Three-Layer OWL Ontology Design

Michel Dumontier[1,2], Natalia Villanueva-Rosales[1]

[1] School of Computer Science, [2]Department of Biology,
Carleton University, 1125 Colonel By Drive, Ottawa, Ontario, K1S5B6 Canada
michel_dumontier@carleton.ca, nvillanu@scs.carleton.ca

Abstract. Ontology reuse is often predicated on agreement to the semantics wholly defined within an ontology document. But when faced with overly constrained semantics that one might partially reject, might one instead prefer reduced ontological commitment? In this paper, we describe how three layered ontology design promotes maximal reuse of domain ontologies by separating taxonomically organized domain terminology from i) disjointess, ii) complex expressions that define a world view, and iii) application specific requirements that impose a specific data model for data exchange and document validation.

Keywords: Ontologies, OWL, ontology design, normalization, reuse.

1 Introduction

A major goal of the Semantic Web is to add explicit semantics to web content [1], which can be realized by using ontologies to describe and relate objects with formal, logic-based representations. OWL, the Web Ontology Language, is recommended by the W3C for building Semantic Web ontologies [2]. OWL-DL is a highly expressive variant of OWL and is based on description logics (DL), a subset of First Order Logic that allows description of complex concepts from simpler ones with an emphasis on decidability of reasoning tasks. While the theoretic foundations of DL are well developed [3], there exists an enormous gap for best practices in designing OWL ontologies that enable maximal reuse for unexpected application scenarios.

In line with RDF/XML, OWL entities are named by a Universal Resource Identifiers (URI). A key and controversial issue is that despite any intent of the URI minter, meaning of a semantic web entity is locally defined by the document it is found in, opening the possibility that there can be multiple meanings for any URI [4]. Importantly, semantic web users who want to instantiate an OWL class but disagree with the logical restrictions placed over it in an ontology, could devise or choose the semantics they agree with. Users might prefer this option rather than minting their own URI because these then requires mapping to other URIs, currently a semi-automated and time consuming process. In order to provide ontologies that can be maximally reused and also reduce the possibility of *rogue* documents with potentially inconsistent meanings, we propose a three layer design for OWL ontologies with incremental ontological commitment. This proposal thus has consequences for the normalization, modularization, mapping and evolution of OWL ontologies.

2 Three-layer ontology design pattern

We propose a three-layer ontology design of classes/properties and of instances (Fig. 1). The first layer (primitive) consists of classes/properties forming taxonomic trees in which a single parent may be asserted. The second layer (complex) captures domain knowledge in the form of necessary or necessary and sufficient conditions. The top layer (application) is composed of additional restrictions that effectively describe a particular data model. Similarly, instance assertions are also divided into three increasingly restricted layers. Instances object assertions (primitive) are separated from more sophisticated expressions (complex) using specific OWL language constructs. Finally, axioms that effectively impose more specific assumptions (unique name assumption, closed world assumption) are maintained in a separate layer for data model validation. Each layer is now described in greater detail.

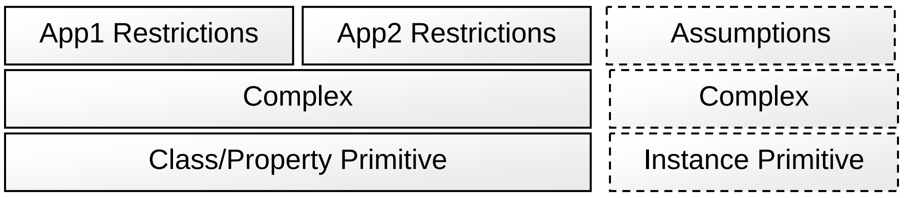

Fig. 1. Three-layer ontology design for classes/properties (solid box) and instances (dashed box).

2.1 Class/Property Taxonomic Layer

The first layer contains the declaration of domain terminology and relations in the form of OWL entities (*owl:Class; owl:ObjectProperty; owl:DatatypeProperty*) that are annotated with a human readable label (*rdfs:label*) and associated with a clear, concise and precise definition (*rdfs:comment*). We adhere to Alan Rector's recommendations on normalization [5], in that each entity is asserted to have exactly one parent, thus forming primitive trees. Since the design of primitive trees consisting of "self standing concepts" is acknowledged to be ambiguous and difficult [5], we relax the criteria of forming disjoint trees, thereby avoiding potential inconsistencies arising from disjointness. This omission is particularly relevant when reasoning with additional layers having defined necessary and sufficient conditions such that instances are realized into multiple classes. For instance, although a scatterplot and a line graph are different types of statistical graphs[1], the assertion of disjointness of these classes in a graph ontology would preclude the discovery of a line graph from a scatterplot that has lines drawn between the data points. Thus, these primitive layers enable users to benefit from taxonomically organized declarations of domain terminology without having to further commit to semantics that might lead to inconsistencies with additional knowledge. Moreover, it allows users to exchange data identified by class membership and related with the provided basic relations.

