8 Basic Formal Ontology at Work

All good ontology building, in our view, starts with thinking. The author of an ontology should first assemble a collection of the major terms he will need to use, and use careful thinking to ensure that he understands the meanings of these terms within the context of the associated science, and then further careful thinking to ensure that he can define these terms using words that other human beings will understand and in ways conformant to the BFO ontology and to the other principles set forth in the foregoing.

At some point, however, the ontology builder will need to embark on the process of creating the ontology as a computer artifact, a piece of software that can be used and reasoned with. In the future we believe that a range of different sorts of approaches to the building of such computer artifacts will be available, including an approach based on first-order logic (FOL) along the lines illustrated in the set of sample axioms provided in the previous chapter. FOL, in our view, provides the sort of expressivity that is needed to create formal definitions that will be conformant with the content of biological and other scientific disciplines. Currently, however, the most widely used standard approaches are focused on formal languages that have an expressivity weaker than FOL, primarily the Web Ontology Language (OWL). In this chapter we provide a brief description of the Protégé ontology-building tool and of OWL, which is the primary logico-linguistic framework used in what are called Semantic Web technologies, practitioners of which are a target audience for Protégé. We then provide examples of domain ontologies that utilize BFO as an upper-level ontology in order to give the reader an opportunity to see how the principles and recommendations from the previous chapters are used in practice.

The Protégé Ontology Editor and BFO

In this book, our focus has been on giving researchers an introduction to theoretical principles and on the strategies for creating good ontology content—rather than on the computational tools and details of ontology implementations. That said, we will describe briefly one major public-domain ontology building and editing tool known as Protégé.[1] We will talk also about OWL,[2] which is the primary logico-linguistic framework used in what is called the "Semantic Web" that is supported by Protégé.[3]

Protégé is described at http://protege.stanford.edu/ as a "free, open-source ontology editor and framework for building intelligent systems." The software is freely available for download on any computer, and with the help of tutorials provided at the Protégé website it enables users to start composing domain ontologies. These ontologies are intended, potentially with the help of BFO, to be interoperable with other domain ontologies built in a similar way.[4]

The latest OWL version of BFO can always be found at the following link: http://purl.obolibrary.org/obo/bfo.owl. Once users have downloaded and installed the Protégé software, they can simply import this latest version into a new Protégé file and start constructing an ontology with terms ("classes") relevant to their respective domain by defining these as subclasses of terms in BFO. At various points in this chapter, we will feature screenshots of ontologies that have been produced using this method.

The Web Ontology Language (OWL)

Protégé is a tool that has ontologies constructed using OWL as one of its outputs. OWL evolved from two sources that were merged in the early 2000s: the U.S. Defense Advanced Research Projects Agency (DARPA) *Agent Markup Language* (DAML) and the European Union's *Ontology Inference Layer* (OIL).[5] By 2002 the combined resource (then called "DAML+OIL") had been recommended for use by the World Wide Web Consortium (W3C), which is one of the principal Web standards organizations.[6] 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 World Wide Web itself, Tim Berners-Lee. In February 2004 OWL received the status of a recommendation by the W3C with the following abstract, still accurate 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 (RDFS) by providing additional vocabulary along with a formal semantics."[7]

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

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

HTML was developed by Berners-Lee in the late 1980s. It was designed to allow Web developers to display information in a way that is accessible to humans for viewing via Web browsers.[8]

HTML's primary limitation is that it is oriented toward representations of documents, and so it does not provide the capability to describe information in ways that facilitate the use of software programs to find, interpret, validate, or combine it. Thus, in 1998 the W3C recommended for use the Extensible Markup Language (XML) that allows information to be more accurately described utilizing tags that are not only understandable by humans but also readable and interpretable by machines. 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."[9]

Resource Description Framework (RDF)

Unfortunately, while XML is useful when one wants to query documents, it has a limited capability to convey in a standard form the meanings of the statements contained therein, for example, as concerns the relationships between and among entities, and this prompted researchers to develop on its basis the Resource Description Framework (RDF). RDF became a W3C recommendation in February 1999.[10] It is a language created to allow the representation of relationships between entities by means of a simple subject-predicate-object format known as a *triple*. RDF thus to a degree mimics the structure of human language. It also allows for a computational representation of entities and relationships that allows reasoning that is more powerful than can be achieved using XML alone.

