficult and/or error prone. As such, ODPs would likely assist with this development process. In this study, as initial work in evaluating the effectiveness and applicability of ODPs in biomedical ontologies, we examine the prevalence of ODPs in a large corpus of ontologies related to biomedicine.

1.1 ODPs and ODP libraries

Software Design Patterns emerged in the 1990s, capturing recurring software design techniques seen in software [10]. Following a similar motivation, the Semantic Web community developed ODPs to alleviate some of the complexities in developing ontologies. ODPs, defined as "a modeling solution to solve a recurrent ontology design problems" [11], capture best practice and common modeling situations. The developers of ODPs suggest that by using the patterns, one can more easily avoid modeling errors, improve ontology quality, maintainability, and reuse [3].

ODPs have become quite popular recently, with multiple workshops held at ISWC, including one during ISWC 2012. There are two online catalogs of ODPs, the Manchester ODPs Public Catalog for bio-ontologies (MBOP) and OntologyDesignPatterns.org (ODP-Wiki) [9, 1]. These catalogs describe each pattern by the problem that it solves, the proposed solution, and the formal representation by which to instantiate the pattern. MBOP contains 17 patterns derived from its authors' experience in modeling ontologies in the biomedical domain and working with OWL-based ontologies in general. ODP-Wiki is a crowd-sourced effort to create an ODP library. The website owners ask for pattern submissions and then a committee reviews these submissions for approval. The approved patterns are then noted as such online. As of this writing, the committee has not approved any patterns but there are over 150 submissions.

Most of the submissions on ODP-Wiki are "content" ODPs. However, the site categorizes many other different types of ODPs. ODP-Wiki includes "structural" (methods to workaround for language expressivity limitations or define ontology shape/structure), "content" (modeling solutions for a specific domain), "correspondence" (methods to re-engineer an ontology to a different form or map an ontology to another), "reasoning" (patterns that enable one to obtain desired reasoning results),"presentation" (good practices for readability and usability), and "lexico-syntactic" (mapping linguistic structures to ontology entities) patterns—a categorization based on descriptions by Gamgemi and colleagues [11]. MBOP categorizes patterns as "extension" (workarounds for language expressivity limitations), "good practice" (good modeling practice) and "domain modeling" (solutions specific to certain domains). The "structural" classification encompasses the majority of the MBOP patterns. In this work, the structural and content ODPs are most relevant. Structural patterns are either logical, adding logical expressions not contained directly in the ontology language, or architectural, defining the structure/hierarchy of the ontology itself. Content ODPs model a specific domain situation, and are directly re-usable (i.e., they should be directly imported into an ontology and used). We omit lexico-syntactic, presentation, reasoning, and correspondence patterns from this work, as we cannot test for them using our framework.

Accompanying the MBOP, the Manchester group also developed the Ontology Pre-Processing Language (OPPL), both a language based on the Manchester syntax for OWL, and a software library, which leverages the OWL-API [14]. OPPL provides a way to manipulate ontologies, query for ODPs and instantiate them [16, 15, 2].

1.2 Biomedical Ontologies

In biomedicine, ontology use is rapidly increasing [7, 21]. For example, the National Center for Biomedical Ontology's BioPortal,[2] a repository of biomedical ontologies, contains over 300 ontologies and controlled terminologies as of this writing [18]. Biologists use biomedical ontologies to manage the large amount of data. Hospitals and related entities use them in the process of recording information about clinical encounters, during clinical decision support, billing, and so on. Because biomedical ontologies are often large and complex, developing them and ensuring that they conform to best practices poses a formidable challenge. Even the widely used ontologies frequently contain modeling errors. For instance, Rector and colleagues discovered modeling issues in SNOMED CT, one of the most widely used biomedical ontologies [19]. Researchers have found modeling errors in the National Cancer Institute thesaurus [8]. ODPs may be especially important in assisting with the challenge of modeling the large and complex biomedical domains while preventing errors. Before assessing the effect of using ODPs on the biomedical ontology modeling process, we first find the prevalence of ODPs in a large biomedical ontology corpus.

