

Suggestions Toward RDF Semantics Improvement

— Inspired by the Lexical Grid

Cui Tao, Jyotishman Pathak, Harold Solbrig,
Wei-Qi Wei, and Christopher G. Chute
Division of Biomedical Statistics and Informatics,
Mayo Clinic, Rochester, MN 55905

March 13, 2010

Mayo Clinic has been coordinating a community-wide initiative, called Lexical Grid (LexGrid) [1], that is aimed at developing a common terminology model and programming interfaces for uniformly storing, representing, and querying ontologies and vocabularies. During our recent efforts on representing LexGrid using W3C specifications, we have discovered some challenges and proposed possible solutions [7, 8]. Here we want to share the related experience with the RDF community.

1 RDF Literals and Reification

In the biomedical domain, lexical information plays very important roles. In many cases, we have the frequent needs to represent annotations of annotations, (i.e., the source of a definition) or the relations between two annotation properties (i.e., one comment is the Chinese translation of another comment). Figure 1 shows a sample term from an OBO [4] ontology. In this example, we need to represent the annotation of another annotation. Line 4 in Figure 1(a) shows that the term has a definition “middle stages of reproductive phase.” which comes from source “[TAIR:lr]”. Figure 1(b) shows the RDF triple representation for it using reification. There are two potential issues, however, if we use RDF reification. First, in the original OBO definition, it only states that the value of the definition (the literal string) itself comes from TAIR:lr. The reification defines that the whole statement (the term has the definition) comes from the source. Therefore the reification changes the original semantics slightly. Secondly, using reification is

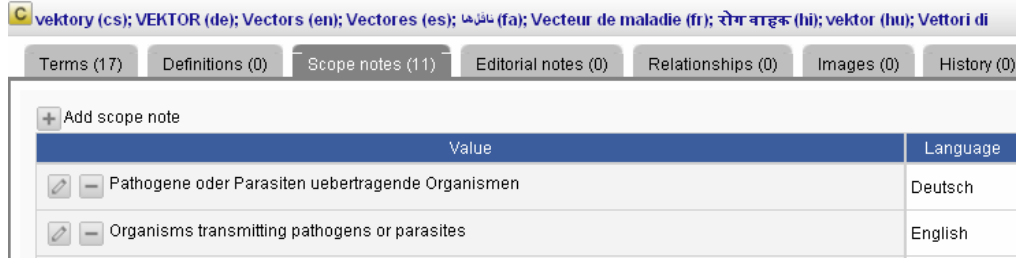
(a)

1. [Term]
2. id: FAO:0000025
3. name: mid reproductive
4. def: ‘‘middle stages of reproductive phase.’’ [TAIR:lr]
5. synonym: ‘‘principal growth stages 6.1-6.3’’

(b)

	Subject	Predicate	Object
1	A1	rdf:type	rdf:Statement
2	A1	rdf:subject	FAO:0000025
3	A1	rdf:predicate	skos:definition
4	A1	rdf:object	‘‘middle stages of reproductive phase.’’
5	A1	dc:source	TAIR:lr

Figure 1: An Example of Property and Property Reification (fungal_anatomy.obo)



The screenshot shows the AGROVOC ontology interface. The top bar displays the term 'vektory' in multiple languages: (cs), VEKTOR (de), Vectors (en), Vectores (es), ناقلات (fa), Vecteur de maladie (fr), रोग वाहक (hi), vektor (hu), Vettori di (it). Below this, a navigation bar shows 'Terms (17)', 'Definitions (0)', 'Scope notes (11)', 'Editorial notes (0)', 'Relationships (0)', 'Images (0)', and 'History (0)'. The 'Scope notes' tab is active, showing a table with two entries:

	Value	Language
☐ ☐	Pathogene oder Parasiten uebertragende Organismen	Deutsch
☐ ☐	Organisms transmitting pathogens or parasites	English

Figure 2: An Example from the AGROVOC ontology [2]

not the most efficient way in both the space and the performance perspectives [3]. Similar problems exist in the example in Figure 2 and Table 1 too. The translation is between the two literals, not two statements.

Currently Second-order descriptions in RDF can only be achieved via reification. OWL2 allows annotations of annotations, but still via reification using owl:annotatedSource, owl:annotatedProperty, and owl:annotatedTarget [5]. It will be helpful if the RDF community could consider to make lexical values first-class objects, so that we can assign them URIs. This way, we can define annotations of annotations or relations between two annotations without using reification.

