The NCBO OBOF to OWL Mapping

Dilvan A. Moreira[1,2,*], Christopher J. Mungall[3], Nigam H. Shah[1], Stuart Aitken[4], John-Day Richter[3], Timothy Redmond[1] and Mark A. Musen[1]

[1]Stanford Center for Biomedical Informatics Research, Stanford University, 251 Campus drive, MSOB, Stanford, CA.

[2]University of São Paulo, SCC-ICMC-USP 400 Trabalhador Sancarlense, São Carlos, Brazil.

[3]Lawrence Berkeley National Laboratory, 1 Cyclotron Road MS64R0121, Berkeley, CA.

[4]Centre for Intelligent Systems and their Applications, The University of Edinburgh, 11 Crichton Street, Edinburgh, UK.

ABSTRACT
Two of the most significant formats for biomedical ontologies are the Open Biomedical Ontologies Format (OBOF) and the Web Ontology Language (OWL). To make it possible to translate ontologies between these two representation formats, the National Center for Biomedical Ontology (NCBO) has developed a mapping between the OBOF and OWL formats as well as inter-conversion software. The goal was to allow the sharing of tools, ontologies, and associated data between the OBOF and Semantic Web communities.

OBOF does not have a formal grammar, so the NCBO had to capture its intended semantics to map it to OWL.

This official NCBO mapping was used to make all OBO Foundry ontologies available in OWL.

Availability: This mapping functionality can be embedded into OBO-Edit and Protégé-OWL ontology editors. This software is available at:
http://bioontology.org/wiki/index.php/OboInOwl:Main_Page

1 INTRODUCTION

With the explosion of ontologies used to drive work in e-commerce, e-science, and many other application areas, the World Wide Web Consortium (W3C) initiated a standards process that led to the recommendation of OWL (McGuinness and Harmelen, 2004), the Web Ontology Language. There is now a significant interest in using the life sciences domain as a “focus” for W3C semantic web activity (Ruttenberg et al., 2007). In this light, biological data described using OBOF (Open Biomedical Ontologies Format) ontologies are a prime resource, and there is great interest from the Semantic Web community to access both the ontologies and the data that have been described (annotated) using these ontologies.

On the other hand, the bio-ontology community needs to leverage the rapid progress that is being made in Semantic Web technologies, especially with OWL. As a result, there is a strong interest in a mapping between the OBOF and OWL.

OBOF is a tag-based format, and its specification can be found online (*http://www.geneontology.org /GO.format.obo-1_2.shtml*). For the NCBO mapping, we used OBOF Version 1.2 and OWL Version 1.0 (sublanguage OWL-DL). Ontology files in OBOF 1.2 consist of a header, a set of terms, and a set of relationships.

[*]To whom correspondence should be addressed (dilvan@gmail.com).

Performing a translation between any formats, when there is some ambiguity involved (as it is the case with OBOF), presents interpretation problems, and the first practical barrier is obtaining a parser that works as intended by the developers of the format. The most reliable solution, to guarantee accurate parsing, is to use a parser written specifically to work with OBOF, this meant using the parser that is built into OBO-Edit (Day-Richter et al, 2007), the most used editor for OBOF ontologies. OBO-Edit is open source software, so its parser can be reused without restrictions. For OWL the parser built into Protégé (Noy et al, 2003) was used in our tools. The conversion problem is then confined to establish a correspondence between OBOF constructs and OWL constructs. Our conversion software uses the respective OBO-Edit API and Protégé OWL API to carry out the actual transformation from OBOF to OWL format and vice-versa. This implementation was written in Java 1.5. In addition, an alternative implementation was written as an XML Style Sheet Transform (XSLT) to convert OBO-XML to OWL.

We have to note the exception that OBOF instances (`Instance stanzas`) and certain tags (`is_anonymous`, `transitive_over`, `is_reflexive`, `is_anti_symmetric`, `builtin` and `is_metadata_tag`) are not mapped into OWL in this mapping. These constructs will not be fully specified in OBOF until its next release.