2 Methods

We quantified the use of ODPs from both MBOP and ODP-Wiki in BioPortal using OPPL and the OWL API. We first encoded ODPs in OPPL and validated their correctness (1) by using an expert opinion and (2) by comparing them to the examples in the library that served as a gold standard. We then obtained the ontologies from BioPortal, removing cases by use of predefined filtering criteria (See section 2.2). We normalized the ontologies to remove any differences in how they were specified, and then checked both the normalized and the original version for each encoded pattern, first filtering out patterns that cannot be represented in the ontology because it lacks the proper relations.

2.1 Pattern Selection

We used the following criteria to select the set of patterns for this study: The pattern must be (1) detectable, (2) non-trivial (that is, not just a template), (3) positively reviewed (if a review is available), and (4) available in a public catalog (in our case, either MBOP or ODP-Wiki). We use these criteria for the following reasons:

1. Using only detectable patterns may seem obvious; however, there are many patterns such as `n-ary relations`, or re-engineering patterns that cannot be detected without more information than just the ontology.
2. A template style pattern may not *require* the presence of any particular elements. Thus, it would be trivially present even if the ontology contained no elements of the pattern.
3. When available, we considered review information on ODP-Wiki. Poorly reviewed patterns may not yet be refined, making them difficult to encode, especially if they have a logical error.
4. We chose only publicly available patterns, as it is a necessary condition for both reproducibility of this study and the expectation of pattern re-use.

[2] http://bioportal.bioontology.org

Applying the criteria above to MBOP and ODP-Wiki, produced the following results:

- From the 17 patterns in MBOP, we used 15. The remaining 2 were undetectable
- From the 150 patterns in ODP-Wiki, we used 53. The remaining patterns were either optional or not positively reviewed.

Thus, we selected 68 patterns of 167.

2.2 Ontology Selection

From the available ontologies in BioPortal, we selected those ontologies that were publicly available, parseable, locatable (a file was easily obtainable), non-retired, available as a single file, and available as either OWL or OBO format. Applying these criteria to the 312 ontologies that were available in BioPortal as of January 2012, resulted in a set of 256 ontologies.

2.3 Pattern Encoding

OPPL and the OWL API are open-source standard libraries available to work with ontology design patterns and ontologies. We encoded the MBOP and ODP-Wiki patterns with OPPL. Some patterns could not be encoded in OPPL. Those patterns we encoded directly in Java using the OWL API. An example OPPL encoding of the `Value Partition` pattern (a way to specify a set of disjoint qualities the describe a concept) follows:

```
?v1:CLASS, ?v2:CLASS, ?param:CLASS
SELECT
ASSERTED ?param EquivalentTo ?v1 or ?v2,
ASSERTED ?v1 DisjointWith ?v2
BEGIN
ADD ?v1 subClassOf Thing
END;
```

In order to reduce computational complexity, we pruned pattern–ontology pairs by first checking whether the ontology contains the specific relationships between concepts that a given ODP requires. An ontology without those relationships cannot have the pattern as the catalog specifies it. Furthermore, for those patterns that could not occur in any ontology from our selection, based on the required relationships, we did not encode the pattern. In particular, many content patterns refer to specific relationships in the ontology. For example, according to ODP-Wiki, the pattern `Part Of` requires the relationship “isPartOf”. Thus, if an ontology does not have this relationship “isPartOf”, we know that it will not have the pattern. When searching, we disregard the namespace of any given pattern, in case the pattern simply uses a different namespace (i.e., we only match on the URI fragment, not including the namespace). One might consider searching with possible lexical variants of this relationship term to ensure one finds occurrences which capture the intension of the specified relationship. However, the point at which a given string no longer matches the initial string is not well defined. Furthermore, content ODPs directly import a small module, thus the relation should not vary across ontologies.

Table 1. Transforms applied exhaustively to an ontology to normalize it.

