

Chapter 8: Basic Formal Ontology at Work (DRAFT: This chapter has evolved and is different in the book.)

In this chapter, after brief descriptions of the basic ideas associated with an ontology-building tool called Protégé and the Web Ontology Language (OWL), we provide a few examples of domain ontologies that utilize BFO as an upper-level ontology, complete with Protégé screenshots. We do this in order to give the reader an opportunity to see how the principles and recommendations from the previous chapters have been applied to actual domains. Numerous ontologies, projects, institutions, and groups have applied the best practices described in this book when constructing and developing their domain ontologies, as table 8.1 shows.¹

Table 8.1: Ontologies, Projects, Institutions, and Groups Utilizing BFO

Ontologies Utilizing BFO

ACGT Master Ontology	Alzheimer Disease Ontology	Adverse Event Ontology
Adverse Event Reporting Ontology	AFO Foundational Ontology	Actionable Intelligence Retrieval System
Bank Ontology	Beta Cell Genomics Application Ontology	BioAssay Ontology
Bioinformatics Web Service Ontology	Biological Collections Ontology	Biomedical Ethics Ontology
Biomedical Grid Terminology	BioTop	BIRNLex
Blood Ontology	Body Fluids Ontology	Cancer Cell Ontology
Cancer Chemoprevention Ontology	Cardiovascular Disease Ontology	Cell Behavior Ontology
Cell Cycle Ontology	Cell Expression, Localization, Development and Anatomy Ontology	Cell Line Ontology
Cell Ontology	Chemical Entities of Biological Interest	CHRONIOUS Ontology Suite
Clusters of Orthologous Groups Analysis Ontology	Cognitive Paradigm Ontology	Common Anatomy Reference Ontology
Communication Standards Ontology	Conceptual Model Ontology	Coriell Cell Line Ontology
CPR Ontology	Document Act Ontology	Drug Interaction Ontology
Drug Ontology	Drug-drug Interaction Ontology	Earth Sciences Ontologies
Eagle-I Research Resource Ontology	Email Ontology	Emotion Ontology
Environment Ontology	Epidemiology Ontology	Evolution Ontology
Experimental Factor Ontology	ExposA©	Financial Report Ontology
Flybase Drosophila Ontology	Fission Yeast Phenotype Ontology	Foundational Model of Anatomy
Gastrointestinal Ontology	Gene Regulation Ontology	General Information Model
Health Data Ontology Trunk	Human Interaction Network Ontology	Human Physiology Simulation Ontology
Infectious Disease Ontology	Information Artifact Ontology	Interdisciplinary Prostate Ontology Project
Lipid Ontology	Mental Disease Ontology	Mental Functioning Ontology
Middle Layer Ontology for Clinical Care	Military Ontology	MIRO and IRbase
Model for Clinical Information	Nanoparticle Ontology, Ontology for Cancer Nanotechnology Research	NeuroPsychological Testing Ontology
Neuroscience Information Framework	Neuroscience Information Framework Standard	Neural Electromagnetic Ontologies

Ocular Disease Ontology	OntoAlign++	Ontologized Information - Biobank
Ontology of Clinical Research	Ontology for Biomedical Investigations	Ontology for Drug Discovery Investigations
Ontology for General Medical Science	Ontology for Guiding Appropriate Antibiotic Prescribing	Ontology for Newborn Screening and Translational Research
Ontology for Mental Health and Quality of Life	Ontology of Biobanking Administration	Ontology for Parasite LifeCycle
Ontology for Rehabilitation (Traumatic Brain Injury)	Ontology of Data Mining	Ontology of Medically Related Social Entities
Ontology of Vaccine Adverse Events	Ontology-Based eXtensible Data Model	Oral Health and Disease Ontology
Parasite Experiment Ontology	Petrochemical Ontology	Phenotypic Quality Ontology
Plant Ontology	Population and Community Ontology	Proper Name Ontology
Protein-Ligand Interaction Ontology	Proteomics Data and Process Provenance Ontology	Protein Ontology
RNA Ontology	Saliva Ontology	Semantic HER
Senselab Ontology	Sequence Ontology	Sleep Domain Ontology
SemanticScience Integrated Ontology	Spatiotemporal Ontology for the Administrative Units of Switzerland	Special Nuclear Materials Detection Ontology
Subcellular Anatomy Ontology	Time Event Ontology	Translational Medicine Ontology
Universal Core Semantic Layer	Vaccine Ontology	Xenopus Anatomy Ontology
YAMATO	yOWL	Zebrafish Anatomical Ontology

Projects, Institutions, and Groups Utilizing BFO

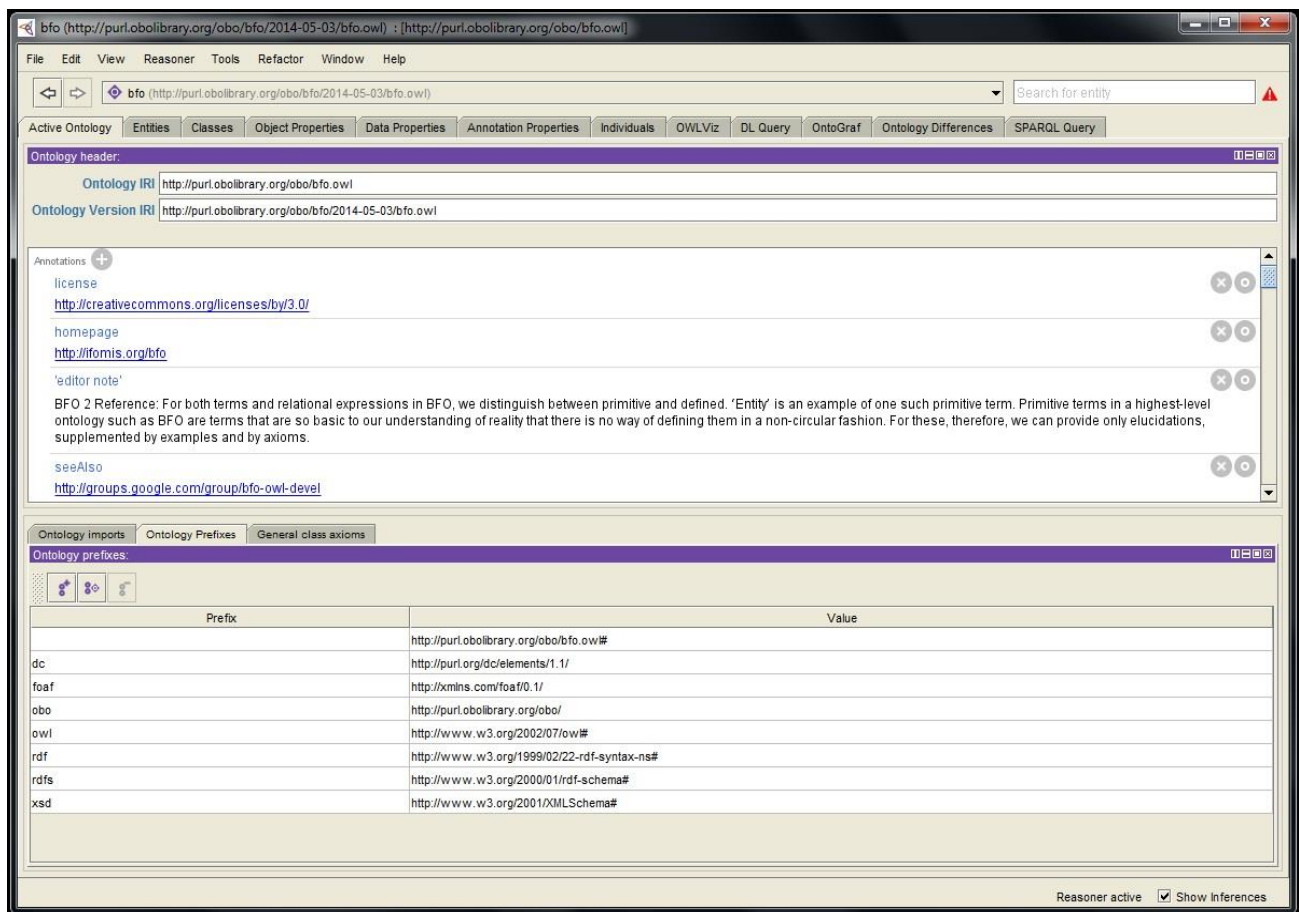
AstraZeneca - Clinical Information Science	Berkeley Bioinformatics Open-Source Projects	Biomedical Knowledge Engineering Lab at Seoul National University
Brain Operation Database	CTSAconnect	Darwin Core
Data-Tactics Corporation	DOQS: Data-Oriented Quality Solutions	DSpace at NTNUA
Dumontier Lab	eagle-i Consortium	Elsevier Smart Content Strategy
EuPathDB	GoodOD	HIGHFLEET
Influenza Research Database	INRIA Lorraine Research Unit	Kobe University, Graduate School of Medicine, Department of Sociomedical Informatics
Language and Computing	Medical University Graz - Informatics, Statistics and Documentation	National Center for Multi-Source Information Fusion
Ontology Research Group	Qualitative Spatiotemporal Unit	Referent Tracking Unit
OntoCAT	OntoCOG	OntoMaton
openEHR	Open PHACTS	Praxis
REMINE	Saitama University - Faculty of Liberal Arts	Science Commons - Neurocommons
Skeletome	University Hospital Erlangen – Radiology	University of Arkansas, Biomedical Informatics Medical Center
University of Augsburg, Computer Science, Software Methodologies for Systems	University of South Alabama School of Computing, South Biomedical Informatics Group	University of Texas Southwestern Medical Center
University of Washington - Structural Informatics Group	Virus Pathogen Resource	VIVO

1. Protégé Ontology Editor and BFO

In this book, our focus lies in giving researchers the theoretical semantic tools—rather than the actual computational algorithmic or implementational tools—for the construction of domain ontologies with the help of BFO. That said, at present there exists an ontology-building tool

known as Protégé that generates the Web Ontology Language (OWL), which is one technique that has been deployed for classification, search, and analytics concerning data and information (more will be said about OWL below).²

Figure 8.1: BFO in Protégé



As noted on the website's homepage (<http://protege.stanford.edu/>), Protégé is a “free, open-source ontology editor and framework for building intelligent systems.” This software can be downloaded quickly onto one's computer, and contains tutorials and other information relevant to the navigation of the various tabs, so that one can start composing a domain ontology with the help of BFO that will contain information that is maximally interoperable with other domain ontologies.³

The latest OWL for BFO may always be found at BFO's website (<http://www.ifomis.org/bfo/>); simply import the OWL for BFO into Protégé, and start populating the subtypes of the BFO categories and relations. At various points in this chapter, we will feature screenshots of ontologies that have been produced using Protégé (version 4.3) such as the one in figure 8.1, which is the homepage for BFO rendered in Protégé.

