

MIREOT: the Minimum Information to Reference an External Ontology Term

Mélanie Courtot ^{a,*}, Frank Gibson ^b, Allyson L. Lister ^c, James Malone ^d, Daniel Schober ^e,
Ryan R. Brinkman ^a and Alan Ruttenberg ^f

^a *BC Cancer Agency, Vancouver, BC, Canada*

E-mail: mcourtot@gmail.com, rbrinkman@bccrc.ca

^b *Abcam plc, 332 Cambridge Science Park, Cambridge, CB4 0WN, UK*

E-mail: fgibson@gmail.com

^c *CISBAN and School of Computing Science, Newcastle University, Newcastle upon Tyne, UK*

E-mail: a.l.lister@newcastle.ac.uk

^d *The European Bioinformatics Institute, Cambridge, CB10 1SD, UK*

E-mail: malone@ebi.ac.uk

^e *Institute of Medical Biometry and Medical Informatics (IMBI), University Medical Center, 70104
Freiburg, Germany*

E-mail: schober@imbi.uni-freiburg.de

^f *Science Commons, Cambridge, MA, USA*

E-mail: alanruttenberg@gmail.com

Abstract. While the Web Ontology Language (OWL) provides a mechanism to import ontologies, this mechanism is not always suitable. Current editing tools present challenges for working with large ontologies and direct OWL imports can prove impractical for day-to-day development. Furthermore, external ontologies often undergo continuous change which can introduce conflicts when integrated with multiple efforts. Finally, importing heterogeneous ontologies in their entirety may lead to inconsistencies or unintended inferences. In this paper we propose a set of guidelines for importing required terms from an external resource into a target ontology. We describe the methodology, its implementation, present some examples of this application, and outline future work and extensions.

Keywords: ontology import, data integration, MIREOT

1. Introduction

The ability to share and reuse existing ontological resources is an important consideration when developing a new ontology. For example, when developing an ontology related to the biomedical domain, it may be useful to include terms from the Gene Ontology (GO) (Gene Ontology Consortium, 2004) to represent biological processes or from Phenotypic Quality Ontology (PATO) (Phenotypic Quality Ontology website) to represent properties of entities. Ontologies such as GO and PATO are built collaboratively by communities of experts and are the products of substantial effort. Recapitulating this work instead of reusing it represents a duplication of development effort and results in multiple ontologies covering the same domain. It could also result in projects having different unique identifiers to denote the same entity, which would require post-hoc, potentially error-prone, identifier mapping systems to enable data integration. While it appears that building upon existing vocabularies is the best way to proceed, ontology developers are faced with a number of practical challenges when trying to do so. The easiest way to integrate an existing body of work is to rely on the Web Ontology Language (OWL) (Smith, M.K. et al., 2004) mechanism *owl:imports*, which imports the external resource as a whole. However, current limitations in tools and reasoners can sometimes make this impractical. Popular OWL tools (*e.g.*, Protégé (The Protégé

*Corresponding author: Mélanie Courtot, BC Cancer Agency, Vancouver, BC, Canada.

Ontology Editor) and Pellet (Sirin E. et al., 2007)) can neither load nor reason over very large ontologies such as the NCBI Taxonomy (Wheeler D. L. et al., 2005) or the Foundational Model of Anatomy (Golbreich C. et al., 2006), making direct OWL imports of such resources impractical. Furthermore, external ontologies may have been constructed using design principles which do not align with the practice of the ontology requiring their import. In this instance, wholly importing such ontologies could lead to inconsistencies or unintended inferences. Other import options are possible, for instance using software that extracts a *module* (Grau BC et al., 2007) of the external ontology. A module can be seen as a fragment of an external ontology that, when imported by an other ontology, allows the same inferences to be drawn with respect to the classes of interest as if the whole ontology had been imported. This solution allows developers to pick only pieces of the source ontology (and thus overcome size issues) without losing any reasoning power. However, if an extracted module is to be useful, the external ontology needs to be structured in a way that is compatible with the importing ontology (*e.g.*, using the same upper ontology and relationship types), and the logical axioms need to be accurate. This is not always the case at the current stage of development of some ontologies. For example, during the development of the Ontology of Biomedical Investigations (OBI) (OBI Ontology), importing the root class of the Common Anatomy Reference Ontology (CARO) (Haendel M.A. et al., 2008) was not desired as its definition intersected multiple classes in OBI, making it difficult to determine how the two ontologies aligned. In addition, although software that extracts modules are available, most are only in early stages of development.