Axiom	Transformation
prop min 1 C	prop some C
prop exactly n C	prop min n C, prop max n C
prop value i	prop some i
Property in Anonymous Class	Simplify Property (Removing inverses) and re-insert
C1 and (C2 and C3)	C1 and C2 and C3
C1 or (C2 or C3)	C1 or C2 or C3
C1 EquivalentTo C2	C1 SubClassOf C2, C2 SubClassOf C1
C1 DisjointUnionOf C2 ... Cn	DisjointClasses: C2 ... Cn, C1 EquivalentTo (C2 ... Cn)
C1 or ... or Cn SubClassOf D1 and ... and Dn	C1 SubClassOf D1 ... Cn SubClassOf D1 ... C1 SubClassOf Dn ... Cn SubClassOf Dn
DisjointClasses: C1 ... Cn	Ci DisjointWith Cj for 1 <= i <j <=n

Finally, during encoding, to gather additional information about a pattern, we noted the OWL 2 description logic constructs each utilizes. We note these because certain constructs have higher expressivity requirements. Higher expressivity comes at a higher computational cost. This fact may provide insight as to why biomedical ontologies instantiate only certain patterns.

2.4 Normalization

When specifying an ontology, one can represent the same conceptualization using different language constructs. This phenomenon is particularly common in OWL. To prevent missing a pattern that may be specified in a slightly different language than that found in an ontology, we applied pattern detection to a normalized version of each ontology as well. Table 1 lists the normalizations that we performed. These transformations follow from the OWL 2 specification and convert 'syntactic sugar' conventions provided by the language to a standard form. For example, applying the transformations to `Dog EquivalentTo Canine` results in `Dog SubClassOf Canine` and `Canine SubClassOf Dog`. While one could use a reasoner to infer equality of certain constructs, this was not computationally feasible in this study.

2.5 Computation

After creating a list of encoded patterns, and of original and normalized versions of all ontologies in the study, we searched for each pattern in the ontologies using the encodings and the software libraries. Checking for 68 patterns in nearly 300 ontologies, some of which have tens of thousands of classes, is a computationally intensive task. A brute-force approach to this task was unreasonable, as a single run through all possible ontology–pattern pairs would have taken over a week. We performed a few optimizations to speed up the runtime of the experiment. First, as described before in section 2.3, we pruned the patterns first by searching whether or not the ontology contains the necessary relations. Next, we created a specialized cache to store computationally intensive search operations that are shared across multiple pattern

queries per ontology. Finally, we distributed the pattern search process to a large cluster of 20 nodes, each with 24 cores and 96GB of RAM. We do not attempt to count the number of times a pattern occurs in a given ontology, as doing so also increases complexity. We limit the running time of any ontology–pattern pair to one week. During our analysis, approximately 150 pairs did not complete. These were complex patterns and complex ontologies (e.g., FMA and `Normalization`).

2.6 Validation

A pattern that is encoded (incorrectly) in a way that is too general could lead to a false positive, matching any ontology. For example, encoding `Value Partition` without the EquivalentClass component would match any ontology that has disjoint classes, which is not a true instantiation of `Value Partition`. An incorrectly specified pattern would lead to a false negative, not matching any ontology even though the pattern is present. We therefore verified that the patterns were correctly encoded both by involving an expert and by comparing them against a reference standard. The pattern encodings were manually inspected by an author of this paper (MH). Second, where possible, we tested the patterns and software against reference examples provided by the catalogs. We expected to find each encoded pattern in its reference example. MBOP provided example OWL ontology implementations of every pattern. Because ODP-Wiki required the import of a specific ontology, the verification was trivial.