Figure 8.2: BFO Continuants and Occurrents in Protégé

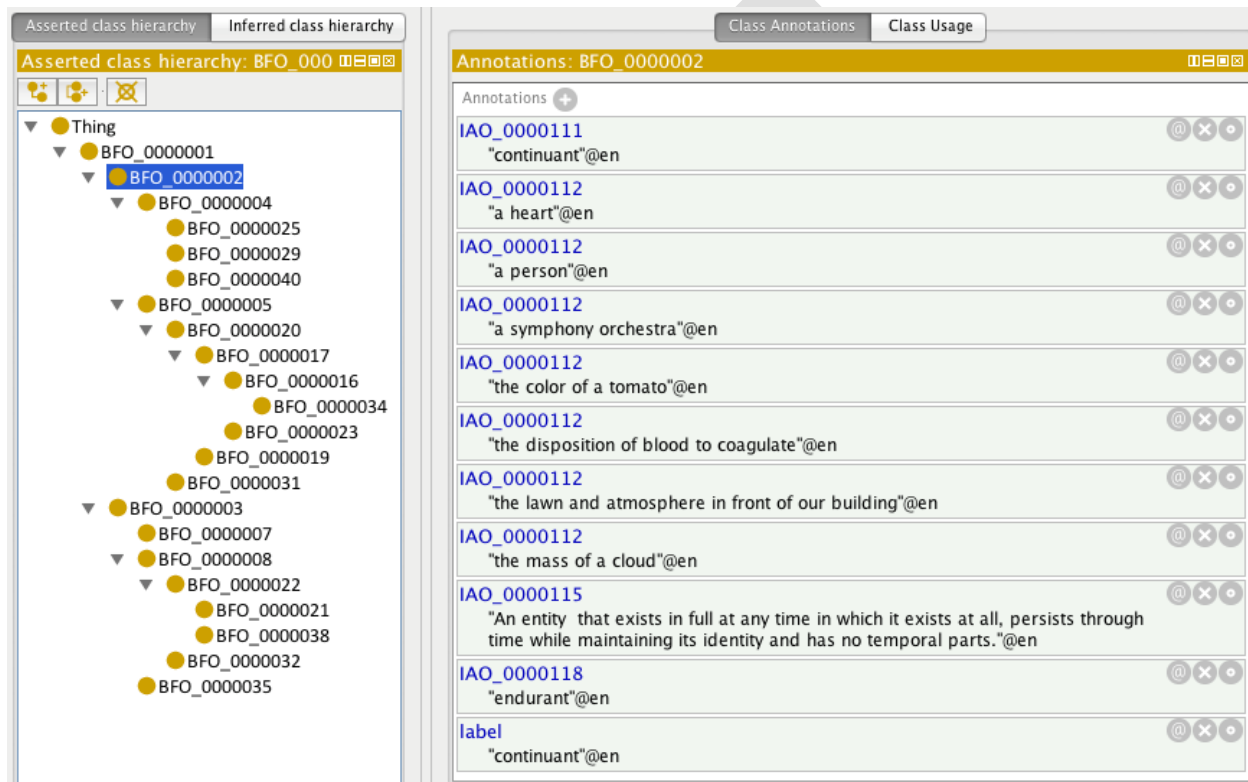
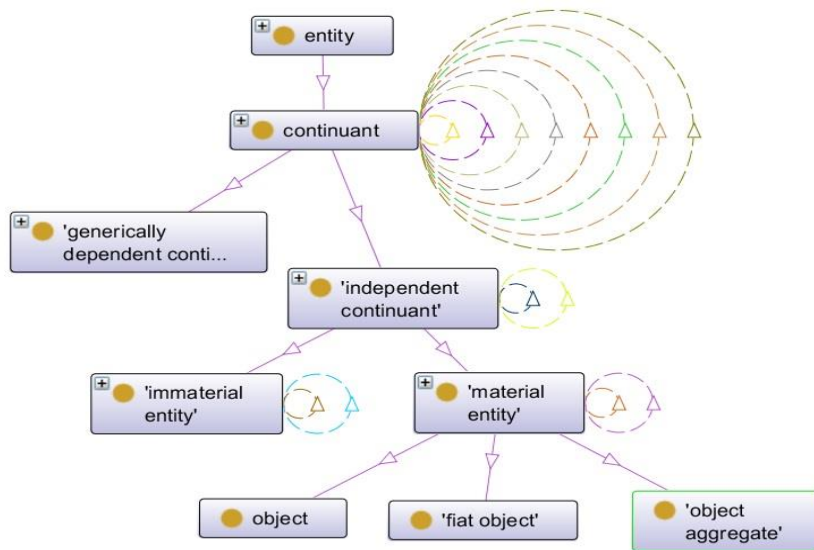


Figure 8.2 contains a screenshot from Protégé of BFO entities that is consistent with our recommendation from the fourth chapter that the terms in an ontology should be associated with a unique alphanumeric identifier (UAI) as well as a unique human-readable definition. For example, BFO_0000002 is the UAI for BFO's continuant, and it is highlighted in the left panel, while the name 'continuant' along with its definition can be found near the bottom of the right panel. Notice in the screenshot in figure 8.1 that there is a tab called OntoGraf that is the third

one from the right at the top of the page. This feature in Protégé enables one to produce a visual representation of the ontology's categories and relationships, as can be seen in figure 8.3, which highlights the BFO term 'object aggregate' and its parents in BFO's taxonomy.

Figure 8.3: BFO's 'Object Aggregate' Displayed in OntoGraf



2. The Web Ontology Language (OWL)

Protégé is a tool that enables one to produce ontologies in the Web Ontology Language (OWL). OWL evolved from two sources that were combined in the early 2000s: the US Defense Advanced Research Projects Agency (DARPA) Agent Markup Language (DAML) and the European Union's Ontology Inference Layer (OIL).⁴ By 2002 the DAML+OIL extension had been approved for use by the World Wide Web Consortium (W3C), which is the main standards organization for the Web (<http://www.w3.org>). The committee approving DAML+OIL was comprised of now-well-known working ontologists such as Ian Horrocks, Peter Patel-Schneider, Jim Hendler, Deb McGuinness, and the person credited with inventing the Web itself, Tim Berners-Lee. In December of 2003 OWL was proposed as a recommendation by the W3C with the following abstract, which still holds true today:

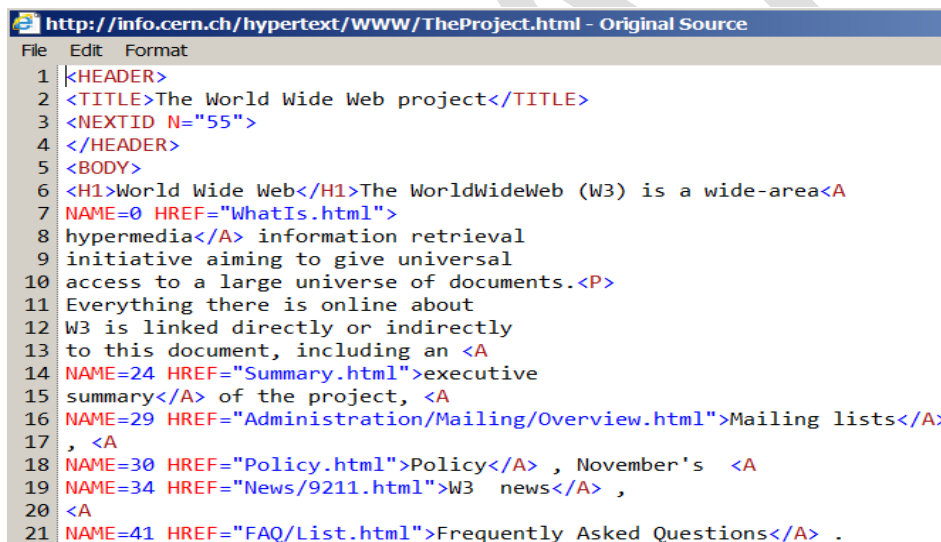
The OWL Web Ontology Language is designed for use by applications that need to process the content of information instead of just presenting information to humans. OWL facilitates greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDF-S) by providing additional vocabulary along with a formal semantics. OWL has three increasingly expressive sublanguages: OWL Lite, OWL DL, and OWL Full.⁵

In the next several sections, we will parse the main ideas in this abstract.

2.1 Hypertext Markup Language (HTML) and Extensible Markup Language (XML)

Hypertext Markup Language (HTML) was developed by Tim Berners-Lee in the late 1980s and was designed to allow Web developers to display information in a way that is accessible to humans for viewing via Web browsers. Figure 8.4 shows the several lines of the HTML behind the first Web page that Berners-Lee built and published on line on August 6, 1991.⁶

Figure 8.4: Part of the HTML Behind the First Web Page

A screenshot of a web browser window showing the source code of a web page. The browser's address bar displays the URL: http://info.cern.ch/hypertext/WWW/TheProject.html - Original Source. The browser's menu bar includes File, Edit, and Format. The source code is displayed in a monospaced font with line numbers from 1 to 21. The code includes HTML tags for a header, title, body, and a list of links. The links are: 'WhatIs.html', 'Summary.html', 'Administration/Mailing/Overview.html', 'Policy.html', 'News/9211.html', and 'FAQ/List.html'.

```
1 <HEADER>
2 <TITLE>The World Wide Web project</TITLE>
3 <NEXTID N="55">
4 </HEADER>
5 <BODY>
6 <H1>World Wide Web</H1>The WorldWideWeb (W3) is a wide-area<A
7 NAME=0 HREF="WhatIs.html">
8 hypermedia</A> information retrieval
9 initiative aiming to give universal
10 access to a large universe of documents.<P>
11 Everything there is online about
12 W3 is linked directly or indirectly
13 to this document, including an <A
14 NAME=24 HREF="Summary.html">executive
15 summary</A> of the project, <A
16 NAME=29 HREF="Administration/Mailing/Overview.html">Mailing lists</A>
17 , <A
18 NAME=30 HREF="Policy.html">Policy</A> , November's <A
19 NAME=34 HREF="News/9211.html">W3 news</A> ,
20 <A
21 NAME=41 HREF="FAQ/List.html">Frequently Asked Questions</A> .
```

However, HTML's primary limitation is that it does not provide much capability to describe information in ways that facilitate the use of software programs to find or interpret it. Thus, in 1998 the W3C recommended using the Extensible Markup Language (XML) that allows information to be more accurately described utilizing tags that are machine-readable, machine-interpretable, *as well as* human-interpretable. Put simply in the words of the members of W3C's

XML Working Group: “XML is the universal format for structured documents and data on the Web. It allows you to define your own mark-up formats.”⁷ Figure 8.5 contains two screenshots: the left one is the first few lines of the HTML behind CNN’s website homepage; the right one is XML related to a book catalogue. Notice that the tags in the XML—<author>, <title>, <genre>, etc.—are more human-readable and understandable than the HTML with its <meta http-equiv> kinds of tags.

Figure 8.5: Examples of HTML and XML

```
<!DOCTYPE HTML>
<html lang="en-US">
<head>
<title>CNN.com - Breaking News, U.S., World, Weather, Entertainment
&amp; Video News</title>
<meta http-equiv="content-type" content="text/html;charset=utf-8"/>
<meta http-equiv="last-modified" content="2014-07-31T00:05:26Z"/>
<meta http-equiv="refresh"
content="1800;url=http://www.cnn.com/?refresh=1"/>
<meta http-equiv="X-UA-Compatible" content="IE=edge"/>
<meta name="robots" content="index, follow"/>
<meta name="googlebot" content="noarchive"/>
<meta name="description" content="CNN.com delivers the latest
breaking news and information on the latest top stories, weather,
business, entertainment, politics, and more. For in-depth coverage,
CNN.com provides special reports, video, audio, photo galleries, and
interactive guides."/>
<meta name="keywords" content="CNN,CNN news,CNN.com,CNN
TV,news,news online,breaking news,U.S. news,world
news,weather,business,CNN
Money,sports,politics,law,technology,entertainment,education,travel,
health,special reports,autos,developing story,news video,CNN Intl"/>
<meta name="application-name" content="CNN"/>
<meta name="msapplication-tooltip" content="Breaking News, U.S.,
World, Weather, Entertainment and Video News"/>
```

```
<?xml version="1.0"?>
<catalog>
  <book id="b1">
    <author>Arp, Robert</author>
    <title>Scenario Visualization</title>
    <genre>Philosophy of Biology</genre>
    <price>44.95</price>
    <publish_date>2008-10-01</publish_date>
    <description>An evolutionary account of vision-related
creative problem solving.</description>
  </book>
  <book id="b2">
    <author>Jones, Janet</author>
    <title>Rain at Dawn</title>
    <genre>Fantasy</genre>
    <price>9.95</price>
    <publish_date>2011-12-16</publish_date>
    <description>A sorcerer's apprentice named Rain learns
the ways of magic to battle a foe from her past.</description>
  </book>
```

2.2 Resource Description Framework (RDF)

XML has a limited capability to describe the relationships between and among entities, however, which prompted researchers to develop Resource Description Framework (RDF) on top of XML in the late 1990s. RDF became a W3C recommendation in February of 1999.⁸ RDF is a language based on the idea of modeling entities and relationships with a simple Subject-Predicate-Object format known as a *triple*. Because RDF mimics the structure of human language this way, it

allows for a computational model of entities and relationships that has greater expressivity and semanticity than can be found in XML alone.