2 MAPPING BASIC ONTOLOGY CONSTRUCTS

It is possible to establish direct one-to-one correspondences from the two basic ontology constructs in OBOF, Terms and Relations, to OWL:

Terms: OBOF terms are mapped into OWL classes (`owl:Class`). Child terms (declared using the is_a relationship tag) use the subclasses (`rdfs:SubClassOf`) relationship. An example is shown in Table 1: The OWL representation equates the OBOF term to a Named Class in OWL using necessary conditions to define the class.

The OBOF tags `intersection_of` and `union_of` allow the creation of compositional terms in OBOF based on intersection or union conditions respectively (using necessary and sufficient conditions). In OWL, Defined Classes represent compositional objects. Compositional terms in OBOF are mapped using Defined Classes in OWL.

Relations: As shown in Table 1, the hierarchical relationships among OBOF terms have a natural mapping to OWL constructs. The OBOF *is_a* tag is mapped to the `rdf:subClassOf` predicate (as it represents a subclass relationship). All other OBOF relationship definitions (`[Typedef]`), such as `part_of` or `develops_from`, are mapped directly into OWL object properties (`owl:ObjectProperties`). These definitions may have additional declarations about the inverse relationship and about transitivity that are also mapped into OWL constructs.

Table 1. OBOF terms are mapped directly into OWL Named-Classes

OBOF	OWL
[Term]	<owl:Class rdf:ID="#SO_0000042">
id: SO:0000042	<rdfs:label
name: pseudogene_attr	rdf:datatype="&xsd;string">
is_a: SO:0000733	pseudogene_attr
	</rdfs:label>
	<rdfs:subClassOf
	rdf:resource="#SO_0000733"/>
	</owl:Class>

Relationships between OBOF terms are encoded by the OBOF *relationship* tag at the term (class) level. For example, if a given OBOF file states that "nerve terminal (`GO:0043679`) is *part_of* neuron projection (`GO:0043005`)", the equivalent OWL representation should state that **all** cell structures (individuals) of the class "nerve terminal" are *part_of* **some** structure of the class "neuron projection". In order to achieve the correct semantics intended in the OBOF format, the relationship definitions are translated to *all-some* quantifications over individuals in OWL and are encoded using the `owl:Restriction` construct, on a certain property (relation), with an `owl:someValuesFrom` quantification, as shown in Table 2.

Table 2. Mapping OBOF relationships

OBOF
relationship: part_of GO:0000087
OWL
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty>
<owl:ObjectProperty rdf:ID="#part_of"/>
</owl:onProperty>
<owl:someValuesFrom>
<owl:Class rdf:ID="GO_0000087"/>
</owl:someValuesFrom>
</owl:Restriction>
</rdfs:subClassOf>

Mapping OBOF relationships as *all-some* quantifications over properties of individuals in OWL.

3 UNIQUE IDENTIFIERS

Both the OWL and OBOF representations require a unique identifier (ID) for the entities in the ontology. The OBO foundry recommends that the term identifier be in bipartite form, with an ID-space and a 'local' identifier (typically numeric) separated by the colon character – for example, `GO:0008045`. The resulting ID would be unique among all OBOF ontologies. In OWL, the unique identifiers are always Uniform Resource Identifiers (URIs). These IDs are completely independent of the name(s) associated with these entities.