The *resource* part of RDF is inspired by the ability provided by the Internet to use Uniform Resource Identifiers (URIs), which is to say standard web addresses, to unambiguously identify web pages on the Internet. In RDF the use of "resource" refers to the fact that all of the subjects, predicates, and objects making up each triple are seen as referring to entities in reality, all of which are called "resources." XML does not represent reality in this way at all, though it does provide specifications for representing dates, numbers, and symbols.

The triples form an RDF database—called a *triplestore*—which can be populated with detailed information about some domain. The subjects, predicates, and objects are tagged with URIs (or in certain cases by what are called "blank nodes"), and the information can be placed on the Web so that anyone in the world can query the database.

RDF Schema (RDFS)

Immediately after RDF began to be used in the early 1990s, RDF Schema (RDFS) was developed as a set of mechanisms for describing groups of related resources (understood, again, in terms of URIs) and of the relationships between them.

Where RDF allows assertions about instances and literals such as integers or alphanumeric strings, RDFS allows also assertions about types: *schematic* assertions. Thus, for example, it allows the representation of the domains and ranges of relations, which in the Semantic Web world are called "properties." For example, it allows the property *authored_by* to be characterized as having a domain *document* and a range *person*. Information of this sort can then be saved in a triplestore and queried with SPARQL (see next section).[11] RDFS introduces a number of predicates including:

- rdfs:label, refers to a string of text describing a resource
- rdfs:comment, points to a human-readable comment about the resource, also often used when providing a definition
- rdfs:seeAlso, which links to other relevant resources
- rdfs:subClassOf, used to state that all the instances of one class are instances of another
- rdfs:domain, used to state that any resource that has a given property is an instance of one or more classes
- rdfs:range, used to specify the set of values that a property can accept

By adding such elements, RDFS was able to augment the expressivity of RDF in order to enable more adequate representation of given domains of interest.

However, almost immediately after RDFS was developed, researchers began to see that it was still too weak to talk about types and about instances belonging to a type, as well as about properties of relations. Further, given that many of these same researchers had been working in the field of artificial intelligence, they wanted a language that could enable machine reasoning.[12] But RDFS is not able, for example, to express the assertion that some relation is transitive in a way that would allow users of RDFS to exploit transitivity in their reasoning.

RDFS is similarly unable to account for the symmetric, reflexive, and other properties of relations that we discussed in chapter 7. It is impossible to say in RDFS, for example, that a relation is symmetrical or that one relation is the inverse of another.

Further, although RDFS added the ability to specify the domain and range of properties, 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 *dog* when applied to dogs, *cat* when applied to cats, and so forth. Also, there is no way to express existential or cardinality constraints in RDFS. One cannot say, for example, that all instances of *person* have a *mother* that is also a *person*, or that *persons* have *exactly two biological parents*.[13] These and other problems with RDFS prompted researchers to develop OWL.

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 triples are queried in a triplestore with what is known as the Simple Protocol and RDF Query Language (SPARQL). SPARQL 1.1 became a W3C recommendation in March 2013.[14]

A simple SPARQL query might look like this:

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

which asks for all of the predicates in a triplestore.