We tried several modularization tools (Grau BC et al., 2007; Jimenez-Ruiz E., et al., 2008; Seidenberg J., Rector A., 2006; Sirin E. et al., 2007). All of them discarded annotations, resulting in modules containing only the class declarations and no annotation properties, such as labels or definitions. We also experienced software crashes on large ontologies (the size of the ontologies capable of being loaded varied with the tool, for example we were able to load ChEBI (Degtyarenko, K. et al., 2008) with SWOOP but not with Protégé 3.4). One tool (Seidenberg J., Rector A., 2006) had undocumented assumptions about the form of URIs used as class names and therefore extracted empty modules. The other tools described were able to extract modules by automatically determining their size. This resulted in either a single term or a large number of terms being extracted, depending on the provided arguments, as the tools attempt to approximate a module without discarding potentially useful information. These large modules undermine the goal of having imports of a manageable size. Our conclusion was that the current ontology tool set is in the early stages of development and, though promising, do not address our current needs.

To address these issues we developed a set of guidelines for importing terms from multiple ontology resources, avoiding the overhead of importing the complete ontology from which the terms derive. The Minimum Information to Reference an External Ontology Term (MIREOT) guidelines were created to aid the development of OBI. OBI uses the Basic Formal Ontology (BFO) Grenon P. et al. (2004) as an upper-level ontology and has been submitted for inclusion in the Open Biomedical Ontologies (OBO) Foundry (Smith B. et al., 2007). One of the fundamental principles of the OBO Foundry is to reuse, where appropriate, existing ontology resources, therefore avoiding duplication of effort and ensuring orthogonality. MIREOT provides a mechanism by which external ontology terms can be selectively imported, even if they do not use a particular upper ontology or OWL DL, and hence contribute to the realization of OBO Foundry principles.

2. Policy

In deciding upon a minimum unit of import, our first step was to consider the practice of other ontology efforts. For example, in the GO, the intended denotation of classes remains stable such that even when the ontology is repaired or reorganized, the effects of such changes do not affect the intended meaning of individual terms. Rather, the changes are towards more carefully expressing the logical relations between them. When a term's changes meaning, the term is deprecated (Go editorial style guide). We can therefore consider a term as stable, in isolation from the rest of the ontology, and use terms (*i.e.*, individual classes in isolation from the ontology) as basic unit of import. The current implementation of MIREOT has been

limited to import of terms from other ontologies that aspire to the OBO Foundry, and so adhere to a similar deprecation policy.

The minimum amount of information needed to reference an external term is its URI (*i.e.*, the identifier for this term) and its source ontology URI (*i.e.*, where the term comes from). Generally, these items remain stable and can be used to unambiguously reference the external term. The minimum amount of information to then integrate this class in the importing target ontology is its desired position in the hierarchy, specifically the URI of its direct superclass (*i.e.*, under which class the term is to be asserted).

Taken together, the following minimal set is enough to consistently reference an external term:

- **source ontology URI** The logical URI of the ontology containing the external term to be imported.
- **source term URI** The logical URI of the specific term to import.
- **target direct superclass URI** The logical URI of the direct asserted superclass in the importing ontology.

To ease development of the importing ontology it is also recommended, although not required, that additional information about the external class be added, such as its label and textual definition, or any other kind of information that may be deemed useful by the ontology developers. This additional information, when appropriate, is mapped into the importing ontology's annotation properties. As it is prone to modification by the source ontology developers (*e.g.*, when updating a definition), it is stored in a separate file that can be removed and rebuilt on a regular basis, allowing for regular update within the importing target ontology.