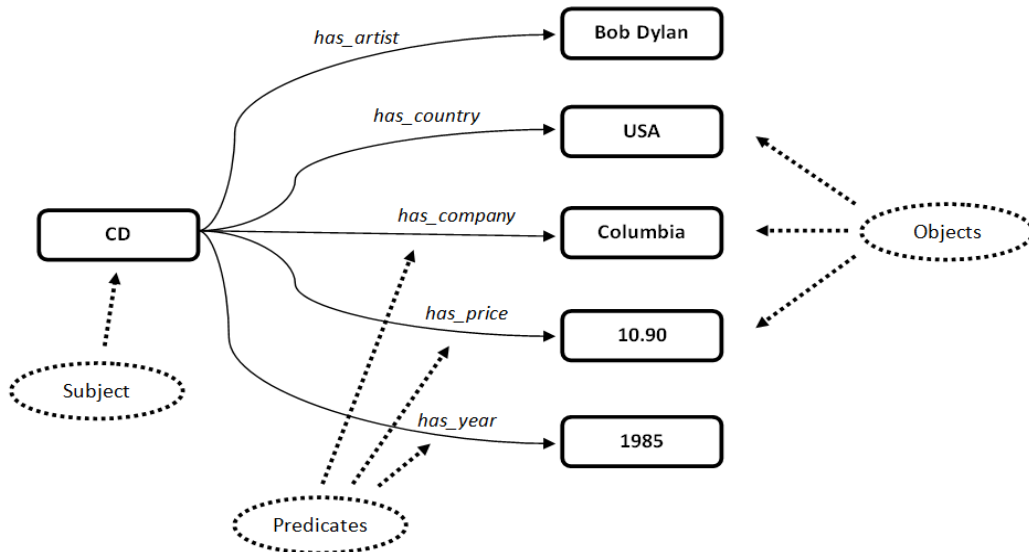
Table 8.2: Examples of Triples Associated with RDF

	Subject	Predicate	Object
English sentence:	“This CD...	has as its artist...	Bob Dylan.”
RDF rendering:	cd	Artist	Bob Dylan
English sentence:	“This CD...	was produced in the country...	USA.”
RDF rendering:	cd	Country	USA
English sentence:	“This CD...	was produced by the company...	Columbia.”
RDF rendering:	cd	Company	Columbia
English sentence:	“This CD...	has as its price...	10.90.”
RDF rendering:	cd	Price	10.90
English sentence:	“This CD...	has as its year of production...	1985.”
RDF rendering:	cd	Year	1985

Figure 8.6: Example RDF and Graph

```
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:cd="http://www.recshop.fake/cd#">

<rdf:Description
rdf:about="http://www.recshop.fake/cd/Empire_Burlesque">
  <cd:artist>Bob Dylan</cd:artist>
  <cd:country>USA</cd:country>
  <cd:company>Columbia</cd:company>
  <cd:price>10.90</cd:price>
  <cd:year>1985</cd:year>
</rdf:Description>
```



The *resource* part of RDF refers to the fact that the subject, predicate, and object of a triple each has its own Uniform Resource Identifier (URI) that can be utilized to locate it on the Web or in some database consisting of such URIs. Figure 8.6 shows some RDF and a graph associated with a description of Bob Dylan’s 23rd studio album, *Empire Burlesque*, released by Columbia Records in 1985. Here, the URI for the CD in this fake CD shop is `http://www.recshop.fake/cd#`, while the triples in table 8.2 can be found in the RDF.

A machine can be set up to read the RDF, and someone who works at the fake CD shop can query the database composed of triples—called a *triplestore*—and receive detailed information about Dylan’s recording. Further, because the subjects, predicates, and objects are tagged with URIs, this information can be placed on the Web so that anyone in the world could query the fake CD shop’s inventory and database to learn about Dylan’s recording. A few standard examples of RDF include:

- `rdf:subject`, referring to the subject of the RDF statement
- `rdf:predicate`, referring to the predicate of the RDF statement
- `rdf:object`, referring to the object of the RDF statement
- `rdf:type`, which is used to denote an instance of a class
- `rdf:property`, referring to the class of properties
- `rdf:XMLLiteral`, referring to the class of XML literal values

Notice also that the example RDF in figure 8.6 uses “`xmlns:...`” in the second line of code, which refers to the designation of an XML namespace followed by the standard syntax: `xmlns:prefix = “URI”` (in this case, `xmlns:rdf = “http://www.w3.org/1999...”`). RDF is built on XML—in fact, often one sees and hears researchers talking about “the RDF/XML code” working behind the scenes of a certain ontology—and this is just one example of XML acting as a subset language for RDF.⁹

2.3 Simple Protocol and RDF Query Language (SPARQL)

Just as tuples are queried with Structured (or Standard) Query Language (SQL) in a standard relational database, so too, triples are queried in a triplestore database with Simple Protocol and RDF Query Language (SPARQL). In early 2008 SPARQL 1.0 became an official W3C recommendation, while SPARQL 1.1 became a W3C recommendation in March of 2013.¹⁰

A simple SPARQL query asking for all of the predicates in a triplestore database is formed like this:

```
SELECT DISTINCT ?predicate
WHERE { ?subject ?predicate ?object }
ORDER BY ?predicate
```

Looking again at the RDF in figure 8.5 (reprinted below), we could write several SPARQL queries about Bob Dylan's album, *Empire Burlesque*. For example, a SPARQL query about the company that produced the album looks like this:

```
SELECT ?company
WHERE { <http://www.recshop.fake/cd/Empire Burlesque> ?company. }
```

And the result of the query would be: *Columbia*. Replacing ?company in the query with ?country would result in *USA*, while ?price would result in *10.90*, ?year would result in *1985*, and ?artist would result in *Bob Dylan*.

```
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:cd="http://www.recshop.fake/cd#">

<rdf:Description
rdf:about="http://www.recshop.fake/cd/Empire Burlesque">
  <cd:artist>Bob Dylan</cd:artist>
  <cd:country>USA</cd:country>
  <cd:company>Columbia</cd:company>
  <cd:price>10.90</cd:price>
  <cd:year>1985</cd:year>
</rdf:Description>
```

2.4 RDF Schema (RDF-S)

Immediately after RDF began to be used in the early 1990s, RDF Schema (RDF-S) was developed as a way to share RDF vocabulary as well as structure RDF resources (again, URIs) so that they may be saved in a triplestore database and queried with SPARQL.¹¹ A simple SPARQL query asking for all of the subclasses of a class in a triplestore database is formed like this:

```
SELECT ?subject ?object
WHERE { ?subject rdfs:subClassOf ?object }
```

Additionally, researchers began enriching the RDF by adding vocabulary and associated semantics for classes, subclasses, properties, sub-properties, and instances/individuals/members.

A few standard examples of RDF-S include:

- `rdfs:label`, which refers to a string of text describing the resource
- `rdfs:comment`, referring to a potentially longer comment about the resource, also often used as a place to put a definition or description
- `rdfs:seeAlso`, denoting links to other relevant resources
- `rdfs:literal`, referring to something that is a primitive data type
- `rdfs:subClassOf`, whereby the statement “X `rdfs:subClassOf` Y” means that (1) all instances of class X are also an instance of class Y and (2) all the properties of Y are also properties of X
- `rdfs:domain`, which refers to the set of all things to which some property applies
- `rdfs:range`, which refers to the set of values that the property can accept

By adding elements such as `rdfs:subClassOf`, `rdfs:domain`, `rdfs:range`, and others, RDF-S was able to augment the expressivity of RDF.

Figure 8.7: Transitivity Written Into RDF-S

```
<rdfs:Class rdf:ID="cat">
  <rdfs:subClassOf rdf:resource=
    "http://fabricated.org/gen#mammal">
</rdfs:Class>

<rdfs:Class rdf:ID="mammal">
  <rdfs:subClassOf rdf:resource=
    "http://fabricated.org/gen#animal">
</rdfs:Class>

<rdfs:Class rdf:ID="cat">
  <rdfs:subClassOf rdf:resource=
    "http://fabricated.org/gen#animal">
</rdfs:Class>
```

However, almost immediately after RDF-S was developed researchers began to see that it was still too weak to describe entities and relationships in sufficient detail. Further, given that many of these same researchers had been doing work in artificial intelligence, they wanted a language that could enable machine reasoning.¹² While RDF-S allows for the `subClassOf` relationship, it does not enable transitivity, for example. Once it is stated in RDF-S that “X `rdfs:subClassOf` Y” and “Y `rdfs:subClassOf` Z” there is no way for RDF-S to generate the obvious statement that “X `rdfs:subClassOf` Z” other than by writing it into the code, as has been done in figure 8.7.

Besides the transitive property of relationships, RDF-S is unable to account for inverse, functional, symmetric, reflexive, and other basic properties of relationships that we mentioned in the previous chapter. It is impossible to say in RDF-S, for example, that *part_of* is the inverse of *has_part* or that *adjacent_to* is symmetrical.

Further, although RDF-S added the ability to specify the domain and range of entities, there is no ability to constrain or localize the domain or the range. For example, it is not possible to designate that the range of *has_offspring* is person when applied to persons, but then is dog when applied to dogs, cat when applied to cats, aardvarks when applied to aardvarks, and so on. Also, there are no existence or cardinality constraints in RDF-S. One cannot say, for example, that all instances of person have a mother that is also a person, or that persons have exactly two parents.¹³

2.5 Basic Features of OWL

These problems with RDF-S—and others—prompted researchers to develop OWL, and many of the same researchers working on the W3C recommendations for DAML+OIL also worked on the recommendations for OWL that were published in the early to mid 2000s. A key feature of OWL

is its basis in Description Logic (DL)—specifically, *SHOIN* and *SHOIN(D)*—which emerged out of the artificial intelligence community in the mid-1980s and has a formal semantics based on mathematical set theory and first-order logic with its use of quantified variables.¹⁴ Thus, in contradistinction to XML, RDF, and RDF-S, OWL allows for:

- a. Universal (\forall) quantification, called the `owl:allValuesFrom` restriction in OWL.
- b. Existential (\exists) quantification, utilizing the `owl:someValuesFrom` or `owl:hasValue` restrictions in OWL.
- c. Cardinality through `owl:cardinality`, `owl:minCardinality`, and `owl:maxCardinality`.
- d. The Booleans *and*, *or*, and *not*, which in OWL are called `owl:intersectionOf`, `owl:unionOf`, and `owl:complementOf`.
- e. Properties to be added to relationships, to include inverse (`owl:inverseOf`), functional (`owl:FunctionalProperty`), inverse functional (`owl:InverseFunctionalProperty`), transitive (`owl:TransitiveProperty`), symmetric (`owl:SymmetricProperty`), asymmetric (`owl:AsymmetricProperty`), reflexive (`owl:ReflexiveProperty`), and irreflexive (`owl:IrreflexiveProperty`).
- f. The inference that “*X is_a Z*” from “*X is_a Y*” and “*Y is_a Z*” for entities and relationships through `owl:subClassOf` and `owl:subPropertyOf`, given that transitivity is a property of the *is_a* relationship.
- g. The specification of the equivalence of entities or relationships through `owl:equivalentClass` and `owl:equivalentProperty`, as well as various kinds of differences between and among entities or relationships through `owl:differentFrom`, `owl:AllDifferent`, and `owl:distinctMembers`.
- h. The specification of two types of properties: *object properties* and *datatype properties*. In conjunction with the designation of a domain and range, an object property (`owl:ObjectProperty`) specifies a relationship between two individuals (instances, members)—as in, “wing *part_of* airplane” or “wrist *adjacent_to* hand”—while a datatype property (`owl:DatatypeProperty`) specifies a relationship between an individual and a data type value (integer, double, float, string, boolean, etc.); datatype properties in OWL are often equivalent to the notion of *attributes* in some modeling languages, such as the Unified Modeling Language (UML).

In order to generate automated inferences with OWL on some computational platform—for example, the inference that “*X is_a Z*” from “*X is_a Y*” and “*Y is_a Z*”—one must use a *semantic reasoner* (also sometimes referred to as an *inference engine*), which is a piece of software that can infer logical consequences from a set of asserted facts or axioms. The reasoners usually paired with OWL are set up to utilize first-order predicate logic to execute the inferences, and perform standard DL checks of the ontology to: ensure that the ontology does not contain

any contradictory facts (consistency checking); make sure that the subclass relations between every named class are computed to create the complete class hierarchy (basic classification checking); determine whether it is possible for a class in the ontology to have any instances (concept satisfiability checking); and compute the direct classes for any instance in the ontology (realization checking).¹⁵ Protégé comes stock with a reasoner developed by Ian Horrocks called Fast Classification of Terminologies++ (FaCT++), but many ontologists also download reasoners such as Clark & Parsia's Pellet Reasoner (<http://clarkparsia.com/pellet/protege/>), Franz Inc.'s RacerPro Reasoner (<http://franz.com/agraph/racer/>), and Information Systems Group's Hermit OWL Reasoner (<http://hermit-reasoner.com/>) as plug-ins for Protégé.