3 Results

Of the 68 patterns in the study, we encoded 8 patterns in OPPL, 5 with the OWL API, manually verified 7, and pruned 47 content ODPs that contained relationships that were not present in any of the ontologies in the study. These relationships included, for example, “isRegionFor”, “hasRTMSCode”, “participatingInEvent”. One of the patterns MBOP specifies is the `Upper Level Ontology` pattern. MBOP describes the pattern as good practice that allows one to integrate different ontologies in a grounded framework. To find the `Upper Level Ontology` pattern, we checked whether each ontology imported a unique upper-level concept from either of the following upper ontologies: DOLCE [12], BFO [13] or SUMO [17]. We manually verified the remaining 7 patterns that matched the necessary relations for a pattern. These patterns need not necessarily import the pattern, but only capture its intension (e.g., GALEN). For brevity, Table 2 presents only a list of the positive patterns and what ontologies instantiated them. We found that 14 patterns were present in at least one ontology. 33% of the OWL and OBO format ontologies in BioPortal contain a pattern, with `Upper Level Ontology` (20%), `Closure` (9%), and `Value Partition` (6%) most common. Other commonly used patterns include `Normalization` and `Composite Property Chaining`. Thirty ontologies included more than one pattern. The Ontology of Biomedical Investigations included the most patterns: `DefinedClassDescription`, `Closure`, `Value Partition`, `Sequence` and `Upper Level Ontology`.

Table 2: Patterns and the ontologies that instantiate them.

Composite Property Chaining (7 Ontologies)	
Model a double chain of properties, i.e. two chains that link four individuals.	
Brucellosis Ontology	SemanticScience Integrated Ontology
Infectious Disease Ontology	Skin Physiology Ontology
Influenza Ontology	SNOMED CT
RNA ontology	
Adapted SEP (1)	
Properly propagate partomony	
BioTop	
Closure (23)	
Simulate the Closed World Assumption in a concrete class	
Amino Acid	Kinetic Simulation Algorithm Ontology
BioAssay Ontology	Lipid Ontology
BioTop	NanoParticle Ontology
Bleeding History Phenotype	Neomark Oral Cancer-Centred Ontology
Bone Dysplasia Ontology	Ontology for Biomedical Investigations
Breast Cancer Grading Ontology	Ontology for disease genetic investigation
Cancer Research and Management ACGT Master Ontology	Ontology for Genetic Interval
Cognitive Atlas	Skin Physiology Ontology
DIKB-Evidence-Ontology	Subcellular Anatomy Ontology (SAO)
Gene Regulation Ontology	Suggested Ontology for Pharmacogenomics
IMGT-ONTOLOGY	Vaccine Ontology
Infectious Disease Ontology	
Defined Class Description (6)	
Create If-Then structures in OWL DL	
Adverse Event Reporting ontology	Ontology for Biomedical Investigations
Cancer Research and Management ACGT Master Ontology	Ontology for Drug Discovery Investigations
NanoParticle Ontology	SysMO-JERM
Value Partition (16)	
Model values of non-overlapping attributes exhaustively	
Adverse Event Reporting ontology	Influenza Ontology
Basic Formal Ontology	OBOE
Basic Vertebrate Anatomy	Ontology for Biomedical Investigations
BioTop	Ontology for disease genetic investigation
CAO	Ontology for Genetic Interval
Computer-based Patient Record Ontology	RNA ontology
General Formal Ontology	Situation-Based Access Control
Infectious Disease Ontology	Vaccine Ontology
Normalization (14)	

Ensure maintainability and explicit semantics by allowing polyhierarchy only through inference

Basic Formal Ontology	PMA 2010
Human developmental anatomy, abstract version	Proteomics Pipeline Infrastructure for CPTAC
IMGT-ONTOLOGY	SemanticScience Integrated Ontology
Mouse gross anatomy and development	Traditional Medicine Constitution Value Set
NIFSTD	Traditional Medicine Meridian Value Sets
OBO relationship types	Traditional Medicine Other Factors Value Set
Pilot Ontology	Traditional Medicine Signs and Symptoms Value Set

Upper Level Ontology (53)

