Propositions of Conventions for RDF

RDF is a general model and representing a piece of information in RDF/XML can be done in many lexical/syntactic/structural/ontological ways. Unfortunately, these different representations often cannot be automatically compared with each other and therefore retrieved, merged or reused. We cannot expect the metadata providers to follow the same schemas and this would not be prevent incomparable syntactic/structural variations. Metadata providers (including schema creators) need to follow conventions. Here are some propositions.

Table of Contents (Postcript version)

1. Some Lexical Conventions

1.1. InterCap Style for Names

Resources in RDF/XML must have legal XML names and are case sensitive. For its own identifiers, RDF [RDFMS] has adopted the convention that all property names use "InterCap style"; that is, the first letter of the property name and the remainder of the word is lowercase; e.g. `subject`. When the property name is a composition of words or fragments of words, the words are concatenated with the first letter of each word (other than the first word) capitalized and no additional punctutation; e.g. `subClassOf` and `rhetoricalRelation`. Class names follow the same convention except that their first letter is capitalized, e.g. `TaxiDriver` [RDFSchema].

1.2. Singular Nouns for Names (whenever possible)

Representing knowledge using only classes and properties named with singular nouns is most always possible. As a matter of fact, English sentences can also generally be re-written to avoid the use of adjectives and verbs (with the exception of ''to be'' and ''to have''). For instance, ''A cat named Tom jumps toward a wooden table'' can be re-written as ''The cat that has for name Tom is agent of a jump that has for destination a table the material of which is some wood''. This sentence (which happens to be a correct sentence in Formalized English [FE]) seems unnatural but makes explicit the classes and properties of the resources, and can be directly represented using a notation for a directed graph model.

Similarly, writing statements using only nouns, compound nouns or verb nominal forms makes these statements more explicit. Furthermore, with this convention, the number of lexical and structural possibilities to express these statements is significantly reduced (i.e. the choices of classes, properties and ways they can be combined are reduced). Therefore, there is a stronger possibility that statements can be automatically matched, and thus retrieved, merged or reused.

Why Nouns?

- Sentences cannot be rewritten using only adjectives or verbs or adverbs.
- Classes denoted by (and representing the meaning of) adjectives/verbs/adverbs cannot be organized via `subClassOf` links with classes denoted by (and representing the meaning of) nouns. Since most identifiers in current ontologies are nouns (e.g. the Dublin Core [DC] or the Upper Cyc Ontology [CYC]), it is better to use nouns whenever possible for the sake of metadata retrieval and sharing. Furthermore, it is difficult to find subclass relationships between classes denoted by adjectives or adverbs.
- Only classes representing the meaning of nouns can have various kinds of quantifiers. For instance, it is possible to speak about "any abstract_entity" and "at least 3 transformations" but "any abstract" nor "at least 3 transform".
- Sometimes, classes named with adjectives are misnamed because they actually refer to the meaning of some nouns, e.g. a class named `Abstract` instead of `abstraction` or `AbstractEntity`. This leads to less readable ontologies and statements.

Even for Property Names?

Unlike instances of classes, relations (we use the term "relations" to refer to the use of properties within statements) are only existentially quantified. Furthermore, avoiding adverbs for property names is sometimes difficult, e.g. for spatial/temporal relations. Hence, should we always use a nominal form, e.g. `nearLocation` and `aboveLocation` instead of `near` and `above`? Properties can still be organized with subPropertyOf relations in both cases. Names such as `isDefinedBy` and `seeAlso` (both proposed in [RDFSchema]) are more problematic. Better names seem to be `definition` and `additionalInformation`. At least, they are in accordance with the reading conventions for RDF [RDFMS] and other graph directed models (e.g. Conceptual Graphs [CGs]):
"<source resource/concept> HAS FOR <property/relation> <destination resource/concept>" or
"<source resource/concept> IS <property/relation> <destination resource/concept>" or
"<destination resource/concept> IS THE <property/relation> OF <source resource/concept>".

Why Singular Nouns?

Most identifiers in ontologies are singular nouns. Category names must be in the singular in the Meta Content Framework Using XML [MCF/XML]. Class names as plurals are introduced to represent collections. Sometimes, users represent statements about collections although they actually want to talk about each member of a collection. As noted in the the Section 3.3. of [RDFS], distributive referents (i.e. the keyword "aboutEach") should be used to avoid those misrepresentations.

1.3. Inverse Relations

When writing a statement, the RDF/XML user cannot refer to the inverse of a relation (the direction of the relation cannot be reversed using a special property, e.g. `direction="-"`). This leads users either to declare properties as inverses of others (e.g. `AgentOf` as inverse of `Agent`) or to write several statements instead of a more structured one. The first method implies additions to the schema and overhead for RDF inference engines to match statements. Furthermore, there is not yet a standard way to declare that a relation is the inverse of another. The second method is only tedious for the human writers and readers. A solution would be the following convention: the suffix "Of" can be added or removed from the name of a relation to indicate to the RDF parser that its direction is inverted. Thus, users would not have to declare inverse properties and there would not be overhead for the RDF inference engines. However, the RDF parsers have a little overhead to check if a relation name has been declared. Another solution to this problem would be to allow the direction of a relation to be specified with a special property.