2.6 OWL Lite, OWL DL, and OWL Full

OWL actually consists of three languages with increasing expressivity: OWL Lite, OWL DL, and OWL Full, and each is a syntactic extension of its simpler predecessor. In fact, it is true that:

- Every legal OWL Lite ontology is a legal OWL DL ontology
- Every legal OWL DL ontology is a legal OWL Full ontology
- Every valid OWL Lite conclusion is a valid OWL DL conclusion
- Every valid OWL DL conclusion is a valid OWL Full conclusion

Another way of saying the above is that OWL Lite is a simplified subset of OWL DL, and OWL DL is a subset of OWL Full. Also, all three types of OWL build on a restricted subset of RDF/XML and RDF-S terms; thus, it was necessary to describe XML, RDF, and RDF-S earlier in this chapter.¹⁶

All three of these languages allow one to describe classes, properties, and instances, but the weaker languages have restrictions on what can be stated as well as how it may be stated. OWL Lite has a subset of the capabilities mentioned in a. through h. a few paragraphs back, and it is intended for users with simple modeling needs who wish to build a basic *is_a* taxonomy with a few relations, and it only allows cardinality values of 0 or 1.

OWL DL seems to be the most widely used of the three OWL languages, and has all of the capabilities mentioned in a. through h. According to the W3C in their February 10, 2004 recommendation:

OWL DL supports those users who want the maximum expressiveness while retaining computational completeness (all conclusions are guaranteed to be computable) and decidability (all computations will finish in finite time). OWL DL includes all OWL language constructs, but they can be used only under certain restrictions (for example, while a class may be a subclass of many classes, a class cannot be an instance of another class). OWL DL is so named due to its correspondence with *description logics*, a field of research that has studied the logics that form the formal foundation of OWL.¹⁷

One important feature of OWL DL is that it is computationally decidable—when a reasoner is run over the OWL, it always terminates and returns a correct result. In other words, given its basis in DL, this OWL language has an effective method for determining a Boolean true or false value, as opposed to looping indefinitely. Decidability was a core design criterion in the development of OWL DL, and many of the syntactic restrictions that exist therein were introduced to retain decidability.

Both OWL DL and OWL Lite require that every entity in the ontology either be a class, object property, datatype property, or instance/individual; an entity cannot be treated as both a class and an instance. Also, each class must be made explicit in the ontology. OWL Full has the same features as OWL DL, but loosens these restrictions: in OWL Full an entity *can* be treated as both a class and an instance, and there is no need explicitly to declare the type of each class in the ontology.

Further, OWL Full is *undecidable*, so when a reasoner is run over it, it never returns results, looping indefinitely. Thus, while even greater modeling expressivity is had in OWL Full—one can engage in *metamodeling*, essentially, where RDF, RDF-S, and OWL constructs of entities and relationships can be augmented or redefined—the tradeoff is that the standard DL

checks we mentioned above (consistency checking, basic classification checking, concept satisfiability checking, and realization checking) are not possible. The authors of “A Practical Guide to Building OWL Ontologies Using the Protégé-OWL Plugin and CO-ODE Tools: Edition 1.0” note simply: “The choice between OWL DL and OWL Full may be based upon whether it is important to be able to carry out automated reasoning on the ontology or whether it is important to be able to use highly expressive and powerful modeling facilities such as meta-classes (classes of classes).”¹⁸

Figure 8.8: An Abridged Section of the OWL for the Infectious Disease Ontology

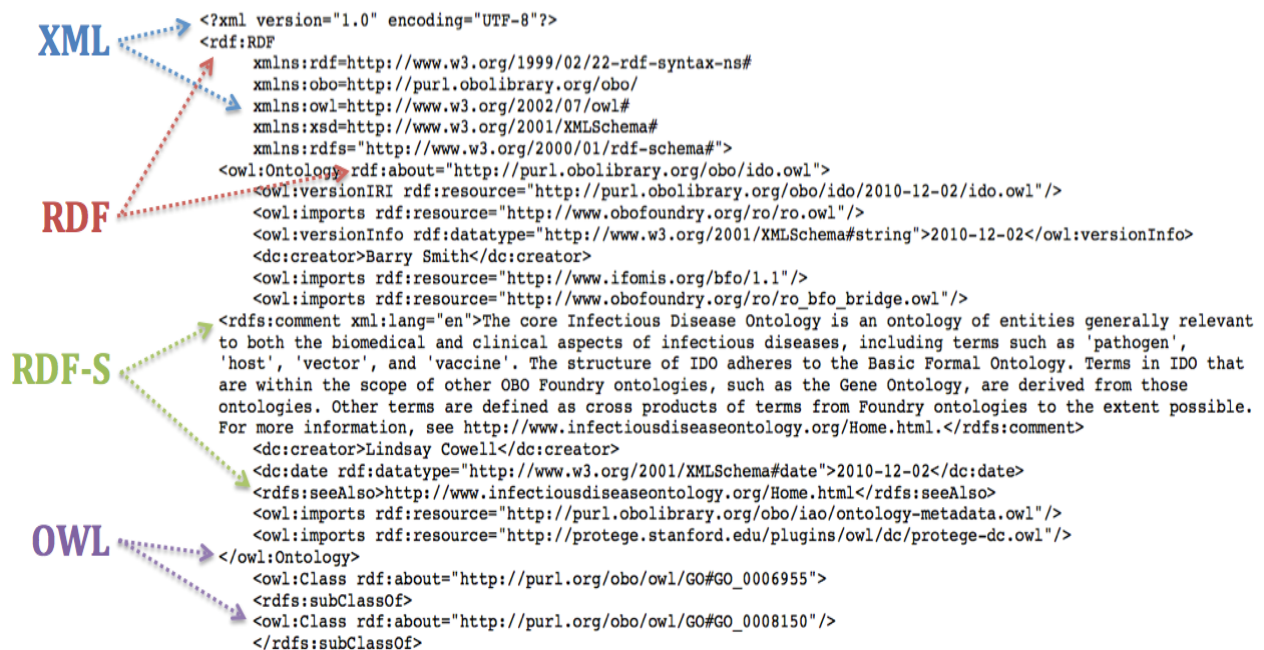


Figure 8.8 contains an abridged section of the OWL for the Infectious Disease Ontology, which is a BFO-based ontology of entities that are considered relevant to the biomedical and clinical aspects of infectious diseases. This domain ontology utilizes OWL DL, and the OWL, RDF-S, RDF, and XML can be seen in the code. Figure 8.9 shows the OWL associated with BFO’s ‘material entity,’ a subclass of ‘independent continuant.’

Figure 8.9: The OWL Associated with BFO's Material Entity

```
<!-- http://purl.obolibrary.org/obo/BFO_0000040 -->
<owl:Class rdf:about="http://purl.obolibrary.org/obo/BFO_0000040"><!-- material entity -->
  <rdfs:label xml:lang="en">material entity</rdfs:label>
  <rdfs:subClassOf rdf:resource="http://purl.obolibrary.org/obo/BFO_0000004"/><!-- independent continuant -->
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="http://purl.obolibrary.org/obo/BFO_0000176"/><!-- part of continuant at some time -->
      <owl:allValuesFrom rdf:resource="http://purl.obolibrary.org/obo/BFO_0000040"/><!-- material entity -->
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="http://purl.obolibrary.org/obo/BFO_0000083"/><!-- occupies spatial region at some time -->
      <owl:allValuesFrom rdf:resource="http://purl.obolibrary.org/obo/BFO_0000028"/><!-- three-dimensional spatial region -->
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="http://purl.obolibrary.org/obo/BFO_0000186"/><!-- part of continuant at all times that whole exists -->
      <owl:allValuesFrom rdf:resource="http://purl.obolibrary.org/obo/BFO_0000040"/><!-- material entity -->
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

2.7 OWL vs. Standard Relational Databases (RDBs)

There are three other points to make about ontologies built in OWL that can best be seen in contradistinction to a standard Relational DataBase (RDB). First, in an RDB each instance/individual must have a unique name; for example, the planet Venus must be referenced using one name only, such as ‘The Morning Star,’ or a grandmother must be referenced using only one name, such as ‘Grandma.’ In an OWL ontology, on the other hand, an instance can have more than one name; Venus can be referenced as either ‘The Morning Star’ or ‘The Evening Star,’ or both, and one’s grandmother can be referenced as ‘Grandma,’ ‘Granmommy,’ ‘Nana,’ all three, or any two of the three.

Second, the closed-world assumption (CWA) is usually (though, not necessarily) at work in an RDB whereby what is *not* known to be true in the database is always considered false, as if knowledge of the world were complete. In contradistinction, OWL ontologies utilize the open-world assumption (OWA) whereby what is not known to be true is always considered to be *unknown*—we are not sure if it is true or false—due to the fact that our knowledge of the world is always incomplete. The way that the CWA is practically made manifest in data modeling can

be shown with the following simple example: if Fido, Rover, dog, and “Fido is an instance of a dog” are specified in an RDB, and a SQL query asking, “Is Rover an instance of a dog?” is formulated, the answer to the query would be a definite *no*, given the CWA. The OWA answer, on the other hand, would be *unknown*—we are not sure if Rover is an instance of a dog or not. Also, if you queried, “How many dogs are there?” the RDB would come back with a definite answer of 1 and only 1, while the OWA answer would be at least 1, but possibly more.

Third, and most importantly, in an RDB the schema (with its attendant axioms and rules) exists simply to constrain and structure the data, and it is not possible to run a reasoner over the data. The relations in a database *themselves* do not (and cannot) have properties inherent in them such as transitivity or symmetry. Given its basis in DL, however, OWL’s axioms and rules—to include properties such as transitivity, symmetry, and the others we have already mentioned—have been designed to facilitate the generation of implications and inferences. For example, in OWL if you specify, “dog *is_pet_of* human” and create an individual named Rover as a member of the class of dogs and an individual named Jim as a member of the class of humans, the reasoner will infer that “Rover *is_pet_of* Jim.”¹⁹

2.8 OWL 2

Shortly after researchers started using OWL in the early 2000s, they began to realize its limitations. One basic and prevalent problem with OWL is its inability to handle qualified cardinality restrictions without workarounds that are unsound or incomplete. For example, while it is easily to say in OWL that someone has four dogs as pets, it cannot be stated easily that someone has four dogs as pets, two of which are male, or someone has four dogs as pets, one of which is a poodle and the other three are boxers. Another problem has to do with restrictions placed on data type values. For example, while it is possible to say that the barometric pressure

in a particular part of the atmosphere has a weight of 10 pounds, it is not possible to say that the weight is between 10 and 12 pounds.²⁰ These and several other problems prompted researchers to develop OWL 2, which became a W3C recommendation in October of 2009.

In their document describing new features of OWL 2 from December of 2012, members of the W3C note:

OWL 2 is a backwards-compatible revision to the Web Ontology Language (OWL). OWL 2 adds several new constructs to extend the expressivity of OWL including those for qualified cardinality restrictions, role chains, and expressive data predicates. OWL 2 also includes a new XML Serialization (targeted to the XML tool chain, i.e., XSLT, schema languages, etc.) and a set of subsetting profiles with various desirable application and computational properties.²¹

Thus, OWL 2 has addressed the two problems with OWL already mentioned, as well as many others having to do with expressivity.

OWL 2 also defines three profiles that are language subsets with useful computational properties and implementation capabilities: OWL 2 EL enables polynomial reasoning and is ideal for large-scale ontologies; OWL 2 RL enables polynomial reasoning using rule-based technologies that operate directly on RDF triples in triplestores; OWL 2 QL enables the querying of RDBs through query rewriting of SQL and other RDB-querying languages, a very useful feature for practical, working ontologists.

3. Ontology for General Medical Science (OGMS)

Having briefly described the basic features of Protégé and a few essential ideas associated with OWL, we are now in a better position look at a few examples of domain ontologies that not only utilize BFO as an upper-level ontology, but also have applied the principles and recommendations found in this book during ontology development.

An inspection of the homepage for the Ontology for General Medical Science (OGMS) reveals the influence of BFO and the recommendations from this book at work in the very

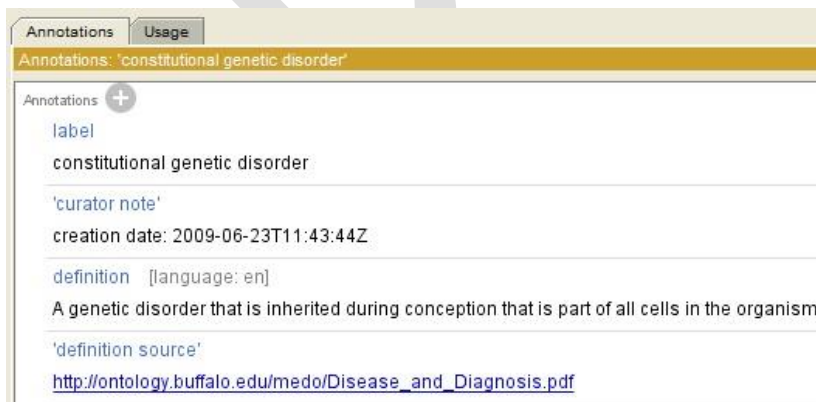
definition of the ontology, which utilizes the Aristotelian definitional structure of “An A is a B that Cs”:

The Ontology for General Medical Science (OGMS) is an ontology of entities involved in a clinical encounter. OGMS includes very general terms that are used across medical disciplines, including: ‘disease,’ ‘disorder,’ ‘disease course,’ ‘diagnosis,’ ‘patient,’ and ‘healthcare provider.’ OGMS uses Basic Formal Ontology (BFO) as an upper-level ontology. The scope of OGMS is restricted to humans, but many terms can be applied to a variety of organisms. OGMS provides a formal theory of disease that can be further elaborated by specific disease ontologies.²²

The domain of clinical medicine is a difficult one from an ontological perspective. Clinical terminology can be inconsistent, vague, and highly dependent on disciplinary context. OGMS is designed to provide a formal, explicit, non-redundant, and unambiguous representation of clinical terms that can begin to address these difficulties. OGMS is not a disease ontology; rather, it is a reference ontology that provides a general theory of disease and formal definitions for terms widely used in clinical encounters to describe different aspects of disease. It is being used as a framework from which to build ontology modules for a range of different diseases and disease families.

The OGMS utilizes the Aristotelian definitional structure for all of its entities, as in the example shown in figure 8.10 of constitutional genetic disorder.

Figure 8.10: OGMS’s Constitutional Genetic Disorder



The screenshot displays the 'Annotations' tab for the term 'constitutional genetic disorder'. The interface includes a search bar at the top with the text 'Annotations: constitutional genetic disorder'. Below this, there is a list of annotations for the term:

- label**: constitutional genetic disorder
- 'curator note'**: creation date: 2009-06-23T11:43:44Z
- definition [language: en]**: A genetic disorder that is inherited during conception that is part of all cells in the organism.
- 'definition source'**: http://ontology.buffalo.edu/medo/Disease_and_Diagnosis.pdf

Figure 8.11 shows a graph of ‘constitutional genetic disorder’ as a child of ‘genetic disorder,’ which itself is a child of ‘disorder.’ In addition to the definition found in the annotations, the unique alphanumeric identifier (UAI), OGMS_0000051, is found in the URI address. Figure 8.12 shows the OWL behind the designation of ‘constitutional genetic disorder’ as its own class. Notice the RDF-S in line five that reads:

```
<rdfs:subClassOf rdfs:resource="&obo;OGMS_0000047"/>.
```

OGMS_0000047 is the alphanumeric identifier for ‘genetic disorder,’ and so this line communicates: “‘constitutional genetic disorder’ is a sub-class (sub-type, kind) of ‘genetic disorder.’”

Figure 8.11: Graph of Constitutional Genetic Disorder with Alphanumeric Identifier

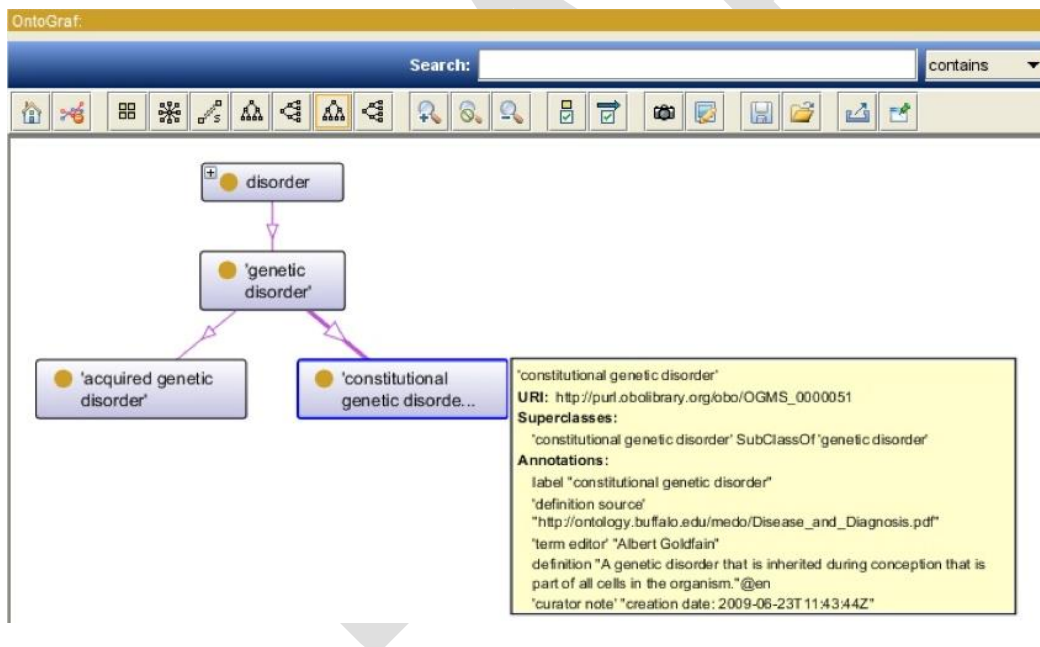


Figure 8.12: The OWL Behind the Categorization of Constitutional Genetic Disorder