Create ontologies with a consistent, philosophically grounded upper ontology

Adverse Event Reporting ontology	Mental Functioning Ontology
Basic Formal Ontology	Mosquito insecticide resistance
Basic Vertebrate Anatomy	NanoParticle Ontology
BioModels Ontology	Neomark Oral Cancer Ontology
BIRNLex	Neural ElectroMagnetic Ontologies
Bone Dysplasia Ontology	NIF Cell
Brucellosis Ontology	NIF Dysfunction
Cancer Chemoprevention Ontology	NIFSTD
Cancer Research and Management ACGT Master Ontology	NMR-instrument specific component of metabolomics investigations
CAO	Ontology for Biomedical Investigations
Cardiac Electrophysiology Ontology	Ontology for disease genetic investigation
Chemical Information Ontology	Ontology for Drug Discovery Investigations
Cognitive Paradigm Ontology	Ontology for General Medical Science
Computer-based Patient Record Ontology	Ontology for Genetic Interval
Drosophila development	Ontology for Parasite LifeCycle
eagle-i research resource ontology	Ontology of Data Mining
Electrocardiography Ontology	Ontology of Glucose Metabolism Disorder
FGED View	Ontology of Medically Related Social Entities
Gene Regulation Ontology	Phenotypic quality
General Formal Ontology	RadLex
General Formal Ontology: Biology	RNA ontology
Host Pathogen Interactions Ontology	Skin Physiology Ontology
IEDB View	Sleep Domain Ontology
Infectious Disease Ontology	Subcellular Anatomy Ontology (SAO)
Influenza Ontology	Translational Medicine Ontology
Information Artifact Ontology	Vaccine Ontology
Interaction Network Ontology	

Agent Role (1)

Represent agents and the roles they play

ICPS Network

Classification (1)

Represent the relations between concepts and entities to which concepts can be assigned

ICPS Network	
Componency (1)	
Represent (non-transitively) that objects either are proper parts of other objects, or have proper parts	
Computational Neuroscience Ontology	
Object Role (2)	
Represents objects and the roles they play	
ICPS Network	International Classification for Nursing Practice
Part Of (2)	
Represent entities and their parts.	
Ontology for Genetic Interval	SysMO-JERM
Place (2)	
Talk about places of things	
Galen	International Classification for Nursing Practice
Sequence (4)	
Model a sequence of events occuring one after another	
FGED View	Ontology for Biomedical Investigations
IEDB View	Vaccine Ontology

We also recorded the more complex logical constructs of the patterns in MBOP (Table 3). A pattern that utilizes one of these constructs cannot belong to the OWL 2 EL Profile, a less expressive, more computationally efficient fragment of OWL 2.

4 Discussion

Our results show a very modest use of patterns in biomedical ontologies in BioPortal. Of the 68 patterns that we studied (which we filtered from the initial list of 167), only 14 appeared among the almost 300 ontologies. Ontology developers may utilize patterns because of documentation, popularity, and support in development tools. For example, all ontologies using `Upper Level Ontology`, the most common pattern, instantiated the BFO, likely due to the OBO Foundry's popularity in the biomedical community and its emphasis on using an upper level ontology. The Protégé ontology development environment provides a quick method to instantiate `Closure`, `Value Partition` (DisjointUnionOf), and `Composite Property Chaining`. We believe that the easy accessibility of these patterns from this tool accounts for their use. While `Composite Property Chaining` is not explicit in Protégé, after creating property chains, composing them is trivial. `Normalization` is both a simple idea to follow (perhaps difficult in practice), and well explained by Rector et al. [20]. Finally,

Table 3. OWL 2 logical constructs in MBOP patterns.

	All Values From	Functional Property	Cardinality Restriction	Disjunction (Union)
Entity Feature Value		✔	✔	✔
Selector				✔
Normalization				
Upper Level Ontology Closure	✔			
Entity Quality	✔		✔	✔
Value Partition				✔
Entity Property Quality		✔		✔
Defined Class Description				
Interaction Role Interaction	✔		✔	✔
Sequence		✔		
Composite Property Chaining				
List		✔		
Adapted Structure, Entity, Part				✔
Nary Datatype				
Exception			✔	✔
Nary Relationship				

with only a few additions to the Relations Ontology (RO), an ontology that can be used along with BFO, the `Sequence` pattern from MBOP can be used.