3. Implementation

An implementation of the MIREOT guidelines was performed in the context of the OBI project, and can be decomposed into a two-step process:

1. Gather the minimum information for the external class.
2. Use this minimum information to fetch additional elements, like labels and definitions.

Once the external term is identified for import, the first step is to gather the corresponding minimum information set. This set is stored in a file called *external.owl* (all scripts and files are available under the OBI Subversion Repository (OBI scripts, 2008)). In the current implementation, a Perl script, *add-to-external.pl*, can be used to append the minimum information set for a given external term to the *external.owl* file, or the information can be entered by directly editing the OWL file. The script takes as arguments the identifier of the external class to be imported and its parent class in the target hierarchy. In addition, a mapping between the prefix used in the identifier and the external source ontology URI is built into the script. For example, when requesting the term *CL:0000767* (see below), the script maps the *CL:* prefix to its source ontology URI <http://purl.org/obo/owl/CL>. Curators therefore need only specify the ID of the external class to import (rather than the full URI) and the ID of the class it should be imported under.

In the current implementation, the additional information can be obtained programmatically via SPARQL (SPARQL Query Language) CONSTRUCT queries (Figure 1). These queries (SPARQL queries template file, 2008) specify, for each source ontology, which extra elements about the term is to be extracted, such as the definition and preferred label, and how to map these into the corresponding OBI annotation properties.

For example, in the current OWL rendering of OBO files, definitions are individuals and the *rdfs:label* of those individuals record the text of the definitions. Within the OBI implementation of the MIREOT guidelines, the value of the *rdfs:label* of the *oboInOwl:Definition* will be set to the value of http://purl.obolibrary.org/IAO_0000115 (*i.e.*, *iao:definition*). Only annotation properties which map directly to the target ontology's own metadata are copied; new properties, if not specified in the source ontology, are not created.

```

prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix owl: <http://www.w3.org/2002/07/owl#>
prefix obi: <http://purl.obofoundry.org/obo/>
prefix obo: <http://www.geneontology.org/formats/oboInOwl#>

construct
{
  _ID_GOES_HERE_ rdfs:type owl:Class.
  _ID_GOES_HERE_ alias:preferredTerm ?label.
  _ID_GOES_HERE_ rdfs:label ?label.
  _ID_GOES_HERE_ alias:definition ?definition.
}
where
{
  {
    _ID_GOES_HERE_ rdfs:label ?label.
  }
  UNION
  {
    _ID_GOES_HERE_ obo:hasDefinition ?blank.
    ?blank rdfs:label ?definition
  }
}

```

Fig. 1. Template SPARQL query. For convenience, we use `alias:preferredTerm` and `alias:definition` to reference our annotations properties `IAO_0000111` and `IAO_0000115` (IAO website, 2008) respectively. The `_ID_GOES_HERE_` pattern will be replaced by the script when building the CONSTRUCT query.

Finally, a script *create-external-derived.lisp*, iterates through the minimum information stored in *external.owl*. Depending on the source ontology URI of each of the imported terms, it then selects the correct SPARQL template and substitutes the relevant ID. The queries are then executed against the Neurocommons OBO SPARQL endpoint (Neurocommons OBO SPARQL endpoint; Ruttenberg A et al., 2009). This supplementary information is stored in a second file, *externalDerived.owl*. This file can be removed on an ad-hoc basis (e.g., before releasing new versions of the importing ontology) so that it can then be rebuilt via script based on *external.owl* in order to refresh the additional information (e.g., label). The two files, *external.owl* and *externalDerived.owl*, are then imported by the target ontology, providing the necessary information to the editors while at the same time keeping it independent from the importing ontology's proprietary classes (see Figure 2). This introduces an additional level of modularity, separating the domain ontology of interest from the external ontologies.

In the following sections we present two different cases of application of the MIREOT guidelines, implemented during the OBI development.