```
<!-- http://purl.obolibrary.org/obo/OGMS_0000051 -->
<owl:Class rdf:about="&obo;OGMS_0000051">
  <rdfs:label
    >constitutional genetic disorder</rdfs:label>
  <rdfs:subClassOf rdfs:resource="&obo;OGMS_0000047"/>
  <obo:IAO_0000117>Albert Goldfain</obo:IAO_0000117>
  <obo:IAO_0000119
    >http://ontology.buffalo.edu/medo/Disease_and_Diagnosis.pdf</obo:IAO_0000119>
  <obo:IAO_0000232
    >creation date: 2009-06-23T11:43:44Z</obo:IAO_0000232>
  <obo:IAO_0000115 xml:lang="en"
    >A genetic disorder inherited during conception that is part of all cells in the organism.</obo:IAO_0000115>
</owl:Class>
```

Figure 8.13 shows how constitutional genetic disorder can ultimately be traced up BFO's taxonomy from 'material entity' to 'independent continuant,' which is a type of 'continuant.' OGMS also utilizes BFO's Relation Ontology, which was described in the previous chapter. Figure 8.14 places 'disorder' in the broader context of the core terms of the OGMS, BFO, and the relations that hold between the entities denoted by them. These core terms are those that are most generally used in clinical medicine—expressions that are also frequently conflated in natural language contexts.

Figure 8.13: Constitutional Genetic Disorder as a Type of Independent Continuant

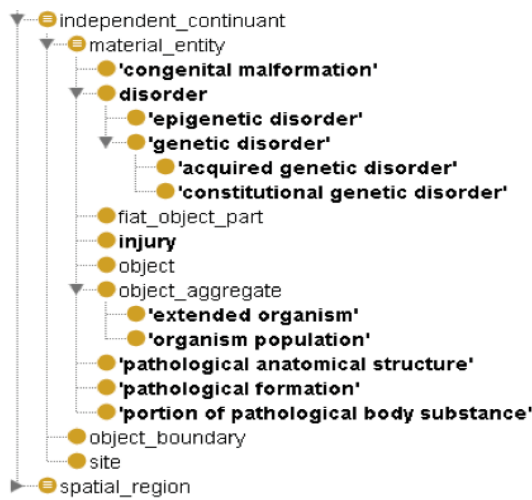
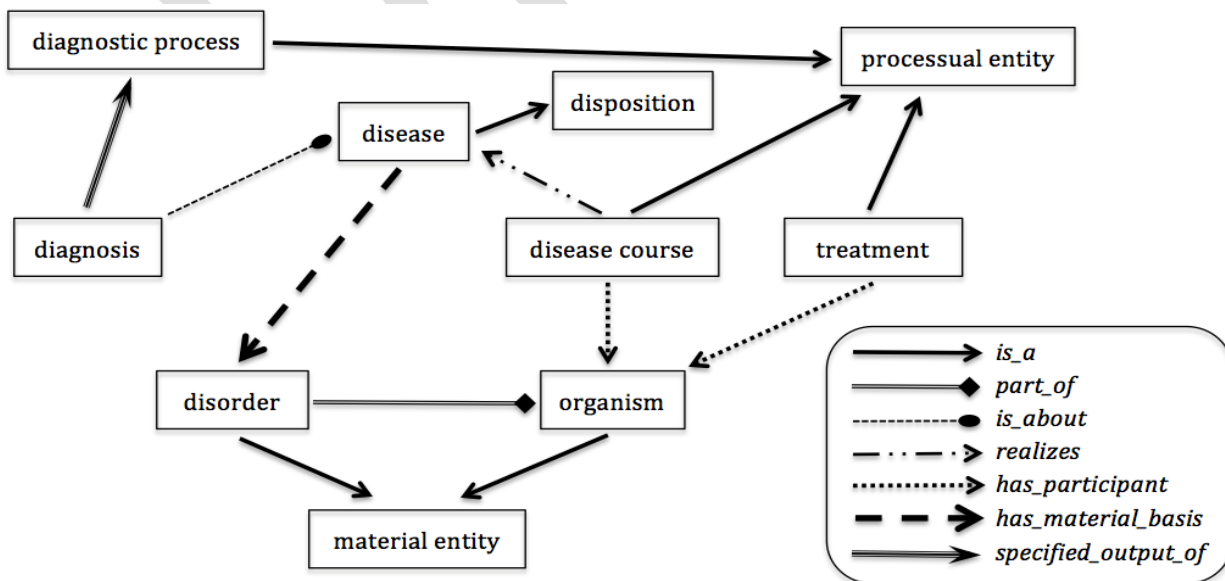
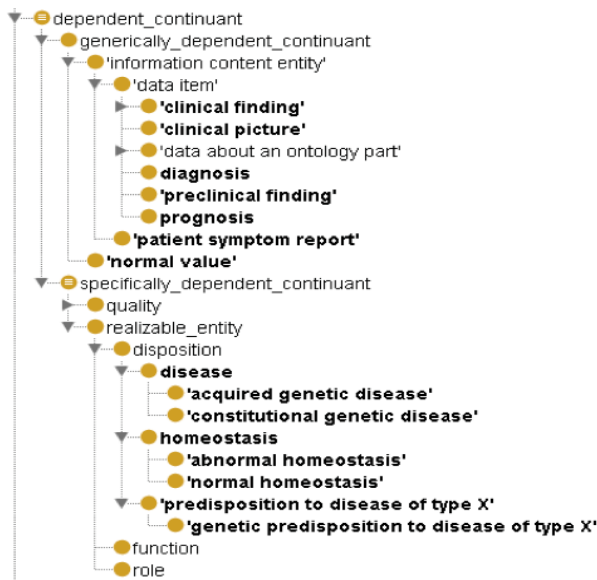


Figure 8.14: A Few Core Terms and Relations of the OGMS



The OGMS is built around a view of disease as a certain sort of power or potentiality—roughly, the potentiality for signs and symptoms to be manifested—which exists in organisms by virtue of physical disorders in the organism. These powers or potentialities are called ‘dispositions,’ and so the OGMS defends the general view according to which diseases are special types of ‘disposition,’ which themselves can be traced up the *is_a* hierarchy to BFO ‘dependent continuant,’ as figure 8.15 shows.

Figure 8.15: Disease as a Sub-type of Dependent Continuant



The OGMS has been designed to serve as the framework for describing how specific classes of physical disorders relate to abnormal dispositions manifesting themselves in pathological processes that are recognized as signs or symptoms in clinical encounters and documented in clinical information systems. Clinical application ontologies extend the OGMS by refining the basic taxonomic and relational structure for a particular domain of interest. Examples of OGMS extensions include the Infectious Disease Ontology (IDO), Sleep Domain Ontology (SDO), Ontology of Medically Relevant Social Entities (OMRSE), Vital Sign Ontology (VSO), Mental Disease Ontology (OPMQoL), Neurological Disease Ontology (ND),

Adverse Events Ontology (AEO), Ontology for Newborn Screening (ONSTR), Drug Ontology (DrOn), Model for Clinical Information (MCI), Ocular Disease Ontology (ODO), Oral Health and Disease Ontology, Mental Functioning Ontology, and Cardiovascular Disease Ontology.²³

4. Infectious Disease Ontology (IDO)

As we mentioned above, one of the OGMS's extensions is the Infectious Disease Ontology (IDO), which is an ontology of over 500 universal entities generally relevant to both the biomedical and clinical aspects of infectious diseases, as can be seen in the definition under the comment in the screenshot of IDO's homepage in Protégé in figure 8.16.

Figure 8.16: IDO's Homepage in Protégé

The screenshot displays the Protégé interface for the Infectious Disease Ontology (IDO). It is divided into several panels:

- Ontology Annotations:**
 - comment:** "The core Infectious Disease Ontology is an ontology of entities generally relevant to both the biomedical and clinical aspects of infectious diseases, including terms such as 'pathogen', 'host', 'vector', and 'vaccine'. The structure of IDO adheres to the Basic Formal Ontology. Terms in IDO that are within the scope of other OBO Foundry ontologies, such as the Gene Ontology, are derived from those ontologies. Other terms are defined as cross-products of terms from Foundry ontologies to the extent possible. For more information, see: <http://www.infectiousdiseaseontology.org/Home.html>."@en
 - creator:** "Barry Smith"
 - seeAlso:** "<http://www.infectiousdiseaseontology.org/Home.html>"
 - creator:** "Lindsay Cowell"
- Imported ontologies:**
 - 1.1 (<http://www.ifomis.org/bfo/1.1>)
 - ontology-metadata.owl (<http://purl.obolibrary.org/obo/iao/ontology-metadata.owl>)
 - protege-dc.owl (<http://protege.stanford.edu/plugins/owl/dc/protege-dc.owl>)
 - ro.owl (<http://www.obofoundry.org/ro/ro.owl>)
- Ontology metrics:**

Metrics	
Class count	509
Object property count	43
Data property count	0
Individual count	178
DL expressivity	SROIF

Class axioms	
SubClass axioms count	585
Equivalent classes axioms count	94
Disjoint classes axioms count	340
GCI count	0
Hidden GCI Count	94

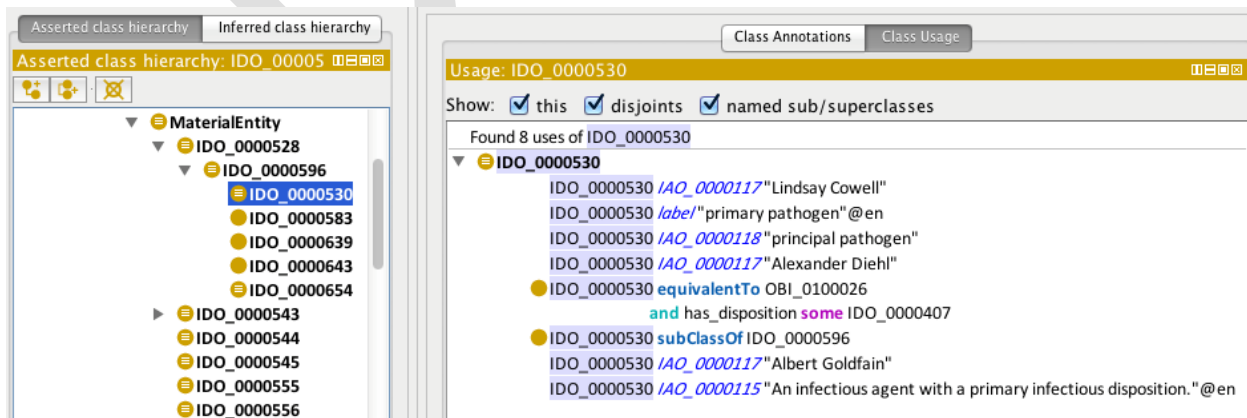
Object property axioms	
Sub object property axioms count	19
Equivalent object properties axioms count	0
Inverse object properties axioms count	12
Disjoint object properties axioms count	0
Functional object property axioms count	1
Inverse functional object property axioms count	0
Transitive object property axioms count	15
Symmetric object property axioms count	0
Anti-symmetric object property axioms count	0
Reflexive object property axioms count	0
Irreflexive object property axioms count	0
Object property domain axioms count	16
Object property range axioms count	20
Object property chain subproperty axioms count	3

IDO actually is a set of ontologies including a general IDO-Core along with sub-domain-specific ontologies developed as extensions of the neutral core. The extensions include IDO-Brucellosis,

IDO-Dengue Fever, IDO-HIV, IDO-Infective Endocarditis, IDO-Influenza, IDO-Malaria, IDO-Staphylococcus Aureus, IDO-Tuberculosis, and IDO-Vaccines.²⁴

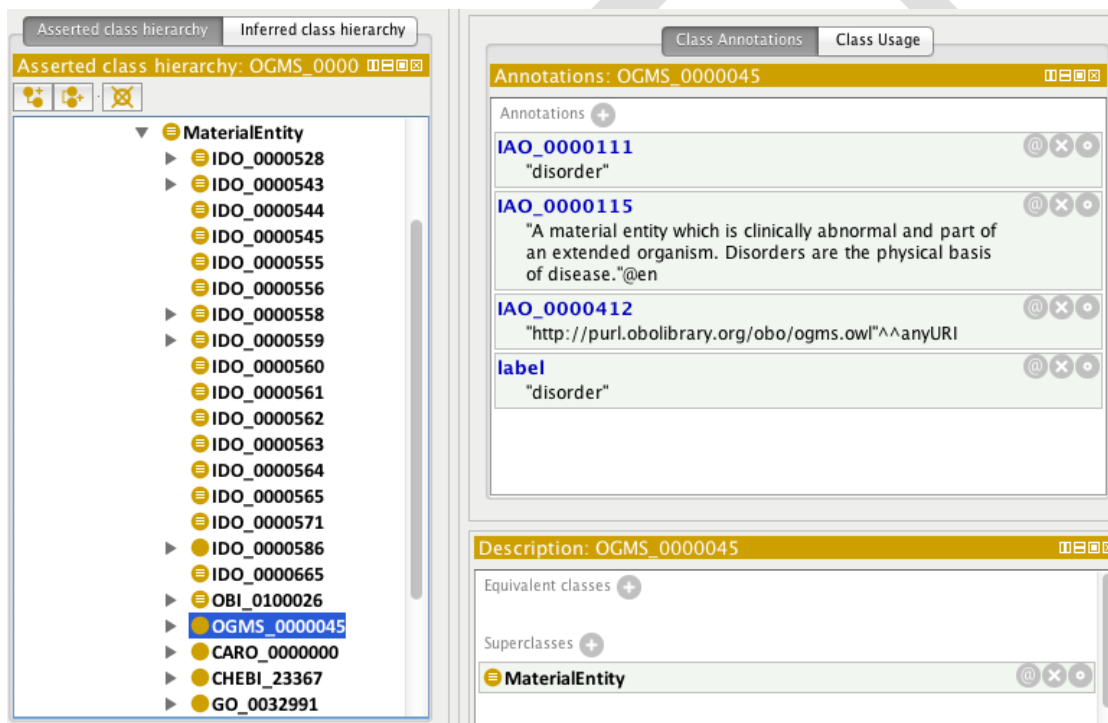
Figure 8.17 shows IDO’s ‘principal pathogen,’ an “infectious agent with a primary infectious disposition.” The circle next to the UAI (IDO_0000530) contains a tribar, which indicates that the term is equivalent to another term. Designating equivalency of terms is a primary way that ontologists map terms between and among various domain ontologies. The OBI prefix refers to the Ontology for Biomedical Investigations (OBI), an ontology for the description of biological and clinical investigations that utilizes BFO as its upper-level ontology as well (http://obi-ontology.org/page/Main_Page). Notice the logical statement that is five lines under IDO_0000530 in the right pane of the screenshot in figure 8.17: OBI_0100026 is the UAI for ‘organism,’ a material entity that is an individual living system—such as animal, plant, bacteria, or virus—that is capable of replicating or reproducing, growth, and maintenance in the right environment; IDO_0000407 is the UAI for ‘primary infectious disposition,’ an infectious disposition to become part of a disorder in organisms that have intact defenses. Thus, in more-understandable language this logical statement communicates, “‘principal pathogen’ is equivalent to (the exact same thing as) an organism that has a primary infectious disposition.”

Figure 8.17: A Mapping of IDO’s Principal Pathogen



It is good practice to re-use authoritative terms from one domain ontology in another domain ontology, and to map the terms as has been done in IDO with ‘principal pathogen.’ IDO re-uses several terms from other authoritative domain ontologies, and figure 8.18 highlights the OGMS’s ‘disorder’ (with the UAI of OGMS_0000045), while relevant UAIs from the Common Anatomy Reference Ontology (CARO_0000000, ‘anatomical entity’), the Chemical Entities of Biological Interest (CHEBI_23367, ‘molecular entity’), and the Gene Ontology (GO_0032991, ‘macromolecular complex’) are shown below it.

Figure 8.18: OGMS Reuses Other Authoritative Domain Ontology Terms



5. Information Artifact Ontology (IAO)