Basic Features of OWL

In the preceding chapters we have distinguished universals (such as *person* and *role*), whose instances are entities in their own right, from defined classes, which we can think of as devices to capture certain ways of speaking about reality (as, when we speak of lawyers or students, we are in fact speaking merely of *persons* with special *roles*). In the worldview of RDF and OWL, now, universals and defined classes are in effect run together under the single heading of "classes"; and (binary) relations are referred to as "properties." (OWL does share with BFO the use of "instance"; it uses this term to refer to the individuals that are the members of classes as it conceives them.) OWL is in this respect close to a traditional set-theoretic view of reality; however, because it allows for there to be two classes with the same members, OWL classes are in fact intensional.

A key feature of OWL is its basis in Description Logics (DLs), a family of logics that are expressively weaker than standard FOL, but enjoy certain computational properties advantageous for purposes such as ontology-based reasoning and data validation. DLs emerged out of the artificial intelligence community in the mid-1980s and have a formal semantics typically expressed using model theory of the sort familiar from the world of FOL. In contradistinction to XML, RDF, and RDFS, OWL allows for the expression of

a. universal (∀) quantification, through the owl:allValuesFrom restriction;

b. existential (∃) quantification, through the owl:someValuesFrom or owl:hasValue restriction;.

c. cardinality through owl:cardinality, owl:minCardinality, and owl:maxCardinality;

d. the Boolean *and*, *or*, and *not*, which in OWL are called owl:intersectionOf, owl:unionOf, and owl:complementOf, respectively;

e. assertions of equivalence, through owl:equivalentClass and owl:equivalentProperty (thus it is possible to assert for example that two classes X and Y are equivalent, by which is meant that they have the same members [apply to the same instances]); and

f. properties declared to hold of relations, including inverse (owl:inverseOf), functional (owl:FunctionalProperty), inverse functional (owl:InverseFunctionalPrope rty), transitive (owl:TransitiveProperty), symmetric (owl:SymmetricProperty), asymmetric (owl:AsymmetricProperty), reflexive (owl:ReflexiveProperty), and irreflexive (owl:IrreflexiveProperty).

OWL recognizes two types of properties: *object properties* and *data 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,” where “wing,” “airplane,” “wrist,” and “hand” refer to specific instances of the corresponding classes, not to the classes (or universals) themselves. A data property (owl:DatatypeProperty) specifies a relationship between an individual and a literal (integer, double, float, string, boolean, etc.); for example, “the number of participants in the meeting was 6.”

To check the consistency of any given set of facts or axioms (including definitions as this term is used in the preceding), and thus in particular to check the consistency of an ontology in OWL, we must use *reasoners* which can determine for each class in the ontology whether it is possible for that class to have instances.[15] A number of reasoners can be used in conjunction with Protégé, which are discussed in the appendix.

OWL vs. Standard Relational Databases

There are three further points to be made about ontologies built in OWL that can best be seen by contrasting the use of such ontologies with the use of standard relational databases.

First, in the latter each instance/individual must have a unique identifier; for example, the planet Venus must be referenced using one name only, such as “Venus,” or your grandmother must be referenced using only one identifier, for example, her Social Security number. This is because it is rows in a database that are typically used to represent instances and there is a rule (the primary key uniqueness constraint) that prevents two rows being used to represent the same instance. In an OWL ontology, however, 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," "Nana," "Florence Smith," or all three. SameAs axioms are then included (or SameAs relations are inferred) to assert identity.

Second, the closed-world assumption is standardly presupposed by those working with relational databases. The means that *what is not known to be true in the database* is by default considered false, because knowledge of the world represented in the database is assumed to be complete. (This assumption is possible in virtue of the fact that the world of the database is defined exhaustively by the database itself.) Ontologies, by contrast, utilize the open-world assumption (OWA), whereby *what is not known to be true* is always considered to be, simply, *not known*. Thus ontologies are particularly useful for dealing with domains, such as biological science, where our knowledge of the world is seen as being always incomplete, because the science itself is rapidly advancing.