Use Case One - Basophil and Cell classes

We replaced the OBI *cell* class with that from the Cell Type (CL) ontology (Bard J. et al., 2005). CL is part of the OBO Foundry effort, and we would like to use the *cell* class as defined by this resource, instead of creating our own duplicated class. This class can then be used in turn to import other classes as needed. For example the following invocation of the *add-to-external.pl* script:

```
perl add-to-external.pl CL:0000767 CL:0000000
```

will add the class *basophil* (CL:0000767) as subclass of the class *cell* (CL:0000000), and set the source ontology URI as `http://purl.org/obo/owl/CL`. Once imported, the *basophil* and *cell* classes can be used like any other OBI class. For example, the process *electroporation* is defined as:

```

Class: electroporation
  SubClassOf: 'cell permeabilization'
              and has_specified_input some cell
              and has_participant some 'power supply'

```

More generally, additional axioms may be used to relate members of the class to other entities in the ontology.

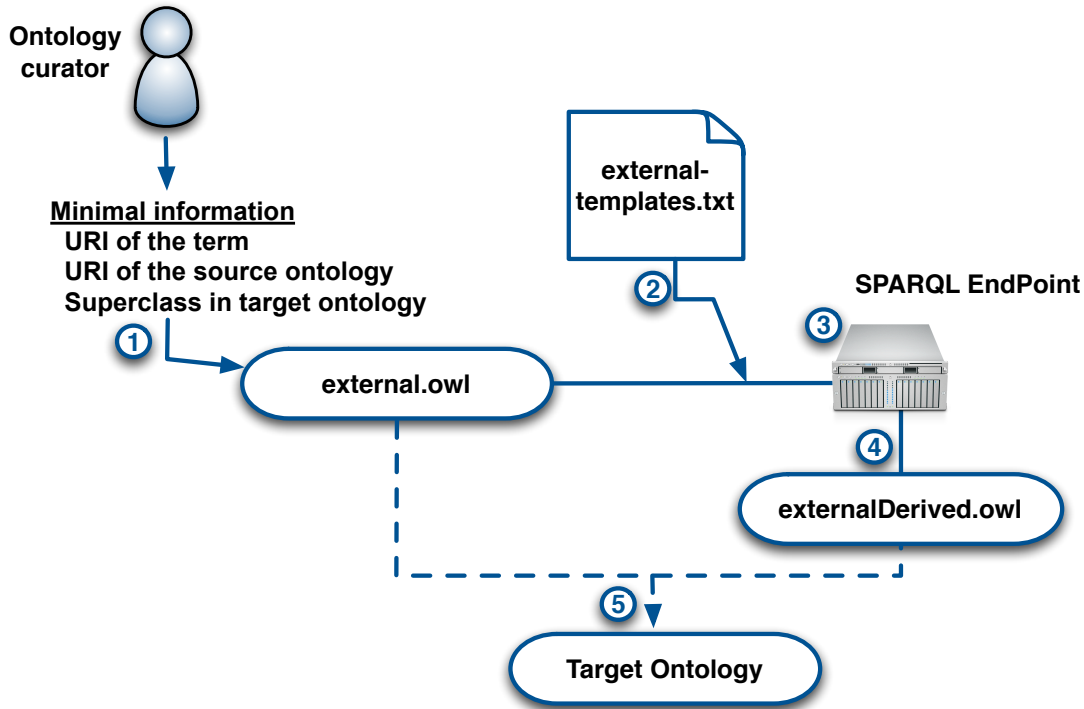


Fig. 2. Diagram of the MIREOT mechanism. 1. The ontology editor gathers the minimal information for the class to import and adds it into the external.owl file 2. A script parses the external.owl file, and for each class a SPARQL template is selected from external-templates.txt by matching the URI to patterns specified for each template. It uses the URI and the appropriate SPARQL template to generate a CONSTRUCT query 3. The SPARQL query is executed against a SPARQL endpoint (e.g., Neurocommons) 4. The results of the SPARQL queries are combined into the externalDerived.owl file 5. The target ontology imports the external.owl and externalDerived.owl files.