We found only a subset of patterns in a subset of the BioPortal ontologies. Despite the suggested positive effects of ODPs, the relatively modest use of ODPs in BioPortal may be due to ontology age, domain specificity, minimal tooling support, lack of pattern portability and generality, and a tension between the high logical expressivity of certain patterns and a concurrent desire for minimally expressive biomedical ontologies to enable computationally tractable reasoning (because of their large size). With regard to ontology age, it is clear that older ontologies that began development before the introduction of ODPs may not have them, as restructuring an ontology to incorporate an ODP during maintenance may be impractical. Also, one might suggest that many patterns are not found simply because they are not related to the biomedical domain.

Of more relevance is the apparent lack of end-user tooling for instantiating ODPs. The XD Tools provide with such functionality as a plugin [6]. OPPL is also available as a Protégé plugin. However, a domain expert may not be familiar with ODPs, may not necessarily know how to use ODPs or OPPL, and may not even seek out the plugins. Concurrently, Protégé and other widely used ontology development tools do not have a general method to instantiate ODPs. However, for those patterns that are already available in a development environment (`Closure`, `Value Partition`), we do see their occurrence. In this study, we cannot easily link pattern usage and any particular ontology development environment. However, most biomedical ontology developers do use Protégé. Thus, we suggest that for ODP use to increase, end-user oriented ontology

development environments must include a way to instantiate various patterns, including those available in the ODP public libraries. Furthermore, such tools could note the use of patterns in the source file of the ontology.

We have noted the opposing tension in the expressivity in the patterns and biomedical ontologies. The OWL 2 EL profile in many ways was designed with large scale ontologies in mind. SNOMED CT [22, 4] served as its driving example. Table 3, which lists the constructs that each pattern uses, shows that only 25% of the MBOP patterns follow the OWL 2 EL profile. There is a dilemma: Patterns are designed to assist in reducing errors in large ontologies, but by using them, reasoning (and finding errors) becomes computationally intractable. This dilemma may explain the lack of ODP use by some ontologies, as they maintain something similar to EL expressivity.

Additionally, we found that BioPortal ontologies used more patterns from MBOP than from ODP-Wiki. MBOP has patterns oriented toward biomedical applications, explaining a portion of this bias. However, we found very few patterns from ODP-Wiki. Many patterns in ODP-Wiki either were domain specific, or, by definition, their instantiation required inclusion of a small ontology unit. While many ontologies might follow the intension of the content ODPs on ODP-Wiki, they did not import the required ontology to truly instantiate the pattern. Thus, more generalizable patterns, and a method to capture the intension of a content ODP might also increase ODP use.

4.1 Future Work

We consider the formalization of software-design patterns in a bottom–up fashion. Gamma and colleagues extracted recurring patterns from existing software, suggesting these patterns constituted best practice in solving various software development problems [10]. Contrary to this method, the development of ODPs in the Semantic Web community appears to be top–down, especially in the case of content ODPs. Instead, we propose to find ODPs in a bottom-up fashion, as with software design patterns, by finding recurring patterns in large corpora of ontologies, such as BioPortal.

5 Conclusions

Ontology Design Patterns provide a means to enhance ontology development by capturing best practice and reducing errors. As such, ODPs may be especially applicable to large-scale biomedical ontology development. As a starting point for a larger project, we first find the prevalence of ODPs in biomedical ontologies. To do so, using the available software for manipulating ODPs, we surveyed their use in BioPortal,a large repository of biomedical ontologies. We found only a small subset of patterns in use in a portion of the corpus, with `Upper Level Ontology` being the most common pattern. To increase future ODP use in biomedical ontologies, we highlight a need for end-user ontology development tools that include a way to instantiate ODPs and a need for consideration of the logical expressiveness of ODPs . Finally, we suggest a bottom-up approach to develop generalized ODPs for re-use.

Acknowledgments

This work was supported by the NIH Grants GM086587, HG004028, and LM007033. We thank Luigi Iannone for assistance with OPPL.