The way that the closed-world assumption is practically made manifest on the standard database approach can be shown with the following simple example. We imagine a database consisting exactly of the information presented in this table:

Individuals	Attributes
Fido	dog
Rover	

An SQL query addressed to this database asking, "Is Rover an instance of a dog?" would yield the answer "no." On the open-world approach adopted by ontologists, in contrast, a query would not return *anything*, since we have no evidence as to whether Rover is or is not an instance of dog. Similarly, to the query "How many dogs are there?" the database would come back with an answer: exactly 1; the ontology would return: at least 1, but possibly more.

Third, and most important, a relational database schema exists primarily to constrain and structure the data, and it is very difficult to address complex queries over the data, or to use reasoners to check for consistency. The relations in a database themselves do not (and cannot) have properties inherent in them such as transitivity or symmetry. Rather, they are simply listed, like all other entries in the database. Given its basis in DL, (and thus in FOL), however, OWL's axioms and rules—including attributes of relations such as transitivity and symmetry—have been designed to facilitate the generation of implications and inferences from the ontologies that are formulated in its terms. For example, if you specify an OWL ontology in which the relation *is_pet_of* is defined to have domain *nonhuman animal* and range *person*, then if you assert the statement Rover *is_pet_of* Jim, you will be able conclude that Rover is a nonhuman animal and that Jim is a person.

OWL 2

Shortly after researchers started using OWL in the early 2000s, they began to realize its limitations. One basic problem with the initial version of OWL was its inability to express and reason over qualified cardinality restrictions. For example, while it is easy to say in OWL 1 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 that someone has four dogs as pets, one of which is a poodle and the other three are boxers. Another problem had to do with restrictions placed on literals. For example, while it is possible to say in OWL 1 that the barometric pressure in a particular part of the atmosphere has a value of 1,000 millibars, it is not possible to assert, for example, that this value is more than 900 and less than 1,100 millibars.[16] These and other problems prompted researchers to develop OWL 2, which became a W3C recommendation in October 2009.

OWL 2 also defines three profiles that are language subsets—EL, RL, and QL—each with useful computational properties and implementation capabilities. OWL 2 EL is ideal for large-scale ontologies, since using this profile allows many inference problems to be computed in polynomial time. OWL 2 RL is designed to take advantage of rule-based systems to solve inference problems. And in OWL 2 QL inference problems can be implemented as SQL queries against relational databases (RDBs), a very useful feature for practical, working ontologists.

Building Ontologies with Basic Formal Ontology

Tables 8.1 and 8.2 list some of the ontologies, institutions, and groups that have utilized BFO in developing their domain ontologies.[17] Having briefly described the basic features of Protégé, OWL, and associated resources, we are now in a better position to look at a few specific examples of domain ontologies that not only utilize BFO as upper-level ontology, but have also applied the principles and recommendations found in this book during ontology development.

Example: The Ontology for General Medical Science (OGMS)

An inspection of the homepage for the Ontology for General Medical Science (OGMS) reveals the influence of BFO and the recommendations from this book 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.[18]

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, nonredundant, 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 the terminological core of 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 for ontology modules for a range of different diseases and disease families.

OGMS utilizes the Aristotelian definitional structure for all of its entities, as in the example of constitutional genetic disorder, a disorder that pervades the whole organism, shown in figure 8.1, which is a Protégé-generated visualization of a fragment of OGMS that represents *constitutional genetic disorder* as a child of *genetic disorder*, itself a child of OGMS: *disorder*.

Figure 8.2 shows a fragment of the OWL representation of OGMS: *constitutional genetic disorder*. Notice the use of RDFS in line five, which reads

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

Because OGMS_0000047 is the alphanumeric identifier for "genetic disorder," this line communicates the fact that, in OGMS, "constitutional genetic disorder" is a subclass (subtype, kind) of "genetic disorder."

Figure 8.3 shows a portion of the Protégé-generated hierarchy for OGMS containing *constitutional genetic disorder* as part. It shows how the latter can be traced back in the ontology to BFO's *material entity* and from there to *independent continuant*. OGMS also utilizes the Relation Ontology.