Use Case Two - taxonomic information

The `cell` use-case highlights what is likely to be the most common import scenario, *i.e.*, a simple import of one external term, making it available for direct use in the target ontology. However, in some cases, we may require more than that single external term, and to account for this MIREOT has been devised to be flexible.

Consider the scenario in which we have two experiments, one in human and one in mouse. The files are annotated with the classes `human` and `mouse` from the ontology, which are in turn mapped from the NCBI taxonomy. We can easily imagine that somebody would want to have a query of the form “give me all experiments in mammals”. In this case, we would need to know that `human` and `mouse` are subclasses (even indirect) of `mammals` in the NCBI taxonomy. Therefore, when mapping towards an NCBI term, we decided to retrieve all its superclasses as well up to the root of the NCBI taxonomy.

When the `create-external-derived.lisp` script parses the `external.owl` file and encounters an NCBI taxonomy ID, it will therefore invoke a specific SPARQL query (cf figure 3).

As per the mechanism described above, the minimum information about the imported external class (e.g., *Mus musculus*) is defined in `external.owl`, whereas the additional information (rank information - genus, kingdom, phylum, etc.- *i.e.*, its superclasses) is stored in `externalDerived.owl`. On the same model, any information that the importing ontology editors would require could be added in the `externalDerived.owl` file: the only requirement is to write the corresponding SPARQL query.

```

# give names to the top taxa
alias:bacteria=tax:_2
alias:eukaryota=tax:_2759
alias:archaea=tax:_2157
alias:viruses=tax:_10239
alias:cellularOrganism=tax:_131567

prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix owl: <http://www.w3.org/2002/07/owl#>
prefix obi: <http://purl.obofoundry.org/obo/>
prefix tax: <http://purl.org/obo/owl/NCBITaxon#NCBITaxon>

construct
{
  ?super rdfs:type owl:Class.
  ?super rdfs:subClassOf ?parent.
  ?super alias:preferredTerm ?label.
  ?super rdfs:label ?label.
  ?super alias:importedFrom <http://purl.org/obo/owl/NCBITaxon>
}
where
{
  {
    # We harvest the transitive superclass annotations
    _ID_GOES_HERE_ rdfs:subClassOf ?super.
    graph <http://purl.org/science/graph/obo/NCBITaxon>
    {
      ?super rdfs:subClassOf ?parent.
      ?super rdfs:label ?label.
    }
  }
}
UNION

{
  graph <http://purl.org/science/graph/obo/NCBITaxon>
  {
    ?super rdfs:subClassOf ?parent.
    ?super rdfs:label ?label.
    FILTER (?super=_ID_GOES_HERE_)
  }
}

FILTER (!((?super=alias:bacteria) || (?super=alias:eukaryota) || (?super=alias:viruses) || (?super=alias:archaea)
|| (?super = alias:cellularOrganism) || (?parent = alias:cellularOrganism)))

```

Fig. 3. Template SPARQL query for import from the NCBI taxonomy. The `_ID_GOES_HERE_` pattern will be replaced by the relevant NCBITax ID dynamically via script when building the CONSTRUCT query. This query allows retrieval of the class of interest and its parents up to a set of defined root classes. Note that the graph `<http://purl.org/science/graph/obo/NCBITaxon>` contains the source ontology, but the full store includes inferred `subClassOf` triples.

Discussion

The MIREOT mechanism offers a lightweight mechanism for importing specific classes from external ontologies. The approach is decoupled from the importing ontology, allowing a computational update mechanism which does not interfere with the primary ontology under development.

MIREOT is currently implemented and used by several ontology efforts, including OBI, the Information Artifact Ontology (IAO), the Vaccine Ontology (VO) (VO website), the Infectious Disease Ontology (IDO) (IDO website) and the Influenza Ontology (InfluenzO) (InfluenzO website). In the context of OBI, 472 terms are currently explicitly imported, which in turn leads to actual integration of 1447 classes (due to the automatic retrieval of parents when using the NCBI taxonomy).