2. Some Structural Conventions

2.1. Binary Basic Relations

Why binary relations?
As with most frame-based models, RDF only supports binary relations. Relations of greater arity may be represented by using structured objects or collections, or using more primitive relations. For instance, "the point A is between the points B and C" may be represented using the relation `between` and a collection object grouping B and C, or using the relation types `left` and `right`, `above` and `under`, etc. Thus, the fact that RDF only supports binary relations is not a conceptual limitation but a structural limitation (which is good since it leads to more comparable statements) and often leads to more explicit and precise statements.

Why basic relations?
Let's consider the sentences "Tom has bought a car" and "Tom has bought a car for Mary on the 17/5/1999". A statement representing the first sentence and using the relation `buyer` cannot be automatically compared with a statement representing the second sentence and using the class `Purchase` and the relations `agent`, `object`, `recipient` and `time` (unless `buyer` has been given a definition in terms of `Purchase`, `object` and `agent`, and the RDF engine is able to exploit this to expand the first statement). Decomposition leads to more explicit and comparable statements. Furthermore, it permits a limited set of basic relations to be often reused and therefore declared in reusable ontologies. These relations (spatial/temporal/thematic/attributive/...) are basic but precise: detailed signatures (range and domain) can be associated to them (cf. our top-level ontology) and be exploited for metadata checking, merging or mining.

Metadata providers often use names of attributes/characteristics (e.g. of physical characteristics such as mass and color) as relation names. This practice would not be a problem if all attributes could be represented as properties and organized via subPropertyOf relations. Unfortunately, after exploring this option with WordNet [WN], we realized that relatively few attributes can be used as relations. Therefore, we introduced the class `AttributeOrMeasure` and classified the top-level WordNet attribute categories and measure categories under it (it is sometimes difficult to distinguish these two notions, e.g. `Color` is an attribute but `Red` and its corresponding wave length may be seen as a measure). Though we also provided a few relations such as `mass` and `length`, these relations that could be decomposed/defined using the combination of an instance of `AttributeOrMeasure` and the relation `attribute` (plus possibly the relation) should be considered as exceptions.

A few role nouns, such as `child`, `creator` and `driver` are also used as relations. However, these relations are not basic (they refer to processes) and, except for those that are very commonly used, should be avoided.

2.2. Avoid Disjunction or Negation in Statements

For tractability reasons, most logic-based languages do not permit the use of containers, disjunctions and general negations in statements, but many (e.g. the Business Rules Markup Language [BRML]) permit conjunctive existential formulas, and type definitions (if only as relations between classes, e.g. subclass relations or exclusion relations) or IF-THEN rules based on these formulas. To ease the management of metadata by a RDF engine or permit its conversion into other languages, it seems better to avoid using relations such as `or` or `not`, whenever possible.

As a simple example, instead of writing that a resource X has for type *DirectFlight* OR *IndirectFlight*, it seems better to declare X as an instance of a type *Flight* that has *DirectFlight* and *IndirectFlight* as exclusive subtypes (i.e. types that cannot have common subtypes or instances). Exclusion links between types (or between entire statements) are the kinds of negations that can be handled efficiently, and are included in many expressive but efficient logic models, e.g. Courteous logic on which the BRML is based.

2.3. Precision, contextualization and constraints

The more precise the representations the less chance of conflicts between them. The more primitive its components, and the more constraints associated to their classes, the more likely the representation can be cross-checked and compared with others to respond to queries. Representations should be contextualized in space, time and author origin. No relevant concepts should be left implicit. It is stated in [RDFMS] that for some uses, writing property values without qualifiers is appropriate, e.g. "the price of that pencil is 75" instead of "the price of that pencil is 75 U.S. cents". However, a representation of the first sentence would be ambiguous and incomparable with other prices. This violates the original purpose of RDF, that is, to permit metadata exchange and reuse. To achieve that goal, the metadata providers should be precise.