Figure 8.4 shows how "disorder" fits into the broader context of OGMS, and the relations that connect it to the other core terms in the ontology, including terms that are among the most generally used in clinical medicine. What OGMS provides is a set of

Table 8.1
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	Exposé	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

Table 8.1 (continued)

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 Event	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

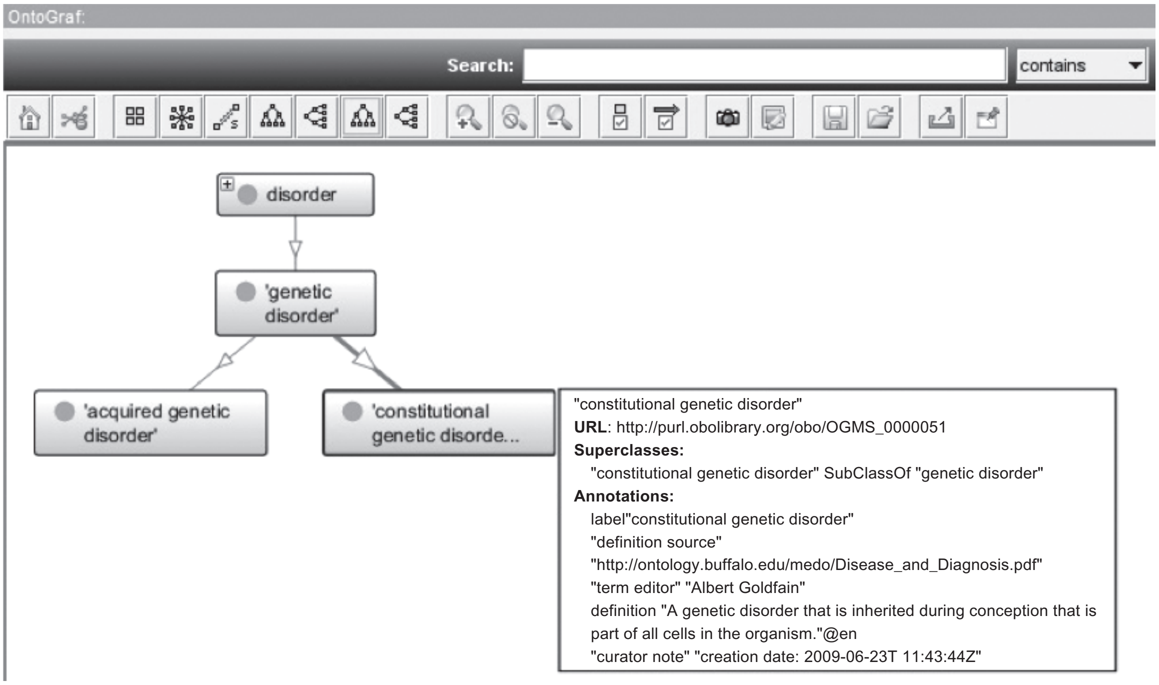
coherent definitions for these terms, built around a view of a disease as a certain sort of power or potentiality—roughly, the potentiality for signs and symptoms to be manifested. The disease exists in the organism by virtue of physical disorders in that organism, for instance a disordered liver or a disordered lung. These powers or potentialities are BFO: *dispositions*, and so the OGMS defends the general view according to which diseases are special types of *disposition* that themselves can be traced up the *is_a* hierarchy to BFO: *specifically dependent continuant*, as shown in figure 8.5.