With broader use of the MIREOT mechanism by OBI and other resources, several minor issues arose. For example, consider the case of IAO developers requiring import of the term *investigation*. This class already exists in OBI, and IAO developers therefore chose to MIREOT it, effectively integrating the class `http://purl.obolibrary.org/obo/OBI_0000066` and distributing it as part of the IAO releases. However, OBI imports IAO, and therefore re imports, *via* IAO, its own *investigation* class. This is not problematic in general, redundancy of information in OWL files being of no consequence. However when OBI curators decided to update the definition of the *investigation* class, the information natively in OBI and that imported from IAO became out-of-sync: two different definitions were displayed to the curators. Moreover one of them could not be edited as it is outside the remit of OBI to edit IAO definitions.

One solution to this problem would be to update the IAO import - but this requires a release of OBI with the updated *investigation* definition, its upload on Neurocommons, and for the IAO developers to update

their information and produce a new release of IAO. At best, this implies a delay of a few days, more realistically of a few weeks until the information in both files is again synchronized. A better solution would be for tools to recognize and prioritize the origin of a class based on its URI. Ontology editing tools would display only the information originating from the target ontology when editing the target ontology file. Such a solution also has consequences; when updating the information from Neurocommons, a specification of which Resource Description Framework (RDF) graph (RDF/XML Syntax Specification) the term *originally* belonged to is required. Taking again the example of the *investigation* class, when querying based on its URI without specifying the RDF graph, the SPARQL endpoint would return the OBI class, but also the one distributed by IAO, which is not the desired behavior; in our example, the IAO annotation property values are now out of date compared to the original and authoritative OBI file.

Finally, when updating MIREOTed information, the SPARQL endpoint where the information resides must be up-to-date. The implementation currently relies on the OBO Foundry resources at the Neurocommons OBO distribution. This is updated nightly with the latest information from the OBO server, and can therefore be reasonably relied upon for accessing current resources. The timeliness of the information may not always be known if extending the mechanism to another SPARQL endpoint, or other sets of ontological resources.

The MIREOT standard presents an approach to importing classes from external ontologies that removes the overhead of full ontology imports whilst maintaining a decoupled but usable reference to the external classes. There is a clear trade-off that MIREOT offers between practicality and full, axiomatic completeness. By importing only the desired parts of an external ontology there is the risk that inferences drawn may be incomplete or incorrect; correct inference using the external classes is only guaranteed if the full ontology, or a module, is imported. It does present one important advantage which is that it overcomes the obstacle presented by ontologies which are not fully interoperable at present.

Since only partial, reasoner-supported consistency checking is undertaken, extra care is taken when we need to make assertions about an imported term.

In adding axioms, such as the subclass axiom when importing the external term, the aim is to only assert true statements. In our experience with the better OBO ontologies, the denotation of the term, as explained in the definition or documentation, is clearer and more correct than the axiomatization. If additional restrictions are required those should be stored in the importing ontology: the *external.owl* and *externalDerived.owl* are meant to include only the imported information. It is anticipated that some of the statements added by the importing ontology may migrate to the source ontologies at some point in the future; a fruit of the collaborative nature of OBO Foundry ontology development.

When deciding to import an external term the textual definition is reviewed and, if required, discussion with the original editor is undertaken. As we are importing from OBO Foundry candidate ontologies we have a community process for monitoring change, a shared understanding of the basics of our domain, and the intention to eventually share the same upper-level ontology. Therefore, it is expected that terms will be deprecated if there is a significant change in meaning, and are flexible enough to adjust and update our import of terms as the other ontologies start enhancing their logical definitions.

