XX ENMC
ENCONTRO NACIONAL DE MODELAGEM COMPUTACIONAL
VIII ECTM
ENCONTRO DE CIÊNCIA E TECNOLOGIA DE MATERIAIS
16 a 19 de Outubro de 2017
Instituto Politécnico - Universidade do Estado de Rio de Janeiro
Nova Friburgo - RJ

ENHANCEMENTS OF LINKED DATA EXPRESSIVENESS FOR ONTOLOGIES

Renato Fabbri[1] - renato.fabbri@gmail.com
[1]University of São Paulo, Institute of Mathematical and Computer Sciences - São Carlos, SP, Brazil

Abstract. *The semantic web has received many contributions of researchers as ontologies which, in this context, i.e. within RDF linked data, are formalized conceptualizations that might use different protocols, such as RDFS, OWL DL and OWL FULL. In this article, we describe new expressive techniques which were found necessary after elaborating dozens of OWL ontologies for the scientific academy, the State and the civil society. They consist in: 1) stating possible uses a property might have without incurring into axioms or restrictions; 2) assigning a level of priority for an element (class, property, triple); 3) correct depiction in diagrams of relations between classes, between individuals which are imperative, and between individuals which are optional; 4) a convenient association between OWL classes and SKOS concepts. We propose specific rules to accomplish these enhancements and exemplify both its use and the difficulties that arise because these techniques are currently not established as standards to the ontology designer.*

Keywords: *Semantic web, Linked data, Ontology, RDFS, OWL*

1. INTRODUCTION

The semantic web is constituted by data which is linked in the same way web pages are: through HTTP and URLs. W3C recommendations are the main sources of protocols and best practices. In practice, the terms 'linked data' and 'semantic web' are most often used interchangeably. A distinction might arise in some contexts where one needs to refer to the linked data or the semantic web created by all or some portion of linked data, but, as a knowledge and technological field, the terms are equivalent.

The semantic web is built using the Resource Description Framework (RDF). The RDF data model is based on making statements in the form of triples ("subject-predicate-object") and using Unique Resource Identifiers (URIs) for objects and concepts. It is also part of the framework to use URIs that are URLs whenever possible, to enable the data linkage. Accordingly, one can write:

```
<http://example.org/people/mary>
        <http://example.org/properties/name> "Mary Shastacian" .
<http://example.org/people/mary>
        <http://example.org/properties/age> "57" .
```

```
<http://example.org/people/mary>
        <http://example.org/properties/likes>
                <http://example.org/concepts/reading> .
```

to express that there is person called Mary Shastacian which is 57 years old and likes reading. There are many formats to write/serialize RDF data. The example above is written in Turtle and we will use this format throughout this document.

One of the most important notions of the semantic web is that of an ontology. An ontology, in this context, is a formalized conceptualization, comprised by concepts and relations between the concepts and between the relations themselves. In current semantic web, most simple ontologies are written using the RDFS protocol, by which one can specify, among other things:

- concepts;
- properties, which are concepts that are used as predicates and thus relate concepts;
- special relations between concepts that state that one concept is more general than the other[1];
- the subjects and objects that can occurs in a triple where a specific property is a predicate.

One can also write an ontology by using OWL, in which all the expressive capabilities of RDFS are available, but one can also, among other things:

- state "property axioms", i.e. specify if a property is e.g. reflexive or transitive;
- state "class restrictions", i.e. specify if a class instance e.g. necessarily holds a relation to another class instance or data.

OWL has a richer vocabulary than RDFS and is (way more) complex. This complexity is a drawback together with the greater computational cost for performing inference. The advantage is the greater power to represent conceptualizations.

The semantic web practices evolves by constant practice and recommendations written by many specialists usually after years of research and discussion. Therefore, this document is written in a very humble manner, without the pretense of yielding new recommendations abruptly, but as a way to report issues and candidate workarounds/solutions that emerged by years of intense practice in designing ontologies (Fabbri, 2014, 2017; Fabbri & Oliveira, 2016; Fabbri et al., 2015).

Section 2 is dedicated to exposing some of the problems in current RDFS and OWL ontologies, with emphasis on the ones to which this document holds proposals to resolve or lessen. In Section 3 such proposals are described and discussed. Final remarks are articulated in Section 4.

2. THE PROBLEMS

The semantic web technology standards are very appreciated by the researchers, developers and users (Hitzler & Janowicz, 2013). Even so, there are many issues regarding current state

[1]This kind of relation is called hypernymy. Examples: mammal is a hypernym of monkey, drink is a hypernym of beer.

of the art. For example, if an individual of a class is found also to be a class, it implies OWL Full which is a setting where inferences are presently regarded as not computationally feasible. This document is dedicated to proposals that resolve (or alleviate) the problems described in the following subsections.