References

1. Aranguren, M.E., Antezana, E., Kuiper, M., Stevens, R.: Ontology design patterns for bio-ontologies: a case study on the cell cycle ontology. BMC bioinformatics 9(Suppl 5), S1–S1 (2008)
2. Aranguren, M.E., Antezana, E., Stevens, R.: Transforming the axiomisation of ontologies: The ontology pre-processor language. In: Proceedings of OWLED (2008)
3. Aranguren, M.E., Rector, A., Stevens, R., Antezana, E.: Applying ontology design patterns in bio-ontologies. Knowledge Engineering: Practice and Patterns pp. 7–16 (2008)
4. Baader, F., Brand, S., Lutz, C.: Pushing the EL envelope. In: Proc. of IJCAI 2005. pp. 364–369. Morgan-Kaufmann Publishers (2005)
5. Blomqvist, E.: Semi-automatic Ontology Construction based on Patterns. Ph.D. thesis, Linkoping University Institute of Technology (2009)
6. Blomqvist, E., Presutti, V., Daga, E., Gangemi, A.: Experimenting with extreme design. In: Proceedings of EKAW'10. pp. 120–134. Springer-Verlag, Berlin, Heidelberg (2010)
7. Bodenreider, O., Stevens, R.: Bio-ontologies: current trends and future directions. Briefings in bioinformatics 7(3), 256–274 (2006)
8. Ceusters, W., Smith, B., Goldberg, L.: A terminological and ontological analysis of the nci thesaurus. Methods of information in medicine 44(4), 498–507 (2005)
9. Daga, E., Presutti, V., Gangemi, A., Salvati, A.: http://ontologydesignpatterns. org [odp]
10. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design patterns: elements of reusable object-oriented software. Pearson Education (1995)
11. Gangemi, A., Presutti, V.: Ontology design patterns. Handbook on Ontologies pp. 221–243 (2009)
12. Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., Schneider, L., Gómez-Pérez, A., Benjamins, V.: Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web, vol. 2473, pp. 223–233. Springer Berlin / Heidelberg (2002)
13. Grenon, P., Smith, B., Goldberg, L.: Biodynamic ontology: Applying BFO in the biomedical domain. Stud. Health Technol. Inform 102, 20–38 (2004)
14. Horridge, M., Bechhofer, S.: The OWL API: a Java API for working with OWL 2 ontologies. In: Proceedings of OWLED. vol. 529 (2009)
15. Iannone, L., Aranguren, M.E., Rector, A., Stevens, R.: Augmenting the expressivity of the ontology pre-processor language. In: Proceedings of OWLED (2008)
16. Iannone, L., Rector, A., Stevens, R.: Embedding knowledge patterns into OWL. The Semantic Web: Research and Applications pp. 218–232 (2009)
17. Niles, I., Pease, A.: Towards a standard upper ontology. In: Proceedings of the international conference on Formal Ontology in Information Systems - Volume 2001. pp. 2–9. FOIS '01, ACM, New York, NY, USA (2001)
18. Noy, N.F., Shah, N.H., Whetzel, P.L., Dai, B., Dorf, M., Griffith, N., Jonquet, C., Rubin, D.L., Storey, M.A., Chute, C.G., Musen, M.A.: Bioportal: ontologies and integrated data resources at the click of a mouse. Nucleic Acids Research 37, W170–W173 (05 2009)
19. Rector, A.L., Brandt, S., Schneider, T.: Getting the foot out of the pelvis: modeling problems affecting use of snomed ct hierarchies in practical applications. Journal of the American Medical Informatics Association 18(4), 432–440 (04 2011)
20. Rector, A.L.: Modularisation of domain ontologies implemented in description logics and related formalisms including owl. In: Proceedings of the 2nd international conference on Knowledge capture. pp. 121–128. K-CAP '03, ACM, New York, NY, USA (2003)
21. Rubin, D.L., Shah, N.H., Noy, N.F.: Biomedical ontologies: a functional perspective. Briefings in Bioinformatics 9(1), 75–90 (2008)
22. Spackman, K.: An examination of OWL and the requirements of a large health care terminology. In: Proceedings of OWLED (2007)