OGMS has been designed to serve as the framework for describing the entities involved in a clinical encounter. This means: describing how specific types of physical disorders relate to abnormal dispositions on the side of patients, dispositions realized

Table 8.2
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	CUBRC
Data-Tactics Corporation	DOQS: Data-Oriented Quality Solutions	DSpace at NTNUA
Dumontier Lab	eagle-i Consortium	Elsevier Smart Content Strategy
EuPathDB	GoodOD	HIGHFLEET
U.S. Army Intelligence and Information Warfare Directorate	Influenza Research Database	INRIA Lorraine Research Unit
Kobe University Department of Sociomedical Informatics	Medical University Graz—Informatics, Statistics and Documentation	National Center for Multi-Source Information Fusion
National Center for Ontological Research	OBO (Open Biological and Biomedical Ontologies) Foundry	OntoCAT
OntoCOG	Open PHACTS	OpenEHR
REMINE	Saitama University	Science Commons—Neurocommons
Skeletome	University Hospital Erlangen—Radiology	University of Arkansas for Medical Sciences, Biomedical Informatics
University of Augsburg, Computer Science, Software Methodologies for Systems	University of Florida Biomedical Informatics	University of Texas Southwestern Medical Center
University of Washington Structural Informatics Group	Virus Pathogen Resource	VIVO

(manifesting themselves) in pathological processes that are recognized by the clinician in a clinical encounter as signs or symptoms and documented in clinical information systems. Clinical application ontologies extend OGMS by refining the basic taxonomic and relational structure for a particular domain of interest. Examples of such 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 Event Ontology (AEO), Ontology for Newborn Screening (ONSTR), Drug Ontology (DrOn), Model for Clinical Information (MCI), Ocular Disease Ontology (ODO), Oral Health and Disease Ontology (OHDO), Mental Functioning Ontology (MFO), and Cardiovascular Disease Ontology (CDO).[19]


OntoGraf:
Search:
contains
disorder
'genetic disorder'
'acquired genetic disorder'
'constitutional genetic disorde...

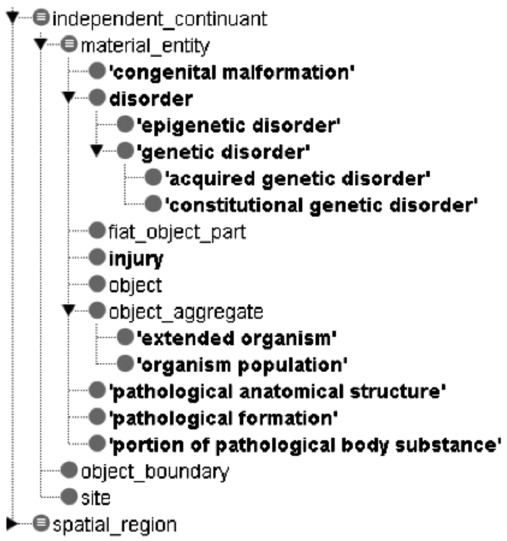
"constitutional genetic disorder"
URL: http://purl.obolibrary.org/obo/OGMS_0000051
Superclasses:
"constitutional genetic disorder" SubClassOf "genetic disorder"
Annotations:
label"constitutional genetic disorder"
"definition source"
"http://ontology.buffalo.edu/medo/Disease_and_Diagnosis.pdf"
"term editor" "Albert Goldfain"
definition "A genetic disorder that is inherited during conception that is part of all cells in the organism."@en
"curator note" "creation date: 2009-06-23T 11:43:44Z"

Figure 8.1
Protégé visualization of OGMS fragment for 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 rdf: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 8.2
Fragment of OWL representation of OGMS



```
▼ independent_continuant
    ▼ material_entity
        'congenital malformation'
        ▼ disorder
            'epigenetic disorder'
            ▼ 'genetic disorder'
                'acquired genetic disorder'
                'constitutional genetic disorder'
        fiat_object_part
        injury
        object
        ▼ object_aggregate
            'extended organism'
            'organism population'
        'pathological anatomical structure'
        'pathological formation'
        'portion of pathological body substance'
    object_boundary
    site
▶ spatial_region
```