2.1 No (standard) way to express potential relations

There is no recommendation about how to bind to a property a *possible* subject or object. Using `rdfs:domain` or `rdfs:range` has the consequence that **any** subject or object (respectively) will be inferred to be a member of the class specified by range or domain. There are, however, many situations where this is not intended. For example, one might find or create a property `example:name`. Suppose it is intended to be used to express names of animals, say because the person created the property while designing an ontology about animals. If that is formalized in the property by `rdfs:domain`, it implies that whenever the `example:name` is used, the subject will be inferred (e.g. by an automated reasoner) to be an animal. Now suppose someone is describing a tool in RDF and wants to write its name. If he/she uses the `example:name` property, it will be erroneously expressing that the tool is an animal. Such a person is required to create another property `example2:name` to write the name(s)[2]. What is worse: if the person wants to create an ontology about tools, and wants to express that tools are likely to have names, the way to represent this if by assigning to the property `exampleX:name` the "domain tool" and fall into same problem as above, but now with tools instead of animals. If **all** the tools have, without exception, a name, then one can express it as an OWL existential class restriction. But, again, that is often not the case. Example: algorithms are likely to have names, but not all of them have names. One workaround to this problem is to describe these potential relations as notes to the classes and properties, but it does not favor the machine readable design of the semantic web.

2.2 All classes and properties are equally fundamental

The relevance of this issue can be grasped by tackling the task of expressing a conceptualization for humans. Say someone wrote an ontology about machine learning and that it is very complete in the sense that it contains many concepts, all those that the author could find. It is straightforward to acknowledge that it will be very difficult for a newcomer to make sense of the ontology. Also, in many cases it is difficult for anyone to make sense of an ontology when it comprises many concepts and relations. Thus, it is very useful to have a standard way to express that some set of concepts or triples are more fundamental then others in a layered fashion, but there isn't a standard way to achieve this.

2.3 Distinction in diagrams of relations between classes and individuals and between imperative and optional relations

This is not a problem found in formalizing the ontology, but a problem that arises when depicting the ontology as a diagram. For example, the (class of) dinosaurs were succeeded

[2]The `rdfs:label` is often used to assign names, but the example is simple and valid. It is straightforward to 1) conceive some property to which a potential domain might be intended but the restriction implied by `rdfs:domain` is not convenient; or 2) find an example in a widely adopted ontology, e.g. `foaf:topic` has `rdfs:domain foaf:Document`.

by the (class of) mammals, this is a relation between classes. A specific animal might have siblings, this is a relation between individuals. Also, a relation might be imperative for a class (i.e. it might be expressed as a class restriction) or optional. These are all poorly distinguished in diagrams that are meant to represent ontologies.

2.4 No obvious way to associate an OWL ontology and a SKOS vocabulary

In a SKOS vocabulary, a concept is an individual of the class `skos:Concept`. In an OWL ontology, a concept is a class. As stated in the beginning of this section, if an individual of a class is also a class, one has an OWL Full context, which is currently considered to entail inference that is not computable. This implies that one has to choose between making an OWL ontology or a SKOS vocabulary or find a way to make them compatible. This might be the most complex issue tackled in this document.

3. ENHANCEMENT PROPOSALS

As the problems stated in the previous section are distinct, the proposed enhancements are described case-by-case.

3.1 Potential domain, range, predicate and post predicate

To address the problem described in Section 2.1, I propose creating a new set of properties:

- potential/nonrestrictive domain and range of a property might be expressed using `example:potentialDomain` and `example:potentialRange`.
- potential predicates when an instance of a class is a subject or an object might be expressed using `example:potentialPredicate` and `example:potentialPostPredicate`.

The `example:` in these triples should be replaced by a true namespace prefix, such as `http://purl.org/socialparticipation/rdfe/`. This was not done until the moment because we chose to wait, present this article and distribute it online to find out if the problem is sound, if there is already some solution currently in use or if there is some trustworthy maintainer that is willing to keep these properties online in a more suitable namespace. As a last resort, we (me and partners) might create a reasonable namespace and bind these properties.

3.2 Ascribing numeric values that reflect how fundamental is something

To address the problem described in Section 2.2, every property and class might be a subject to a triple:

```
example:something example:fundamental 4 .
```

with "4" replaced by any other numerical value. At first, it seems reasonable to assume that these values are valid within the ontology and relative to each other. Accordingly, one might ascribe `fundamental 100` some to some nucleus of meaning and lower values to other layers of concepts and statements with a (more-or-less) equal weight with respect to being fundamental to the ontology. As statements are usually not given a URI explicitly in RDF files

(one has to use "RDF reification" to give a triple a URI), to bind a triple or a set of triples to a level, one will probably be better of by using named graphs. Reification is a solution, but a rather verbose and unusual one.

3.3 Different arrow heads and tails

To address the problem described in Section 2.3, we notice that: 1) the edges are often small, so using line styles (such as dashed, dotted or full) is likely to yield bad results often. The use of colored edges is an option, but given that many journals require images to be black and white, it is not suitable as the standard at least at the moment. Nevertheless notice that different arrow heads are in fact used (almost as a standard) as visual cues to differentiate relations that are yield by properties from those that associate a class to its superclass, This is an indicative that using different arrow heads might be a good strategy to reaching a good solution. Also, the start of the edge might be a good visual cue to discern different types of relations. We should recall here that in OWL a property might be specified to satisfy a number of axioms, which are also prone to be represented with these same artifices.