The Information Artifact Ontology (IAO) is an ontology of information entities based on BFO, emerging out of the OBI’s desire to categorize digital entities and realizable information entities correctly.²⁵ It concerns the material bearers of information (books, disks, photographic prints, traffic signs), information content entities themselves (ideas in a book, XML files on a disk,

charts and symbols on a photographic print, directions on a traffic sign), the processes that produce or consume information content entities (writing, documenting, encoding, drawing, client-server processing), and the relations between and among such entities (*is_about*, *denotes*, *is_rendering_of*, *is_translation_of*). A foundational notion in the IAO is that information content entities are related to other things by being “about” them or denoting them—they are types of BFO ‘generically dependent continuant.’ The ideas, tables, and figures being communicated right now in this book are examples of information content entities that denote other things such as BFO, OWL, RDF, ontologies, and others, while the hard copy of the book in your hand or the e-reader used to read or view the ideas, table, and figures are examples of the material bearers of the information content entities.

Figure 8.19: Documenting in IAO

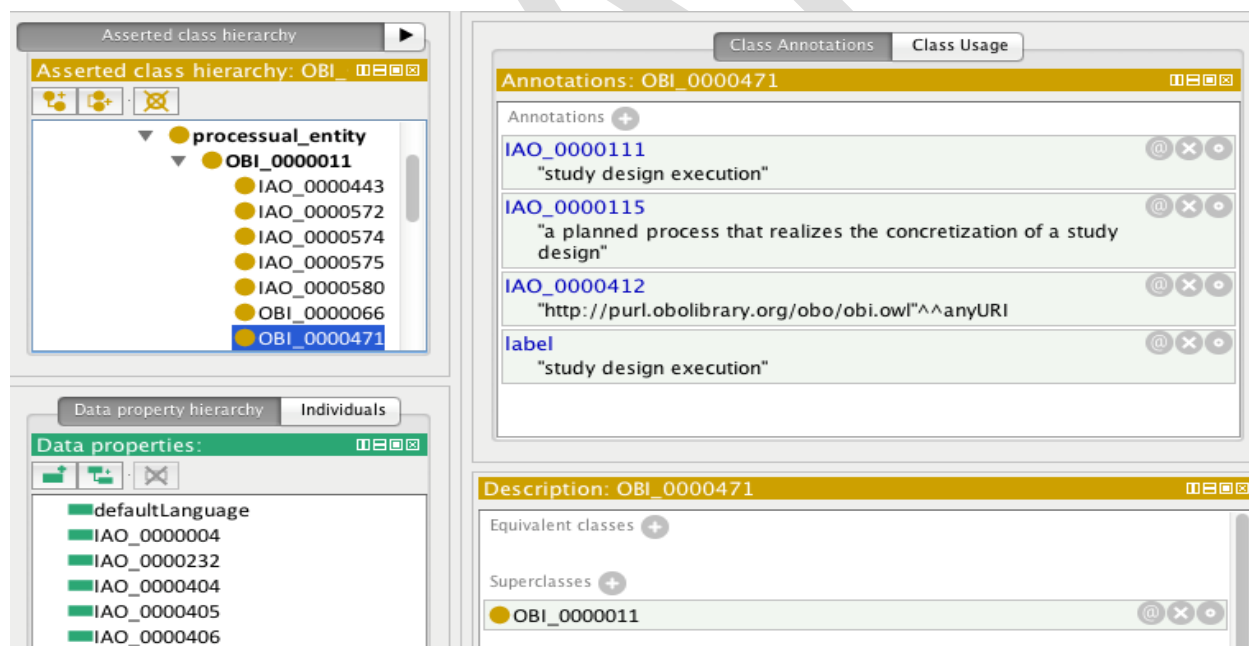
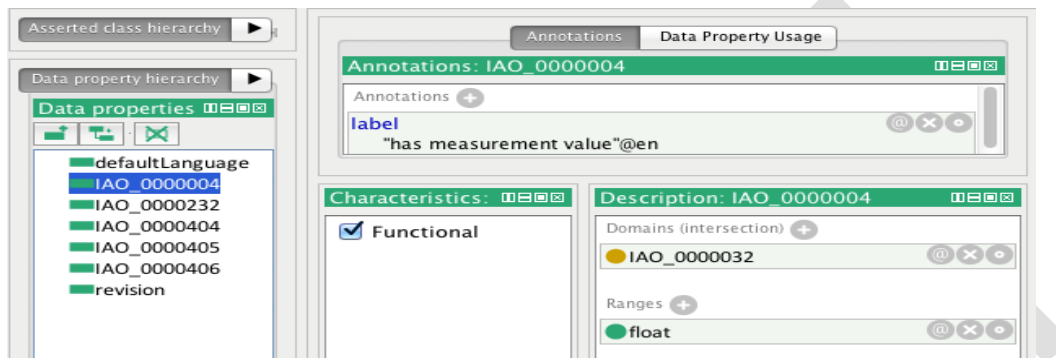


Figure 8.19 features the IAO’s ‘study design execution’ with its UAI, definition, and classification ultimately as a type of BFO ‘processual entity.’ Note in the screenshot that the IAO utilizes data properties, as is consistent with OWL’s best modeling practices. Figure 8.20 shows

one of IAO's data properties, *has_measurement_value*, with its domain being IAO_0000032 ('scalar measurement datum') and its range being a float (a floating-point number). This relation is functional, meaning that an individual scalar measurement datum has one, and only one, floating measurement value.

Figure 8.20: IAO's *has_measurement_value* Data Property



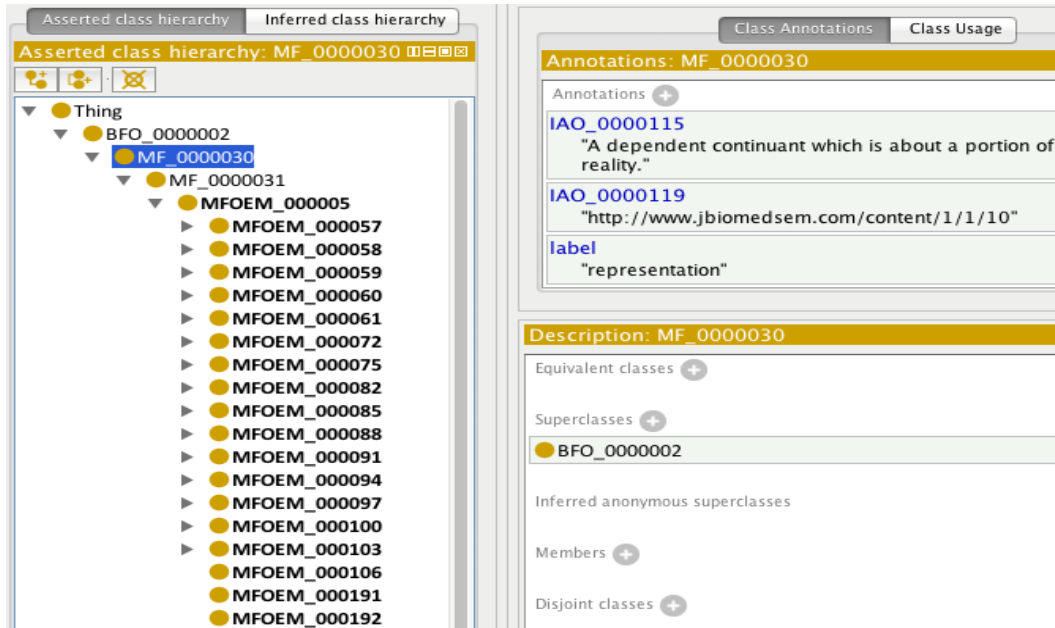
6. The Emotion Ontology (EO)

The Emotion Ontology (EO) is an ontology of over 850 universal entities associated with affective phenomena such as emotions, moods, appraisals, and subjective feelings.²⁶ Each aspect of the ontology is rooted in BFO; for example, BFO 'occurrent' is utilized in the EO with 'emotion occurrent' as a subtype that includes 'anger' and 'grief' as subtypes of it. Also, 'memory' is a type of 'mental disposition,' which falls under 'bodily disposition,' and 'bodily disposition' is a subtype of BFO's 'disposition.'

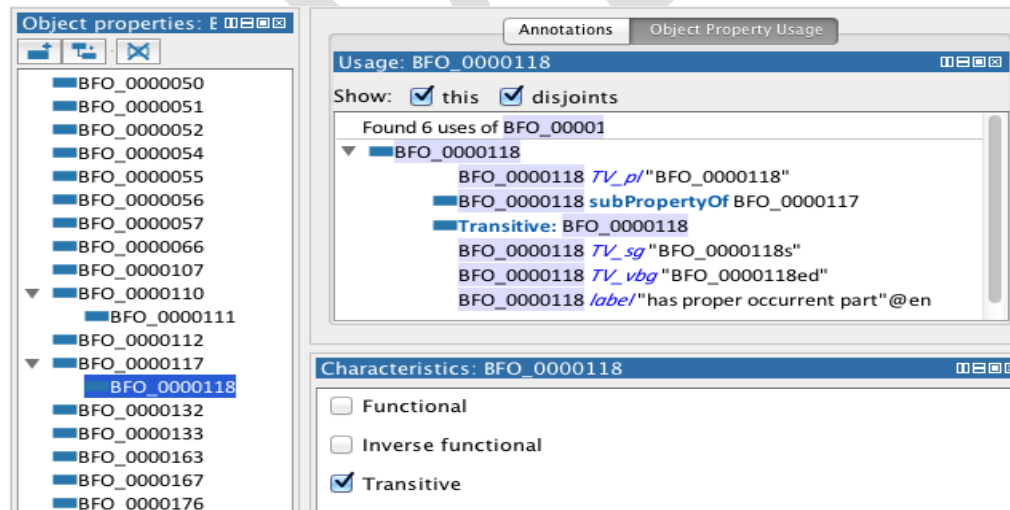
Besides being grounded in BFO, the EO is an extension of the Mental Functioning Ontology (MFO), an ontology for mental functioning, including mental processes, traits, and related mental diseases and disorders that is developed in the context of the OGMS and rooted in BFO.²⁷ Figure 8.21 underscores this by showing how numerous EO universal entities with UAI prefixes of MFOEM are subtypes of 'cognitive representation,' itself a child of MFO's

‘representation’ (highlighted in the figure with a UAI of MF_0000030), which is a child of ‘continuant’ (with a UAI of BFO_000002).

Figure 8.21: Several of the EO’s Universal Entities



8.22: Several of BFO’s Relations in the EO



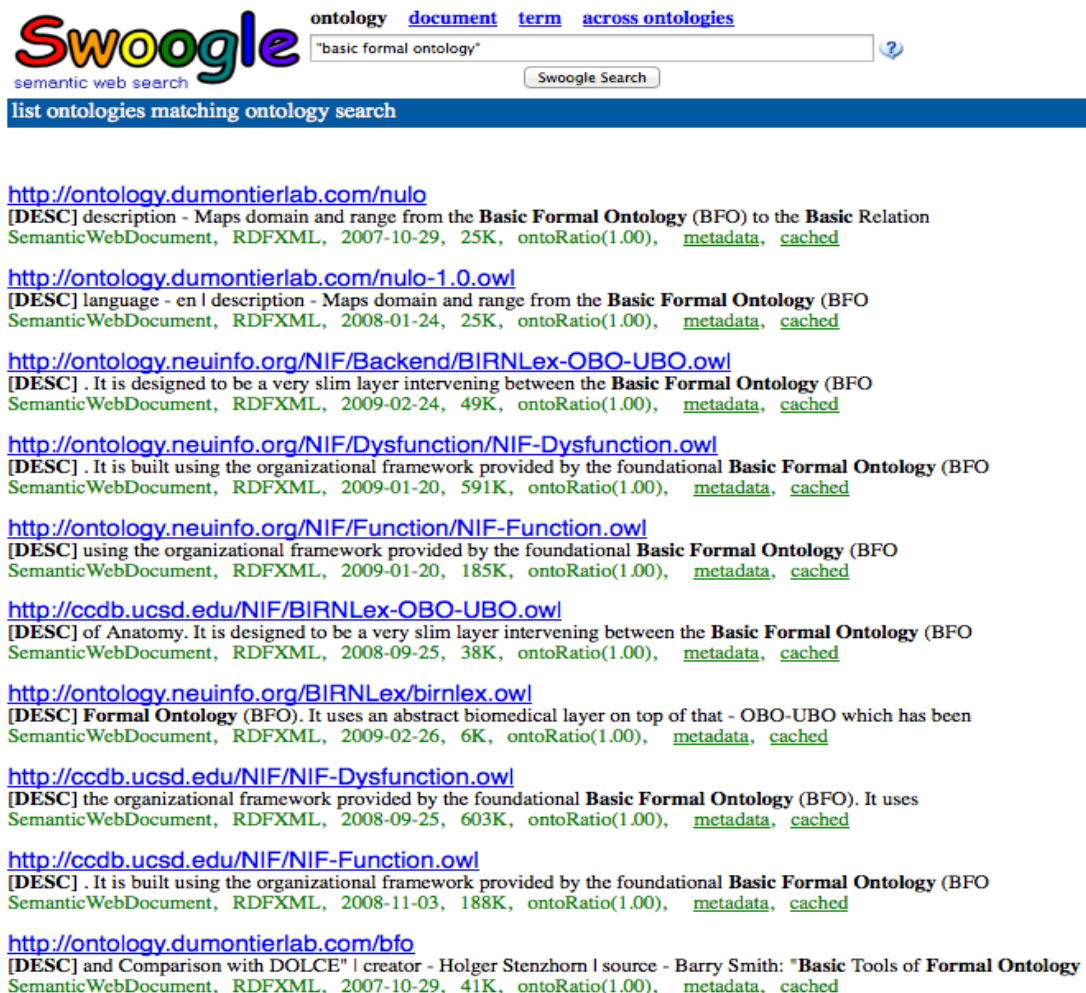
The EO makes extensive use of BFO’s Universal-Universal Relation Ontology, which we described in the previous chapter. BFO_0000118 is the UAI for BFO’s *has_proper_occurent_part* relation, and figure 8.22 shows one of the ways that this relation is

used in the OE (in the Object Property Usage box), along with many other BFO relations listed on the left in the Object properties box. Recall that relations of the kind specified in BFO's Relation Ontology are referred to as object properties in OWL.

7. Facilitation of Interoperability

Earlier in this text, we emphasized the perhaps self-evident point that for two or more computer-based information systems to be interoperable it is essential that they employ a common terminology or have an explicit set of instructions for the mapping of one terminology into the language of the other and conversely. BFO works with OWL and other semantic technologies to facilitate such interoperability.

Figure 8.23: Several BFO Results from Swoogle



The screenshot shows the Swoogle search interface. The search bar contains the text "basic formal ontology". Below the search bar, there is a blue header with the text "list ontologies matching ontology search". The search results are listed below, each with a URL, a description, and metadata.

Swoogle [ontology](#) [document](#) [term](#) [across ontologies](#)
semantic web search "basic formal ontology" Swoogle Search

<http://ontology.dumontierlab.com/nulo>
[DESC] description - Maps domain and range from the **Basic Formal Ontology** (BFO) to the **Basic Relation**
SemanticWebDocument, RDFXML, 2007-10-29, 25K, ontoRatio(1.00), [metadata](#), [cached](#)

<http://ontology.dumontierlab.com/nulo-1.0.owl>
[DESC] language - en | description - Maps domain and range from the **Basic Formal Ontology** (BFO)
SemanticWebDocument, RDFXML, 2008-01-24, 25K, ontoRatio(1.00), [metadata](#), [cached](#)

<http://ontology.neuinfo.org/NIF/Backend/BIRNLex-OBO-UBO.owl>
[DESC] . It is designed to be a very slim layer intervening between the **Basic Formal Ontology** (BFO)
SemanticWebDocument, RDFXML, 2009-02-24, 49K, ontoRatio(1.00), [metadata](#), [cached](#)

<http://ontology.neuinfo.org/NIF/Dysfunction/NIF-Dysfunction.owl>
[DESC] . It is built using the organizational framework provided by the foundational **Basic Formal Ontology** (BFO)
SemanticWebDocument, RDFXML, 2009-01-20, 591K, ontoRatio(1.00), [metadata](#), [cached](#)

<http://ontology.neuinfo.org/NIF/Function/NIF-Function.owl>
[DESC] using the organizational framework provided by the foundational **Basic Formal Ontology** (BFO)
SemanticWebDocument, RDFXML, 2009-01-20, 185K, ontoRatio(1.00), [metadata](#), [cached](#)

<http://ccdb.ucsd.edu/NIF/BIRNLex-OBO-UBO.owl>
[DESC] of Anatomy. It is designed to be a very slim layer intervening between the **Basic Formal Ontology** (BFO)
SemanticWebDocument, RDFXML, 2008-09-25, 38K, ontoRatio(1.00), [metadata](#), [cached](#)

<http://ontology.neuinfo.org/BIRNLex/birnlex.owl>
[DESC] **Formal Ontology** (BFO). It uses an abstract biomedical layer on top of that - OBO-UBO which has been
SemanticWebDocument, RDFXML, 2009-02-26, 6K, ontoRatio(1.00), [metadata](#), [cached](#)

<http://ccdb.ucsd.edu/NIF/NIF-Dysfunction.owl>
[DESC] the organizational framework provided by the foundational **Basic Formal Ontology** (BFO). It uses
SemanticWebDocument, RDFXML, 2008-09-25, 603K, ontoRatio(1.00), [metadata](#), [cached](#)

<http://ccdb.ucsd.edu/NIF/NIF-Function.owl>
[DESC] . It is built using the organizational framework provided by the foundational **Basic Formal Ontology** (BFO)
SemanticWebDocument, RDFXML, 2008-11-03, 188K, ontoRatio(1.00), [metadata](#), [cached](#)

<http://ontology.dumontierlab.com/bfo>
[DESC] and Comparison with DOLCE" | creator - Holger Stenzhorn | source - Barry Smith: "**Basic Tools of Formal Ontology**
SemanticWebDocument, RDFXML, 2007-10-29, 41K, ontoRatio(1.00), [metadata](#), [cached](#)

Consider BFO's presence on the Web. Figure 8.24 shows several of the many ontologies that utilize BFO as a result of a query in Swoogle, which is a Web search engine that looks for RDF on the Web and is funded by the Computer Science and Electrical Engineering Department at the University of Maryland, Baltimore County (UMBC).

Figure 8.24: BFO's Object Usage in Ontobee

Ontology listed in Ontobee	Ontology OWL file	View class in context	Project home page
Ontology of Adverse Events	oae.owl	'object' in oae.owl	OAE
Vaccine ontology	VO.owl	'object' in VO.owl	VO
Uber anatomy ontology	ext.owl	'object' in ext.owl	Project home page