We defined a protocol for composing an OWL ID from an OBOF ID. OWL requires all IDs (`rdf:ID`) to be well formed URIs. As a result, there are many alphanumeric characters that may not appear in OWL IDs. The OBOF identifier must be manipulated in order to render it as a valid URI. In this mapping, each ontology has a *base URI*, and, based on it, a URI is constructed for each term and relationship from its OBO ID:

- If the ID has a prefix, such as `GO:0000001`, their URI is constructed concatenating their prefix (GO) onto their *base URI*, followed by a hash ('#') symbol. This URI is then concatenated with the local part of their ID (`0000001`). If this string is numeric (as is commonly the case), then the characters must be prefixed with the OBOF ID-space, followed by an underscore. In this way, OBOF IDs of the form `GO:0000001` from an ontology that have *http://purl.org/obo/owl/* as its *base URI* are mapped to URIs of the form *http://purl.org/obo/owl/GO#GO_0000001*.
- If the ID has no prefix (what is usually the case for relationships), their URI is constructed concatenating their OBOF default-namespace (declared in the OBOF file) onto their *base URI*. This URI is then concatenated with their ID. In this manner, IDs, such as `part_of` in the GO, will be mapped to *http://purl.org/obo/owl/gene_ontology#part_of*, where *gene_ontology* is the OBOF default-namespace for GO. If the OBOF default-namespace is not declared, then the ID will be mapped to the *base URI*'s namespace or, if the ID refers to a relationship definition, it can be explicitly assigned to a particular namespace where this relationship is defined.

4 METADATA

When developers create an ontology in OBOF, they describe both formal ontological elements (i.e., the relationships among entities) and metadata. OBOF provides a uniform mean of encoding relationships holding among a set of entities, terminological and lexical aspects of those entities (synonyms, comments, text definitions), and information pertaining to the ontology lifecycle (including tracking of obsolete terms and metadata for migrating annotations forward across versions). This metadata is very useful for human understandability and can be added to the ontology as a whole or with individual classes in that ontology.

OWL, in and of itself, does not provide a standard way of capturing this metadata. Instead, it allows ontology developers to develop their own ways of capturing ontology metadata.

Any full translation from OBOF to OWL must include a mechanism to accommodate such metadata elements. For that, we have created a set of new classes and properties, a small metadata ontology, to be used in annotation properties `owl:AnnotationProperty` called **`oboInOwl`**. This metadata ontology has the URI *http://www.geneontology.org/formats/oboInOwl* (conventionally abbreviated as the XML qname `oboInOwl:`).

For each of the OBOF metadata elements, we have specified corresponding elements in the **`oboInOwl`** ontology. We also make use of two RDFS properties – `rdfs:label` (for names) and `rdfs:comment` (for comments).

5 MAPPING EACH PART OF AN OBOF FILE

Ontology files in OBOF consist of a header and sets of terms and relationships. The header has documentation and information tags, such as format version and saved date, and comes first in the file. Terms and relationships can be mixed and distributed along the document.

OBOF Header

OBOF header constructs are mapped to OWL annotations (`owl:AnnotationProperty`) in the ontology class (`owl:Ontology`). OBOF tags processed by parsers are not mapped; there is no need to tie the mapping to the way OBOF and OWL process parsing commands, such as imports. The OBOF tag `format-version` is ignored, as there is no need to map an OBOF file to a particular OBOF version. Table 3 shows all header constructs.

Table 3. OBOF ontology header metadata

OBOF	OWL
data-version	oboInOwl:hasVersion
Date	oboInOwl:hasDate
saved-by	oboInOwl:savedBy
Subsetdef	oboInOwl:hasSubset
Synonymtypedef	oboInOwl:hasSynonymType
default-namespace	oboInOwl:hasDefaultNamespace
Remark	rdfs:comment
Idspace	oboInOwl:hasIdSpace
format-version	Ignored
auto-generated-by	Ignored (Generated in each write)
Import	Ignored (Processed by the parser)
default-relationship-id-prefix	Ignored (Processed by the parser)
id-mapping	Ignored (Processed by the parser)

OBOF Terms

Entities in OBOF are referenced as Terms. Table 1 showed the basic mapping of terms, Table 4 shows all possible constructs. This table describes the terminological information associated with entities in OBOF (mapped to OWL classes), their relationships with other entities, cross-references to other ontologies, as well as restrictions, if any, on the terms. Note that there is no semantic difference between `rdf:about` and `rdf:ID` tags (syntactically `rdf:ID` provides an additional check since the same name can only appear once in the scope). Some OBOF tags will be fully specified only in OBO 1.3, so they are not mapped at this time.