Future work

The current implementation of the MIREOT guidelines relies on command-line scripts, making it difficult for some curators to use. A web service, OntoFox (OntoFox webserver, 2009), has been developed at the University of Michigan to facilitate the process: ontology editors can use web forms to input their requirements, or submit specific OntoFox-formatted files for batch creation or update. A useful extension to this work would be in the form of a Protégé plugin to improve the interaction between the curators and the tool. NCBO developers have created a widget allowing the insertion of external references in an ontology (BioPortal Reference Plugin), and it is our hope that a future extension would fully support the MIREOT guidelines. It is also expected that an option in the OBI distribution would enable the replacement of *external.owl* with *imports.owl*, a file of imports statements generated by extracting the ontology

URIs mentioned in *external.owl*. Users would then be able to import all of the external resources, therefore replacing the MIREOT selected terms. As module extraction technology matures, we intend to include the ability to use such mechanisms for doing targeted imports, on a source-by-source basis.

Acknowledgments

In memory of our friend and colleague William Bug.

The OBI consortium is (in alphabetical order): Ryan Brinkman, Bill Bug, Helen Causton, Kevin Clancy, Christian Cocos, Mélanie Courtot, Dirk Derom, Eric Deutsch, Liju Fan, Dawn Field, Jennifer Fostel, Gilberto Fragoso, Frank Gibson, Tanya Gray, Jason Greenbaum, Pierre Grenon, Jeff Grethe, Yongqun He, Mervi Heiskanen, Tina Hernandez-Boussard, Philip Lord, Allyson Lister, James Malone, Elisabetta Manduchi, Luisa Montecchi, Norman Morrison, Chris Mungall, Helen Parkinson, Bjoern Peters, Matthew Pocock, Philippe Rocca-Serra, Daniel Rubin, Alan Ruttenberg, Susanna-Assunta Sansone, Richard Scheuermann, Daniel Schober, Barry Smith, Larisa Soldatova, Holger Stenzhorn, Chris Stoeckert, Chris Taylor, Jessica Turner, John Westbrook, Joe White, Trish Whetzel, Stefan Wiemann, Jie Zheng. The author's work is partially supported by funding from the NIH(R01EB005034), the Public Health Agency of Canada / Canadian Institutes of Health Research Influenza Research Network (PCIRN), the EC EMER-ALD project (LSHG-CT-2006-037686), the BBSRC(BB/C008200/1, BB/D524283/1, BB/E025080/1), the EU FP7 DebugIT project (ICT-2007.5.2-217139), and the Michael Smith Foundation for Health Research.

References

- Bard J., Rhee S. Y., and Ashburner M. An ontology for cell types. *Genome biology*, 6(2):R21, 2005.
- BioPortal Reference Plugin. Retrieved December 2, 2009, from http://protegewiki.stanford.edu/index.php/BioPortal_Reference_Plugin.
- Degtyarenko, K., de Matos, P., Ennis, M., Hastings, J., Zbinden, M., McNaught, A., Alcantara, R., Darsow, M., Guedj, M. and Ashburner, M. (2008) ChEBI: a database and ontology for chemical entities of biological interest. *Nucleic Acids Res.* 36, D344-D350.
- Gene Ontology Consortium. The gene ontology (go) database and informatics resource, *Nucleic acids research* 32(90001), D258-D261.
- Go editorial style guide. Retrieved December 2, 2009, from <http://www.geneontology.org/GO.usage.shtml>.
- Golbreich C., Zhang S., and Bodenreider O.. The foundational model of anatomy in owl: Experience and perspectives. *Web semantics (Online)*, 4(3):181-195, 2006.
- Grau B.C., Horrocks I., Kazakov Y., and Sattler U. Extracting Modules from Ontologies: A Logic-based Approach. *Proc. of the Third OWL Experiences and Directions Workshop*, number 258 in CEUR
- Grau B.C., Horrocks I., Kazakov Y. and Sattler U. Just the right amount: Extracting modules from ontologies. *In proc. of the 16th International World Wide Web Conference (WWW 2007)*
- Grenon P., Smith B., and Goldberg L.. Biodynamic ontology: applying bfo in the biomedical domain. *Studies in health technology and informatics*, 102:20-38, 2004.
- Haendel M.A., Neuhaus F., Osumi-Sutherland D., M. Mabee P., Mejino Jr. J.L.V., Mungall C.J., Smith B.. CARO - The Common Anatomy Reference Ontology, in *Anatomy Ontologies for Bioinformatics Principles and Practice, Series: Computational Biology*, Vol. 6, ed. Burger, Albert; Davidson, Duncan; Baldock, Richard
- Jimenez-Ruiz E., Grau B.C., Sattler U., Schneider T. and Berlanga R. Safe and Economic Re-Use of Ontologies: A Logic-Based Methodology and Tool Support. *5th European Semantic Web Conference (ESWC 2008)*
- Neurocommons OBO SPARQL endpoint. Retrieved December 2, 2009, from <http://sparql.obo.neurocommons.org/>.
- OBI Ontology. Retrieved December 2, 2009, from <http://purl.obolibrary.org/obo/obi>.
- OBI scripts. Retrieved December 2, 2009, from <http://purl.obolibrary.org/obo/obi/repository/>.
- OntoFox. Retrieved December 2, 2009, from <http://ontofox.hegroup.org/>.
- Phenotypic Quality Ontology. Retrieved December 2, 2009, from http://obofoundry.org/wiki/index.php/PATO:Main_Page
- Seidenberg J., Rector A. Web ontology segmentation: analysis, classification and use. *In proc. of the 15th International World Wide Web Conference (WWW 2006)*
- Sirin, E., Parsia, B., Grau, B. C., Kalyanpur, A., and Katz, Y. (2007). Pellet: A practical OWL-DL reasoner. *Web Semant.* 5, 51-53.
- Smith B., Ashburner M., Rosse C., Bard J., Bug W., Ceusters W., Goldberg L. J., Eilbeck K., Ireland A., Mungall C. J., OBI Consortium, Leontis N., Rocca-Serra P., Ruttenberg A., Sansone S. A., Scheuermann R. H., Shah N., Whetzel P. L., and Lewis S. The obo foundry: coordinated evolution of ontologies to support biomedical data integration. *Nature biotechnology*, 25(11):1251-1255, Nov 2007.