[1] http://ontology.dumontierlab.com/statistical-graph-primitive

2.2 Complex layer

The complex layer refines the primitive layer by imposing restrictions such as necessary or necessary and sufficient conditions beyond the asserted subsumption. These include more expressive aspects of the OWL-DL language (OWL 1.1) including disjunction, existential and universal restriction, union, intersection, complement, and (qualified) cardinality restrictions, transitivity of properties, reflexivity of properties, etc. The complex layer aims to augment the ontology with domain knowledge that would lead to meaningful inferences, and includes mappings to upper level ontologies. Drawing from the statistical graph domain, a Time Series Graph would necessarily require time intervals as categorical data. Fully defining necessary and sufficient conditions (equivalentClass) enables realization of class membership by DL reasoners. For instance, the complex layer of our statistical graph ontology[2] fully describes a Graph Title, thus titles of a graph are automatically discovered.

2.3 Application restrictions

The application restriction layer applies highly restrictive constraints and may be used for the purposes of document validation and application interoperability. This layer refines primitive classes and their relations, and should extend the complex layer, if defined. Requirements are described using necessary conditions and if necessary applying (carefully) extra closure axioms with universal restrictions (*only*) and cardinality restrictions (*min, max, exactly*). For instance, the iGraph ontology[3] specifies the necessary conditions that defines a valid data model to be used with the iGraph application [6]. Alternatively, one could define new application specific classes (i.e. iGraphLineGraph) and determine whether instances are realized as class members from necessary and sufficient conditions. In this way, the specification for necessary information is completely encoded in the ontology and can be validated using a DL reasoner.

2.4 Instance layers

Assertions about instances are also placed in separate layers for data exchange/integration, reasoning and document validation. The instance primitive layer includes assertions about class membership, object relations and datatype values. This layer can be seen as a solution for developing a flexible model that can be used to annotate publicly available data (e.g. from databases) without interest or need of commitment to more complex logical restrictions. For example, our instantiated statistical graph ontology[4] asserts that *graph* is an instance of the class *Graph* and this is related to the *plot* instance of the class *Plot* class using the *hasPart* property defined in our basic relation ontology.

[2] http://ontology.dumontierlab.com/statistical-graph-complex
[3] http://ontology.dumontierlab.com/igraph
[4] http://ontology.dumontierlab.com/igraph-example

A second layer (complex) adds axioms such as existential restrictions to denote necessary relations between instances (possibly unknown) or (qualified) cardinality restrictions to identify how many of such relations must to exist. Thus, expressive OWL constructs such as *owl:sameAs*, *owl:differentFrom* may be specified in this layer.

Finally, the assumptions layer contains axioms that are needed to enforce a specific data model that could modify the reasoning inferences. For instance, if we want to assume that the information is complete and we would like to obtain inferences bases on this assumption, we might want to "close" the knowledge base by asserting closure axioms (*i.e. complex class membership, ABox closure, etc*).

3 Discussion

The increasing expressivity in the three-layered ontology design is analogous to the formulation of more complex descriptions from simpler ones in description logics. Domain knowledge is first captured as a taxonomic tree of non-disjoint classes which is asserted in an OWL ontology along with basic relations required to populate the ontology. Such normalization is an important aspect of ontology design because it provides front line support for ontology reuse, maintainability and evolution. Additional knowledge leading to meaningful inferences (e.g. membership inferences via realization) is declared in a separate layer, thus enabling multiple "world views", i.e. ontologies with the same primitive layer, but different complex layers, could potentially drive to different sets of inferences. Finally, the specification of application requirements aims to ensure application compatibility and support for interactions with other data models, e.g. relational databases. In providing multiple levels of ontological commitment, users may choose to commit to i) a non-disjoint primitive tree, ii) a logically enriched tree containing additional, explicit domain knowledge, or iii) a constrained data model. Thus, our approach promotes ontology reuse, at various levels of ontological commitments, which should be acceptable to a broad number of requirements. Indeed, since OWL currently only supports document level imports (*owl:imports*), a user now has three documents to choose from, rather than rejecting the ontology altogether due to restrictions placed on domain entities that they may not agree with. Moreover, he may choose to build a new complex layer.

Ontologies containing taxonomically organized terminology are already used to manage biomedical terminology (i.e. the Gene Ontology). As more sophisticated semantic web ontologies (i.e. BioPAX) emerge, we advocate that more restricting axioms, for specific domain and data models can be described in a separate layer. The BioPax OWL-DL ontology aims to exchange molecular interactions, complexes and pathways [7], and places database-like restrictions to enforce single values and closures. While proteins are routinely referred to using multiple names, instances of BioPax Protein class are restricted to having *at most* one name (using datatype property *bp:NAME*). This choice clearly reflects a desire to maintain the relational database model in which name is a single valued field in a table containing protein entries. Importantly, certain restrictions require closed world reasoning that holds in the relational model, but does not hold in OWL. It seems unreasonable to outright reject instantiating the class *bp:Protein* because of this restriction. Thus, cleanly separating application restrictions from domain terminology allow users to instantiate the ontology with the declared classes and relations, but don't have to commit to restrictions placed to facilitate (legacy) data exchange. Should the exclusion of role-based restrictions in the primitive layer lead to inconsistent ontologies using a complex layer, then procedures to resolve the discrepancy may yet have to be developed.

