Context Slices: Representing Contexts in OWL
http://ontologydesignpatterns.org/wiki/Submissions:Context_Slices

Chris Welty
IBM Watson Research
Hawthorne, NY 12540, USA
cawelty@gmail.com

ABSTRACT
This ontology pattern can be used to represent and reason about contextualized statements using standard OWL dialects. The simple idea is to bundle the notion of context into certain nodes in the graph, rather than the more typical treatment of contexts as a property of the statements themselves.

Keywords
Semantic Web, OWL, RDF, Contexts.

1. INTRODUCTION
Most information on the web is contextualized somehow, for example information may be believed by a person or organization, it may hold only for some time period, it may have been reported/observed by an individual, etc. There are myriad proposals and logics for context, but none are standards and few have even prototype implementations.

In RDF and other binary relation languages (like object oriented languages and description logics), one typical way to represent that a binary relation holds in some context is to "reify" the relation-holding in the context as an object with a binary relation between the obtainment and each the two relation arguments and a third binary relation between the obtainment and an object representing the context itself. The downside to this approach is the expressive ability of the language to describe the binary relation, especially in the case of description logics, is lost. One can of course use RDF reification, however this is not supported in OWL, either.

The motivation for context slices is to provide a logical pattern for encoding context information in standard RDF graphs that allows some of the expressiveness of OWL to be used in describing the relations that hold in contexts.

This is a generalization of the four dimensional ontology for fluents published in [1].

2. PATTERN DESCRIPTION
The idea of the context slices pattern is, rather than reifying the statement itself, to create a projection of the "relation arguments" in each context for which some binary relation holds between them.

Take for example the statement "Chris believes Sam is CEO of IBM". Say we already have nodes in some graph representing Sam, Chris and IBM. We create, as shown in Figure 1, the context c_1 corresponding to Chris' belief, and two nodes representing Chris' belief about Sam and Chris' belief about IBM (shown as Sam@c_1 and IBM@c_1).

This allows us to represent ceoOf as a binary relation, which seems more natural, and it allows us to use the expressivity of OWL in more ways. We can say of the ceoOf relation that it has an inverse, hasCeo. We can express cardinality, e.g., a company may have only one CEO within a context. We can say that a relation is transitive or symmetric. We can express relation taxonomies in the usual way.

While clearly OWL does not support RDF reification, and so none of this is possible if statement reification is used. As mentioned above a more standard way of representing this kind of information (including time, belief, knowledge, etc.) is to create an OWL class that represents the relation holding, with properties for the arguments. This approach makes it possible to express global but not local range and domain constraints, global but not local cardinality, and symmetry.

Note that the ContextualProjection class should be considered disjoint with any of the classes in an ontology that have projections.

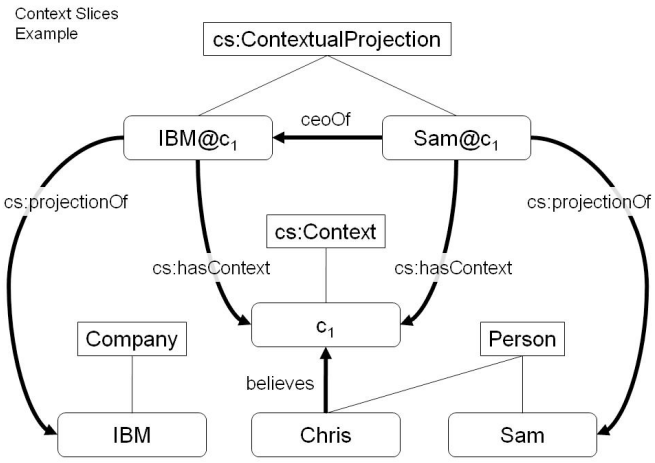

Figure 1: Graphical illustration of an example using the pattern.

3. IMPLEMENTATION
In OWL functional syntax:

```
Ontology(<http://example.org/ContextSlices>
    Annotation(owl:versionInfo "1.0"@en)
    Annotation(rdfs:label "Context slices ontology logical
        pattern"@en)
```

```
Declaration(Class(cs:Context))
DisjointClasses(cs:Context cs:ContextualProjection)
Declaration(Class(cs:ContextualProjection))
SubClassOf(cs:ContextualProjection
    ObjectAllValuesFrom(cs:hasContext cs:Context))
SubClassOf(cs:ContextualProjection
    ObjectExactCardinality(1 cs:hasContext))
SubClassOf(cs:ContextualProjection
    ObjectExactCardinality(1 cs:projectionOf))
DisjointClasses(cs:ContextualProjection cs:Context)
Declaration(ObjectProperty(cs:contextualProperty))
ObjectPropertyDomain(cs:contextualProperty
    cs:ContextualProjection)
ObjectPropertyRange(cs:contextualProperty
    cs:ContextualProjection)
Declaration(ObjectProperty(cs:hasContext))
FunctionalObjectProperty(cs:hasContext)
ObjectPropertyDomain(cs:hasContext
    cs:ContextualProjection)
Declaration(ObjectProperty(cs:projectionOf))
FunctionalObjectProperty(cs:projectionOf)
ObjectPropertyDomain(cs:projectionOf
    cs:ContextualProjection))
```

4. REFERENCES

[1] Welty, Chris and Richard E. Fikes. 2006. A Reusable Ontology for Fluents in OWL. In Bennet and Fellbaum, eds., *Proceedings of the Fourth International Conference on Formal Ontology in Information Systems*. IOS Press. See http://www.booksonline.iospress.nl/Content/View.aspx?piid=2209.