- Smith, M.K.; Welty C., McGuinness D.L., OWL Web Ontology Language Guide. Retrieved December 2, 2009, from <http://www.w3.org/2007/OWL/>.
- SPARQL queries template file. Retrieved December 2, 2009, from <http://obi.svn.sourceforge.net/svnroot/obi/trunk/src/tools/build/external-templates.txt>
- SPARQL Query Language for RDF. Retrieved December 2, 2009, from <http://www.w3.org/TR/rdf-sparql-query/>.
- RDF/XML Syntax Specification. Retrieved December 2, 2009, from <http://www.w3.org/TR/rdf-syntax-grammar/>
- Ruttenberg A, Rees J, Samwald M, Marshall M. Life sciences on the Semantic Web: the Neurocommons and beyond *Briefings in Bioinformatics* 10 , 193-204 2009
- The Infectious Disease Ontology. Retrieved December 2, 2009, from <http://www.infectiousdiseaseontology.org/>
- The Influenza Ontology. Retrieved December 2, 2009, from <https://sourceforge.net/projects/influenzo/>.
- The Information Artifact Ontology (IAO). Retrieved December 2, 2009, from <http://code.google.com/p/information-artifact-ontology/>.
- The Protégé Ontology Editor and Knowledge Acquisition System. Retrieved December 2, 2009, from <http://protege.stanford.edu/>
- The Vaccine Ontology. Retrieved December 2, 2009, from <http://www.violinet.org/vaccineontology/>
- Wheeler D. L. , Barrett T., Benson D. A. , Bryant S. H., Canese K., Church D. M., DiCuccio M., Edgar R., Federhen S., Helmberg W., Kenton D. L., Khovayko O., Lipman D. J., Madden T. L., Maglott D. R., Ostell J., Pontius J. U., Pruitt K. D., Schuler G. D., Schriml L. M., Sequeira E., Sherry S. T., Sirotkin K., Starchenko G., Suzek T. O., Tatusov R., Tatusova T. A., Wagner L., and Yaschenko E.. Database resources of the national center for biotechnology information. *Nucleic acids research*, 33(Database issue), D39-45.