A reasonable choice for most cases, which I've been practicing, and found in literature but not consistently, is:

- Empty arrow heads depict hypernimny: `Class->superClass`.
- Filled arrow heads depict property relations. Here the problem related to Sections 2.1 and 3.1 become very clear: potential property relations are often depicted in these diagrams which are not defined by range or domain, property axioms or class restrictions. Priority should lie on the shape of arrow head and tail for depicting different characteristics of the properties. After that, shape of the edge and then color.
- Nodes have distinct colors datatypes and classes, e.g. orange and green.
- A legend for relating arrows and nodes to RDF namespace, e.g. `owl:(Class,ExistentialRestr` `xls:dataytype`.

One more elaborate template I've been used, and recommend at least for reference, is:

- All arrows with a standard shape are potential relations in the sense exposed in Section 3.1.
- Universal restrictions have inverted arrowheads.
- If there are existential restrictions and potential relations, differ the arrow heads (shape and size).
- Non-functional properties have dashed lines.
- Colors are consistent with nodes: Data properties and data nodes have (possibly different) shades of orange. Objects and object properties have (possibly different) shades of green.
- Redundancy: Universal and existential restrictions might have different greens and arrowheads.

3.4 OWL class as subclass of SKOS concept

There are different strategies for making an OWL ontology and a SKOS vocabulary compatible (Bechhofer & Miles, 2008). The solutions that are fit for general purposes, among the ones that were found or conceived by the author, are:

- describe the ontology as OWL then translate it to SKOS (and probably automate the process by means of a script).
- State that each OWL class is also a subclass of the `skos:Concept`. This yields a less redundant result, but it might not be in accordance to SKOS and OWL specifications or entail OWL Full. For example, the use of some SKOS properties might imply that the subject is an individual of the `skos:Concept` class. This solution needs to be further studied and discussed with specialized groups in order to be used with confidence.

The problem with the other solutions:

- yields OWL Full (if a SKOS concept, an instance, is also an OWL class); or
- yields annotations on instances of OWL classes, not on the classes (if we annotate instances of OWL classes with SKOS properties); or
- yields a less formal ontology (if we derive the ontology from the vocabulary); or
- extrapolates the purpose of SKOS (giving enough expressive power to SKOS to achieve an OWL without degeneration of the conceptualization).

4. CONCLUSIONS AND FURTHER WORK

This article is a formalization of the issues to allow the author and collaborators to confirm (or refute) that these problems and the solutions herein proposed are sound and have not yet received solutions by the semantic web community. These issues and solutions are the result of years of dedication to the subject, including learning by (online and in-person) classes and designing dozens of OWL ontologies for academic and State purposes (Fabbri, 2014, 2017; Fabbri & Oliveira, 2016).

Next steps should include:

- the pursue of feedback through circulation of this document (e.g. in online forums) to enhance the solutions herein proposed and better understand their possible uses (e.g. in reasoners).
- The use of these solutions in ontologies, both new ones and others that are already available.
- If these issues prove themselves relevant over time, we will consider contributing to the full realization of solutions. For example, we might help achieving and maintaining suitable URIs for the properties described in Section 3.2 or help in the elaboration of convenient arrow heads for the problem described in Section 2.3.
- The confirmation or refutation of the second possible solution to make OWL and SKOS compatible as described in Section 3.4.

Acknowledgements

The author thanks CNPq for the funding received while researching the topic of this article, the researchers of IFSC/USP and ICMC/USP for the recurrent collaboration in every situation where we needed directions for investigation.

REFERENCES

Bechhofer, S., & Miles, A. (2008). Using OWL and SKOS. Note of the SWD W3C working group. From https://www.w3.org/2006/07/SWD/SKOS/skos-and-owl/master.html

Fabbri, R., & de Oliveira, O. N. (2016). Linked Open Social Database. Github repositories, from https://github.com/ttm/linkedOpenSocialDatabase/raw/master/paper.pdf

Fabbri, R. (2017). Topological stability and textual differentiation in human interaction networks: statistical analysis, visualization and linked data. Doctoral thesis. From https://github.com/ttm/thesis/raw/master/thesis-rfabbri.pdf

Fabbri, R. (2014). Social participation ontologies and rules to fuel a social participation linked data cloud. Technical report for the United Nations Development Program. From https://github.com/ttm/pnud5/raw/master/latex/produto.pdf

Fabbri, R., de Luna, R. B., Martins, R. A. P., Amanqui, F. K. M., & Moreira, D. D. A. (2015). Social Participation Ontology: community documentation, enhancements and use examples. arXiv preprint arXiv:1501.02662.

Hitzler, P., & Janowicz, K. (2013). Linked Data, Big Data, and the 4th Paradigm. Semantic Web, 4(3), 233-235.