Table 4. Term information

OBOF	OWL
[Term]	owl:Class
Id	rdf:ID / rdf:about
Name	rdfs:label
Comment	rdfs:comment
is_a	rdfs:subClassOf
is_anonymous	To be fully specified in OBO 1.3
alt_id	oboInOwl:hasAlternateID
Namespace	oboInOwl:hasOBONamespace
Def	oboInOwl:hasDefinition
Comment	rdfs:comment
Subset	oboInOwl:hasSubset
Synonym	oboInOwl:hasSynonym oboInOwl:hasExactSynonym (scope=EXACT) oboInOwl:hasNarrowSynonym (scope=NARROW) oboInOwl:hasBroadSynonym (scope=BROAD) oboI-nOwl:hasRelatedSynonym(scope=RELATED)
Xref	oboInOwl:hasDbXref
intersetion_of	owl:intersectionOf
union_of	owl:unionOf
Disjoint_from	owl:disjointFrom
Relationship	owl:restriction
Builtin	To be fully specified in OBO 1.3

OBOF Relationships

Table 2 showed the mapping of OBOF relationships. Table 5 shows the OBOF constructs that describe logical properties of relationships showing their correspondent in OWL. They are used for reasoning over an ontology. Some OBOF tags will be fully specified only in OBO 1.3, so they are not mapped at this time.

Table 5. Relationship information

OBOF	OWL
is_a	rdfs:subPropertyOf
Range	rdfs:range
Domain	rdfs:domain
is_symmetric	owl:SymmetricProperty
is_anti_symmetric	To be fully specified in OBO 1.3
is_transitive	owl:TransitiveProperty
inverse_of	owl:inverseOf
transitive_over	To be fully specified in OBO 1.3
is_cyclic	oboInOwl:isCyclic (AnnotationProperty)
is_reflexive	To be fully specified in OBO 1.3
is_symmetric	owl:SymmetricProperty
is_metadata_tag	To be fully specified in OBO 1.3

OBOF Obsolete entities

Obsolete terms and relationships can have tags with information about direct substitutes, `replace_by`, or similar concepts, `consider` (Table 6).

Table 6. Obsolete terms and relationships

OBOF keyword	OWL annotation property	OWL type
`replaced_by`	`oboInOwl:replacedBy`	`xsd:string`
`Consider`	`oboInOwl:consider`	`xsd:string`

More complex mapping examples

Table 7 shows a more complex example of mapping: a compositional term from the Sequence Ontology (Eilbeck et al, 2005) in OBOF is mapped to a Defined Class in OWL based on intersection, as the two constructs are semantically equivalent.

Table 7. A more complex example of the mapping of OBOF terms to OWL Classes

OBOF

```
[Term]
id: SO:0000111
name: transposable_element_gene
def: "A gene encoded … yeast." [SO:ke]
intersection_of: SO:0000704 ! gene
intersection_of: part_of SO:0000101 !t…
```

OWL

```
<owl:Class rdf:ID="SO_0000111">
   <rdfs:label rdf:datatype="&xsd;string">
      transposable_element_gene
   </rdfs:label>
   <oboInOwl:hasDefinition>
      A gene encoded … yeast.
   </oboInOwl:hasDefinition>
   <owl:equivalentClass>
      <owl:Class>
         <owl:intersectionOf rdf:parseType="Collection">
            <owl:Class rdf:ID="SO_0000704"/>
            <owl:Restriction>
               <owl:someValuesFrom>
                  <owl:Class rdf:ID="SO_0000101"/>
               </owl:someValuesFrom>
               <owl:onProperty>
                  <owl:ObjectProperty
                       rdf:about="#part_of"/>
               </owl:onProperty>
            </owl:Restriction>
         </owl:intersectionOf>
      </owl:Class>
   </owl:equivalentClass>
</owl:Class>
```

Table 8 shows how `is_a` relationships are mapped to OWL subclass relationships and how more complex relationships such as `part_of` are mapped to `owl:ObjectProperties`.