3. Some Syntactic Conventions and RDF Extentions For Logic

3.1. Universal Quantification

[BernersLee99] proposes a construct for universal quantification. Here is an extract from his examples.

```
<!-- All members of the W3C can access the member page -->
<forall id="baz" var="x" rdf:about="#x">
   <if><w3c:memberOf>http://www.w3.org/</w3c:memberOf>
       <then><w3c:canAccess>http://www.w3.org/Member</w3c:canAccess></then>
   </if></forall>
```

Additional properties (e.g. "atLeast", "atMost" and "part") would be interesting to specify some restrictions on the quantification. Here is an example.

```
<!-- At least 2% of persons like most of cats -->
<forall id="baz" atLeast="2%" var="p" rdf:about="#p">
   <if><rdf:type resource="#Person"/>
       <then><forall part="most" var="c" about="#c">
                <if><rdf:type rdf:resource="#Cat"/>
                    <then><objectOf><Liking><agent>#p</agent></Liking>
                          </objectOf></then>
                </if>
             </forall>
       </then>
   </if></forall>
```

Such a construct permits the definition of rules on the instances of a class, or in other words, to associate definitions to that class. Without restricting properties (e.g. "atLeast", "atMost" and "part"), the definition specifies relations "necessarily" connected to all instances of that class (that is, necessary conditions of membership to the class). Using `part="most"`, typical relations can be defined, but more precision is achieved with percentages (e.g. `part="75%"` or `atLeast="75%"`).

[RDFSchema] also permits one to define some restrictions on the use of a class by directly connecting classes via relations. Though this method is convenient for a few well-known special cases (generalization relations, exclusion relations and relation signatures), the semantics of such connections is unknown for other cases. Assume for example that two classes `Airplane` and `Wing` are connected by a relation "part". Does this mean that "any airplane has for part a wing" or "any wing is part of a plane" or "a wing is part of any plane" or "any airplane has for part all the wings"? We propose the first interpretation be adopted (i.e. the source of the relation is universally quantified and the destination existentially quantified).

3.2. Collections and intervals

The properties "atLeast", "atMost" and others such as "size" would also be convenient for containers, and the "forall" construct useful for quantifying over the members of a container. Consider for example the sentence "ten persons, including Fred and Wilma, have each approved a resolution". Since the persons may or may not have approved the same resolution, an existential quantifier must be used with an existential quantifier within to refer to the resolutions.

```
<!-- if p is a person member of the set {Fred, Wilma, ...}
     then there exists a resolution r that has for approver p -->

<Set rdf:ID="s" size="10">
  <rdf:li><Person rdf:ID="Fred"/></rdf:li>
  <rdf:li><Person rdf:ID="Wilma"/></rdf:li>
</Set>
<forall id="baz" var="p" rdf:about="#p">
   <if><rdf:type rdf:resource="#Person"/>
       <memberOf rdf:resource="#s"/>
       <then><exists var="r" rdf:about="#r">
               <rdf:type rdf:resource="#Resolution"/>
               <approver rdf:resource="#p"/>
             </exists></then>
   </if></forall>
```

The properties "atLeast" and "atMost" permit the delimitation of intervals. Here is an example.

```
<!-- Tom is the creator of 10 to 20 documents, including http://www.foo.org/bar --
<Set rdf:ID="s" atLeast="10" atMost="20"/>
   <rdf:li rdf:resource="http://www.foo.org/bar"/>
</Set>
<rdf:Description rdf:aboutEach="#s">
   <rdf:type rdf:resource="#Document"/>
   <creator><Person rdf:ID="Tom"/></creator>
</rdf:Description>
```

This last example could also be represented using the relations `minimalSize` and `maximalSize` which are part of the 120 basic relations of our top-level ontology. However, like conventions, if such common and basic relations are not adopted as standards, the comparison of RDF metadata (and therefore their retrieval, merge and reuse) will remain problematic.

Acknowledgments

Many thanks to Dr OLivier Corby and Pr Peter Eklund for their readings and corrections of this article.

References

[BernersLee99]
The Semantic Toolbox: Building Semantics on top of XML-RDF
http://www.w3.org/DesignIssues/Toolbox.html

[BRML]
Business Rules Markup Language
http://www.oasis-open.org/cover/brml.html

[CGs]
Conceptual Graphs
http://concept.cs.uah.edu/CG/Standard.html

[CYC]
: *The Upper Cyc® Ontology*
`http://www.cyc.com/cyc-2-1/cover.html`

[DC]
: *Dublin Core Metadata Initiative*
`http://purl.org/dc/`

[KIF]
: *Knowledge Interchange Format (KIF)*
`http://logic.stanford.edu/kif/kif.html`

[MCF/XML]
: *Meta Content Framework Using XML*
`http://www.w3.org/TR/NOTE-MCF-XML/#secA.`

[RDFMS]
: *Resource Description Framework (RDF) Model and Syntax*, W3C Recommendation, 22 February 1999
`http://www.w3.org/TR/1999/REC-rdf-syntax-19990222`

[RDFSchema]
: *Resource Description Framework (RDF) Schema Specification 1.0*, W3C Candidate Recommendation 27 March 2000
`http://www.w3.org/TR/2000/CR-rdf-schema-20000327`

[WN]
: *WordNet - a Lexical Database for English*
`http://cogsci.princeton.edu/~wn/`

Comments to the current author: philippe.martin@gu.edu.au
Comments to the W3C: www-rdf-comments@w3.org
Date: 28/05/2000