	Subject	Predicate	Object
1	A1	rdf:type	rdf:Statement
2	A1	rdf:subject	AGROVOC:Vectors
3	A1	rdf:predicate	skos:scopeNote
4	A1	rdf:object	“Organisms transmitting pathogens or parasites”
5	A1	dc:language	English
6	A2	rdf:type	rdf:Statement
7	A2	rdf:subject	AGROVOC:Vectors
8	A2	rdf:predicate	skos:scopeNote
9	A2	rdf:object	“Pathogene oder Parasiten uebertragende Organismen”
10	A2	dc:language	Deutsch
11	A1	sns:translation	A2
12	sns:translation	rdf:subProperty	LexRDF:propertyLink

Table 1: RDF Triples for the Example of PropertyLink in Figure 2

2 RDF Collections

Another suggestions we would like to make is about RDF collections. Currently, when representing a collection of items using RDF triples, we have to use the `rdf:List` notation with `rdf:first`, `rdf:rest`, and `rdf:nil`. This notation implies that the elements in the list has a sequence or order which is not necessarily true. In addition, this notation makes the RDF triple representation long and hard to implement [6]. For example, Figure 2 shows the RDF triple representation of the example in Figure 3 using the `rdf:List` notation. In this case, we want to show that `PositiveChargedAminoAcid` is the intersection of `Amino Acid` and an restriction (`hasCharge` some positive). The triple notation uses a collection to represent the intersection. The two items of the intersection, however, do not have to follow a specific order. Instead, we want to show that these two items are the two members of the collection (or intersection). Figure 3 shows an alternate way to represent the collection using SKOS. We wonder if the RDF work group can adopt the SKOS notations or propose some notations similar to `SKOS:Member`.



Figure 3: An Example Concept from the Amino Acid Ontology

	Subject	Predicate	Object
1	PositiveChargedAminoAcid	rdf:type	owl:Class
2	PositiveChargedAminoAcid	owl:equivalentClass	@_A179
3	@_A179	rdf:type	owl:Class
4	@_A179	owl:intersectionOf	@_A180
5	@_A180	rdf:first	AminoAcid
6	@_A180	rdf:rest	@_A181
7	@_A181	rdf:first	@_A182
8	@_A182	rdf:type	owl:Restriction
9	@_A182	owl:onProperty	hasCharge
10	@_A182	owl:someValuesFrom	Positive
11	@_A181	rdf:rest	rdf:nil

Table 2: RDF Triples for the Example in Figure 3 Using RDF List

	Subject	Predicate	Object
1	PositiveChargedAminoAcid	rdf:type	owl:Class
2	PositiveChargedAminoAcid	owl:equivalentClass	@_A179
3	@_A179	rdf:type	owl:Class
4	@_A179	owl:intersectionOf	@_A180
5	@_A180	rdf:type	skos:Collection
6	@_A180	skos:Member	AminoAcid
7	@_A180	skos:Member	@_A182
8	@_A182	rdf:type	owl:Restriction
9	@_A182	owl:onProperty	hasCharge
10	@_A182	owl:someValuesFrom	Positive

Table 3: RDF Triples for the Example in Figure 3 Using SKOS

References

- [1] LexGrid: The Lexical Grid. <https://cabig-kc.nci.nih.gov/Vocab/KC/index.php/LexGrid>, 2009.
- [2] Agrovoc: from a thesaurus to an ontology. <http://aims.fao.org/en/website/AGROVOC-Concept-Server>.
- [3] Joe Futrelle. Harvesting rdf triples. In *Proceedings of the International Provenance and Annotation Workshop (IPAW'06)*, pages 64–72, 2006.
- [4] The open biomedical ontologies. <http://www.obofoundry.org/>.
- [5] Owl 2 rdf translation of annotations. http://www.w3.org/TR/owl2-mapping-to-rdf/#Translation_of_Annotations.

- [6] Practical problems with `rdf:parseType="Collection"` implementation.
<http://osdir.com/ml/web.rdf/2003-08/msg00066.html>.
- [7] C. Tao, J. Pathak, H.R. Solbrig, and C.G. Chute. LexOWL: A bridge from LexGrid to OWL. In *Proceedings of the First International Conference of Biomedical Ontology (ICBO 09)*, pages 131–134, Buffalo, New York, July 2009.
- [8] C. Tao, J. Pathak, H.R. Solbrig, W-Q Wei, and C.G. Chute. LexRDF model: An RDF-based unified model for heterogeneous biomedical ontologies? In *Proceedings of the Semantics for the Rest of Us Workshop (SemRUs09), collocated with the 8th International Semantic Web Conference (ISWC 09)*, Washington DC, October 2009.