Another issue identified by the BioPAX community involves the validation of OWL documents meant for data exchange, but whose underlying semantics assume an open world. We propose that additional layers over data may be used to place certain assumptions, which can then be used to validate specific data models. This model supports interoperability with relational database systems with primary key requirements and whose implementation assumes unique name assumption and the closed world assumption. The topmost layer of assumptions over the data could also be used to integrate/exchange knowledge with systems with other models such as modal logic systems. We expect that continued development of reasoning systems will make this layer less essential in the future.

Our proposal is compatible with approaches towards the automatic design of ontology modules [8, 9]. Ontology modules include minimal information such that importing the module would provide the same answers to arbitrary queries as importing would have been obtained with the entire ontology. While the taxonomic layer provides some information about the nature of the entities, the complex layer provides valuable role-based descriptions from which subsequent modularization would be most effective for semantic query answering.

This proposal has also interesting considerations for ontology mapping and ontology evolution. Ontology mapping involves the automated mapping of ontological entities such that for each concept in ontology A, a corresponding concept in ontology B is identified (if existing), with the same or similar semantics. Major issues in ontology mapping include 1) ensuring that the restrictions placed in one document are equivalent to another and 2) dealing with ontology evolution such that changes are propagated. Our approach potentially would allow agent-based negotiations to occur in three steps, beginning with the least expressive to the most expressive set of axioms about the resources. Thus, agents may determine an equitable mapping at some less expressive level, thereby avoiding inconsistencies that preclude a mapping. With respect to ontology evolution, class membership assertions are expected to be more stable than assertions that restrict classes, therefore this approach will modularize ontology evolution in different documents, separating the knowledge that evolves more frequently from the one that does not. This model was useful in the design of ontologies in the iGraph system. In particular, the flexibility to separate the application restrictions from the complex layer, allows reusing the latter when modeling other application requirements without committing to the iGraph requirements [10]. Future work includes applying this model to other ontology-centered integration systems.

4 Conclusions

In this paper we described a methodology for a three-layered ontology design that captures domain knowledge at various levels of granularity so as to maximize ontology reuse, maintainability and evolution, yet also satisfy application specific requirements for data exchange. This approach separates the declaration and taxonomic organization of domain concepts from complex class descriptions that may be used for enhanced reasoning or for placing constraints on data exchange. We believe that this three-layered design will be particularly useful for reasoning with across multiple world views and where there is a need for semantic annotations of data and integrate/exchange knowledge with systems based on other models such as relational databases and modal logic systems. Research towards flexible exchange is a key activity that will facilitate the use of the semantic web in real world applications.

Acknowledgments. We thank the anonymous reviewers for providing constructive feedback that has improved the quality of the paper. Funding for NVR and MD was with CONACYT scholarship #150581 and NSERC, respectively. We thank Leo Ferres for providing iGraph system requirements.

References

1. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web. Scientific American. Vol. (2001)
2. W3C: OWL Web Ontology Language Guide. In: Smith, M.K., Welty, C., McGuinness, D.L. (eds.) (2004)
3. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Schneider, P.: The Description Logic Handbook: Theory, Implementation and Applications. Cambridge (2003)
4. Parsia, B., F. Patel-Schneider, P.: Meaning and the semantic web. Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters. ACM Press, New York, NY, USA (2004)
5. Rector, A.L.: Modularisation of domain ontologies implemented in description logics and related formalisms including OWL. KC2003. ACM Press, Sanibel Island, FL, USA (2003)
6. Ferres, L., Verkhogliad, P., Lindgaard, G., Boucher, L., Chretien, A., Lachance, M.: Improving Accessibility to Statistical Graphs: The inspectGraph System. Ninth International ACM SIGACCESS Conference on Computers and Accessibility, Tempe, Arizona (2007)
7. Luciano, J.S.: PAX of mind for pathway researchers. Drug Discov Today. Vol. 10 (2005) 937-942
8. Bernardo Cuenca, G., Ian, H., Yevgeny, K., Ulrike, S.: Just the right amount: extracting modules from ontologies. Proceedings of the 16th international conference on World Wide Web. ACM Press, Banff, Alberta, Canada (2007)
9. Grau, B.C., Parsia, B., Sirin, E., Kalyanpur, A.: Modularizing owl ontologies. KCAP, Banff, Canada (2005)
10. Ferres, L.D., Michel, Villanueva-Rosales, N.: Semantic Query Answering with Time-Series Graphs. The 3rd International Workshop on Vocabularies, Ontologies and Rules for The Enterprise (VORTE 2007), Annapolis, USA. (2007)