Brucellosis Ontology	brucellosis.owl	'object' in brucellosis.owl	IDOBRU: Brucellosis Ontology
Ontology for General Medical Science	oqms.owl	'object' in oqms.owl	OGMS Home
Cardiovascular Disease Ontology	cvdo.owl	'object' in cvdo.owl	CVDO
Mental Functioning Ontology	mf.owl	'object' in mf.owl	MF
Ontology of Genes and Genomes - Plasmodium falciparum	oqq-pf.owl	'object' in oqq-pf.owl	Project home page
Interaction Network Ontology	INO.owl	'object' in INO.owl	INO
Ontology for genetic interval	oqi.owl	'object' in oqi.owl	QGI
Ontology of Genes and Genomes - Yeast	oqq-sc.owl	'object' in oqq-sc.owl	Project home page
Human Interaction Network Ontology	hino.owl	'object' in hino.owl	Project home page
Ontology of Genes and Genomes	oqq.owl	'object' in oqq.owl	Project home page
Ontology of Vaccine Adverse Events	ovae.owl	'object' in ovae.owl	Project home page
microRNA Ontology	mirnao.owl	'object' in mirnao.owl	Project home page
Emotion Ontology	mfoem.owl	'object' in mfoem.owl	MFOEM
Ontology of Genes and Genomes - Arabidopsis thaliana	oqq-at.owl	'object' in oqq-at.owl	Project home page
PRotein Ontology (PRO)	ftp://ftp.pir.georgetown.edu/databases/ontology/pro_obo/pro.obo	'object' in ftp://ftp.pir.georgetown.edu/databases/ontology/pro_obo/pro.obo	Protein Ontology
Ontology of Genetic Susceptibility Factor	oqsf.owl	'object' in oqsf.owl	OGSF
Ontology of Genes and Genomes - Fruit Fly	oqq-dm.owl	'object' in oqq-dm.owl	Project home page
Ontology of Genes and Genomes - Mouse	oqq-mm.owl	'object' in oqq-mm.owl	Project home page
Ontology of Genes and Genomes - Zebrafish	oqq-dr.owl	'object' in oqq-dr.owl	Project home page
Ontology of Genes and Genomes - Caenorhabditis elegans	oqq-ce.owl	'object' in oqq-ce.owl	Project home page
Informed Consent Ontology	ico.owl	'object' in ico.owl	Project home page

In the fifth chapter, we mentioned the Open Biological and Biomedical Ontologies (OBO) Foundry initiative, “a collaborative experiment involving developers of science-based ontologies who are establishing a set of principles for ontology development with the goal of creating a suite of orthogonal interoperable reference ontologies in the biomedical domain.”²⁸ Not only is it the case that BFO and the principles found in this book are an integral part of the

OBO Foundry, but also the OGMS, IDO, IAO, OBI, MFO, EO, and more than 100 other ontologies such as the Gene Ontology, Foundational Model of Anatomy, Cell Ontology, and Protein Ontology are participants in the effort. The Foundry's website features a link to the University of Michigan Medical School's Ontobee, which is the default linked ontology data server for OBO Foundry library ontologies. As noted on their website, Ontobee is "aimed to facilitate ontology data sharing, visualization, query, integration, and analysis. Ontobee dynamically dereferences and presents individual ontology term URIs to (i) *HTML web pages* for user-friendly web browsing and navigation, and to (ii) *RDF source code* for Semantic Web applications."²⁹ Figure 8.23 shows a webpage from Ontobee. This particular page lists several of the domain ontologies that use BFO 'object,' complete with links to the ontologies in OWL, the usage of object in the context of the ontology, and the ontology's website homepage.

Further Reading in OWL, RDF-S, and RDF

Allemang, Dean and Jim Hendler. *Semantic Web for the Working Ontologist*. Wakham, MD: Morgan Kaufmann, 2011.

Cimiano, Philipp, Christina Unger, and John McCrae. *Ontology-Based Interpretation of Natural Language*. San Rafael, CA: Morgan & Claypool, 2014.

Hitzler, Pascal, Markus Krötzsch, and Sebastian Rudolph. *Foundations of Semantic Web Technologies*. Boca Raton, FL: Chapman & Hall, 2009.

Horrocks, Ian. "Ontologies and the Semantic Web." *Communications of the ACM*, 51.12 (2008): 58-67.

¹ Taken from <http://www.ifomis.org/bfo/users>.

² See <http://protege.stanford.edu/>.

³ Protégé has a useful tutorial titled "Protégé OWL Tutorial," written by Matthew Horridge and others found here: <http://130.88.198.11/tutorials/protegeowltutorial/>.

-
- ⁴ See the W3C's document titled "OWL Web Ontology Language Overview: W3C Recommendation," located here: <http://www.w3.org/TR/owl-features/>; also see W3C's "DAML+OIL (March 2001) Reference Description," located here: <http://www.w3.org/TR/daml+oil-reference>.
- ⁵ "OWL Web Ontology Language Overview: W3C Proposed Recommendation," available here: <http://www.w3.org/TR/2003/PR-owl-features-20031215/>.
- ⁶ See the W3C's document titled "A History of HTML," located here: <http://www.w3.org/People/Raggett/book4/ch02.html>.
- ⁷ <http://www.w3.org/MarkUp/>; also see the W3C's document titled "XML: Development History," located here: <http://www.w3.org/XML/hist2002>.
- ⁸ See the W3C document titled "Resource Description Framework (RDF) Model and Syntax Specification," located here: <http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/>.
- ⁹ See the W3C document titled "RDF/XML Syntax Specification," located here: <http://www.w3.org/TR/REC-rdf-syntax/>.
- ¹⁰ <http://www.w3.org/TR/rdf-sparql-query/>; <http://www.w3.org/TR/sparql11-overview/>.
- ¹¹ <http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/>.
- ¹² See Grau, Bernardo, Ian Horrocks, Boris Motik, Bijan Parsia, Peter Patel-Schneider, and Ulrike Sattler. "OWL 2: The next step for OWL." *Web Semantics: Science, Services and Agents on the World Wide Web*. (2008) 6.4: 309-322. Available here: <http://www.sciencedirect.com/science/article/pii/S1570826808000413>.
- ¹³ The problems with RDF and RDF-S just mentioned, as well as others, are discussed in the W3C document titled "Web Ontology Language (OWL) Use Cases and Requirements," located here: <http://www.w3.org/TR/2003/WD-webont-req-20030331/>; also see Antoniou, Grigoris and Frank van Harmelen. "Web Ontology Language: OWL," in Steffan Staab and Rudi Studer (eds.). *Handbook on Ontologies* (91-110). Berlin: Springer, 2009.
- ¹⁴ See the W3C's document titled "OWL Web Ontology Language Semantics and Abstract Syntax," located here: <http://www.w3.org/TR/owl-semantics/>; also Horrocks, Ian, Peter Patel-Schneider, Deborah McGuinness, and Christopher Welty, "Ontology Languages for the Semantic Web," in Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider (eds.). *The Description Logic Handbook* (458-486). Cambridge: Cambridge University Press, 2003.
- ¹⁵ See the W3C's document titled "OWL Web Ontology Language: Test Cases," located here: <http://www.w3.org/TR/2004/REC-owl-test-20040210/>.
- ¹⁶ See the W3C's document titled "OWL Web Ontology Language Overview," located here: <http://www.w3.org/TR/owl-features/>.
- ¹⁷ <http://www.w3.org/TR/owl-features/>.
- ¹⁸ Horridge, Matthew, Holger Knublauch, Alan Rector, Robert Stevens, and Chris Wroe. "A Practical Guide to Building OWL Ontologies Using the Protégé-OWL Plugin and CO-ODE Tools: Edition 1.0," Manchester: University of Manchester, 12.
- ¹⁹ See, for example, Motik, Boris, Ian Horrocks, and Ulrike Sattler. "Bridging the Gap Between OWL and Relational Databases," *Journal of Web Semantics*. (2009) 7.2: 74-89. Available here: <http://www.cs.ox.ac.uk/people/ian.horrocks/Publications/download/2009/MoHS09a.pdf>.
- ²⁰ See Grau, Bernardo, Ian Horrocks, Boris Motik, Bijan Parsia, Peter Patel-Schneider, and Ulrike Sattler. "OWL 2: The next step for OWL." *Web Semantics: Science, Services and Agents on the*

World Wide Web. (2008) 6.4: 309-322. Available here:

<http://www.sciencedirect.com/science/article/pii/S1570826808000413>.

²¹ <http://www.w3.org/TR/owl2-overview/>.

²² <https://code.google.com/p/ogms/>.

²³ See <https://code.google.com/p/ogms/>.

²⁴ http://infectiousdiseaseontology.org/page/Main_Page.

²⁵ <https://code.google.com/p/information-artifact-ontology/>.

²⁶ <https://code.google.com/p/emotion-ontology/>.

²⁷ <https://code.google.com/p/mental-functioning-ontology/>.

²⁸ <http://www.obofoundry.org/>.

²⁹ <http://www.ontobee.org/>.

DRAFT