Table 9 lists the classes and properties used for representing OBOF metadata entities using the `oboInOwl` metadata ontology (The asterisk denotes optional constructs).

6 RESULTS

The NCBO mapping can only be useful if we provide a straightforward means for using it. We have developed software that can be readily embedded into different work environments. The mapping can thus be used with software that function as:

- a command line tool, for batch processing (Moreira and Musen, 2007),
- a Tab plug-in (Moreira and Musen, 2007) to allow Protégé-OWL, a popular tool among the Semantic Web community, to read and save ontologies in OBOF,
- perl and XSLT scripts, for use in web/XML applications (*http://search.cpan.org/~cmungall/go-perl*),
- a Tab plug-in for Protégé-OWL that allows for viewing and editing of lexical information, captured in the `oboToOwl` metadata, in a manner similar to OBO-Edit (Day-Richter et al., 2007), and
- a plug-in to allow the OBO-Edit, a popular tool among the OBO community, to read and save OBOF ontologies in OWL.

All the software described is available online at *http://bioontology.org/wiki/index.php/OboInOwl:Main_Page*. In addition, the LSW tool is also capable of rending the oboToOwl metadata elements for human users (available at *http://esw.w3.org/topic/LSW*).

The NCBO mapping is already being widely adopted by the biomedical community. It was used to convert all ontologies from the OBO Foundry to OWL. As a result the OBO Foundry ontologies are now available in OWL format from http://purl.org/obo (for example, the GO is available via the URL *http://purl.org/obo/owl/GO*).

It is now possible to use these ontologies in OWL with other Semantic Web technologies to integrate biomedical data from different sources. For instance, it is now possible to read the GO ontology and GO annotations (tab delimited format) into OWL (Moreira et al, 2007).

In a larger scale, the W3C Health Care and Life Sciences Interest Group (HCLSIG) demo (Ruttenberg, 2007) populated a RDF data store with these OWL ontologies, together with biological annotations relevant to neuroscience, to demonstrate the value of semantic web technology. This database is now available online as the Neurocommons RDF Store, where 7 OBO ontologies are integrated with 10 other ontologies and data sources in one repository (triple store) accessible using SPARQL queries.

7 CONCLUSION

The NCBO mapping is serving as an interface between the biomedical community and users of Semantic Web technologies. Both communities benefit from a simple mechanism to faithfully translate between OBOF and OWL. Now, users of OBOF ontologies are able to leverage the rapid progress that is being made in computer science—especially in Semantic Web technologies—and the Semantic Web community will be able to interoperate with OBOF bio-ontologies and the data they annotate.

Table 6. Subclasses and properties

OBOF	OWL

[Typedef]
id: OBO_REL:proper_part_of
name: proper_part_of
is_a: OBO_REL:part_of
def: "As for … distinct" [PMID:15892874]
inverse_of: OBO_REL:has_proper_part
is_transitive: true
	<owl:ObjectProperty rdf:about="&oboRel;proper_part_of">
	   <rdfs:label … >proper part of</rdfs:label>
	   <rdfs:subPropertyOf rdf:resource="&oboRel;part_of"/>
	   <oboInOwl:hasDefinition>
	      <oboInOwl:Definition>
	         <rdfs:label ... >
	            As for … distinct
	         </rdfs:label>
	         <oboInOwl:hasDbXref>
	            <oboInOwl:DbXref>
	                <rdfs:label ...>
	                   PMID:15892874
	                </rdfs:label>
	            </oboInOwl:DbXref>
	         </oboInOwl:hasDbXref>
	      </oboInOwl:Definition>
	   </oboInOwl:hasDefinition>
	   <owl:inverseOf rdf:resource="&oboRel;has_part"/>
	   <rdf:type rdf:resource="&owl;TransitiveProperty"/>
	</owl:ObjectProperty>

Table 7. OBOF metadata entities in OWL

OBOF entity description	OWL class description

xref: ***dbxref_name*** *"**description**"*	<oboInOwl:DbXref>
	   <rdfs:label …> ***dbxref_name*** </…>
	   <rdfs:comment …> ***description*** </…>
	   <oboInOwl:hasURI xsd:anyuri> URI </…> *
	</…>

synonym: "***text***" scope ***type*** [***dbxref*** …]	<oboInOwl:Synonym>
	   <rdfs:label …> ***text*** </…>
	   <oboInOwl:hasDbXref> ***DbXref*** </…>
	   <oboInOwl:hasSynonymType> ***SynonymType*** </…> *
	</…>

synonymtypedef: ***name description scope***>	<oboInOwl:SynonymType rdf:ID="***name***">
	   <rdfs:label …> ***description*** </…>
	   <oboInOwl:restrictedToScope> ***scope*** </…> *
	</…>

subsetdef: ***name*** "***description***"	<oboInOwl:Subset rdf:ID="***name***"> <rdfs:comment …> ***description*** </…> * </…>

definition: ***text*** [***dbxref1*** …]	<oboInOwl:Definition>
	   <rdfs:label…> ***text*** </…>
	   <oboInOwl:hasDbXref> ***DbXref*** </…> *
	</…>

idspace: ***idspace URI*** "***description***"	<oboInOwl:IdSpace>
	   <rdfs:label …> ***idspace*** </…>
	   <oboInOwl:hasURI xsd:anyuri> ***URI*** </…>
	   <rdfs:comment …> ***description*** </…> *
	</…>

ACKNOWLEDGEMENTS

The authors gratefully acknowledge the support and advice from all of our colleagues in the National Center for Biomedical Ontology, which led to the development of this mapping, especially to Suzanna Lewis. We also gratefully acknowledge the expertise and input we received from the wider OBOF and OWL communities. Finally, we are extremely appreciative to our users whose feedback and support continues to guide our work.

Funding: NCBO is supported by a grant from the National Institutes of Health (NIH grant U54 HG004028, through the NIH Roadmap program, M.A. Musen, PI). This work was also supported by a scholarship from CAPES-Brazil.

REFERENCES

Day-Richter J, Harris MA, Haendel M, Lewis S. (2007) OBO-Edit - An Ontology Editor for Biologists. *Bioinformatics* 23(16):2198-2200.

Eilbeck, K, Lewis SE, Mungall CJ, Yandell M, Stein L et al. (2005) The Sequence Ontology: a tool for the unification of genome annotations. *Genome Biol*. 6:R44.

McGuinness D, Harmelen F (Eds) (2004) OWL Web Ontology Language Overview. W3C Recommendation 10 February 2004. http://www.w3.org/TR/2004/REC-owl-features-20040210/

Moreira DA, Musen MA. (2007) OBO to OWL: A Protégé OWL Tab to Read/Save OBO Ontologies. *Bioinformatics* 23(14):1868-1870.

Moreira DA, Shah NH, Musen MA. (2007) Interpretation Errors related to the GO Annotation File Format, *AMIA 2007 Symposium Proceedings*, Chicago, November 2007, pp. 538-542.

Noy NF, Crubezy M, Fergerson RW, Knublauch H, Tu SW, Vendetti J, Musen MA. (2003) Protégé-2000: an open-source ontology-development and knowledge-acquisition environment. *AMIA Annu Symp Proc. 2003*; :953.

Ruttenberg A. (2007) Harnessing the Semantic Web to Answer Scientific Questions: A Health Care and Life Sciences Interest Group demo. *WWW2007*, Banff, Canada http://esw.w3.org/topic/HCLS/Banff2007Demo

Ruttenberg A, Clark T, Bug W, Samwald M, Bodenreider O et al. (2007) Advancing translational research with the Semantic Web, *BMC Bioinformatics*, 8(Suppl 3):S2.