Figure 8.3
Constitutional genetic disorder as a type of independent continuant

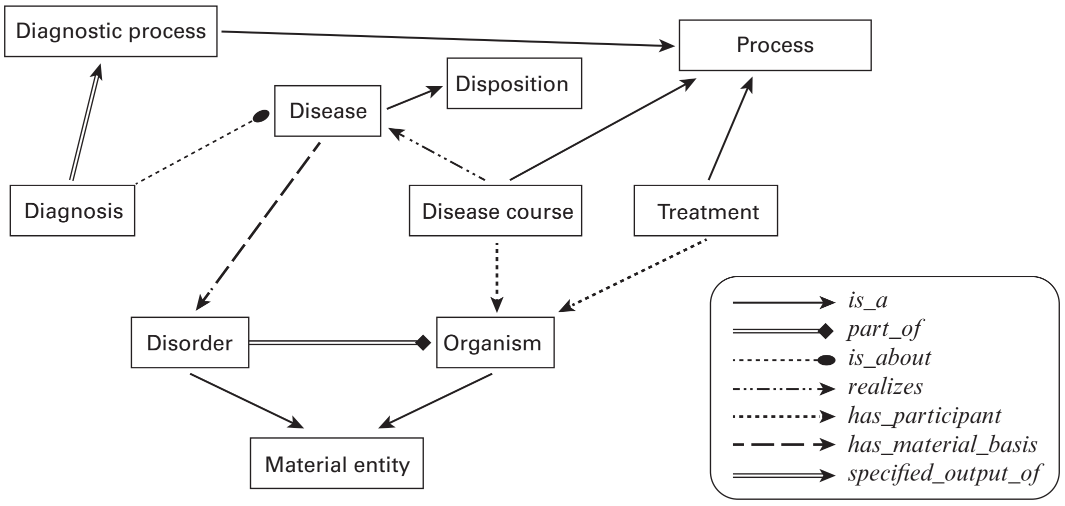
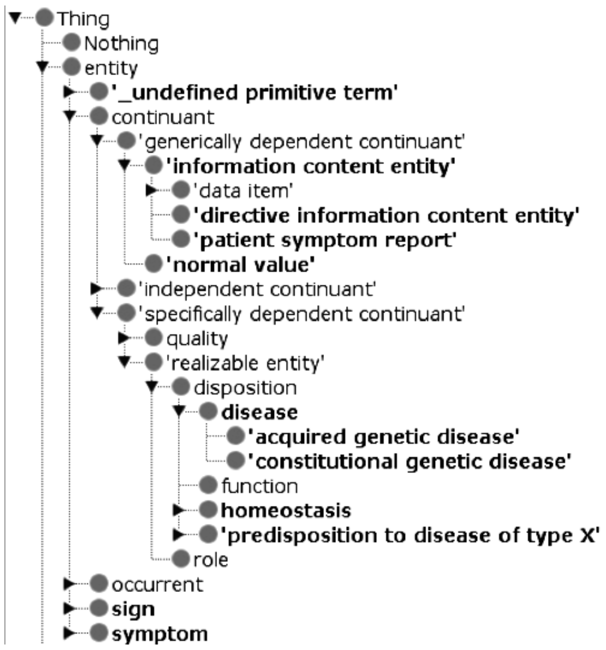


Figure 8.4
Selected core terms and relations of OGMS



```
▼ Thing
    Nothing
  ▼ entity
    ► '_undefined primitive term'
    ▼ continuant
      ▼ 'generically dependent continuant'
        ▼ 'information content entity'
          ► 'data item'
            'directive information content entity'
            'patient symptom report'
          'normal value'
      ► 'independent continuant'
      ▼ 'specifically dependent continuant'
        ► quality
        ▼ 'realizable entity'
          ▼ disposition
            ▼ disease
                'acquired genetic disease'
                'constitutional genetic disease'
              function
            ► homeostasis
            ► 'predisposition to disease of type X'
            role
    ► occurrent
    ► sign
    ► symptom
```

Figure 8.5
Disease as a subtype of specifically dependent continuant in OGMS

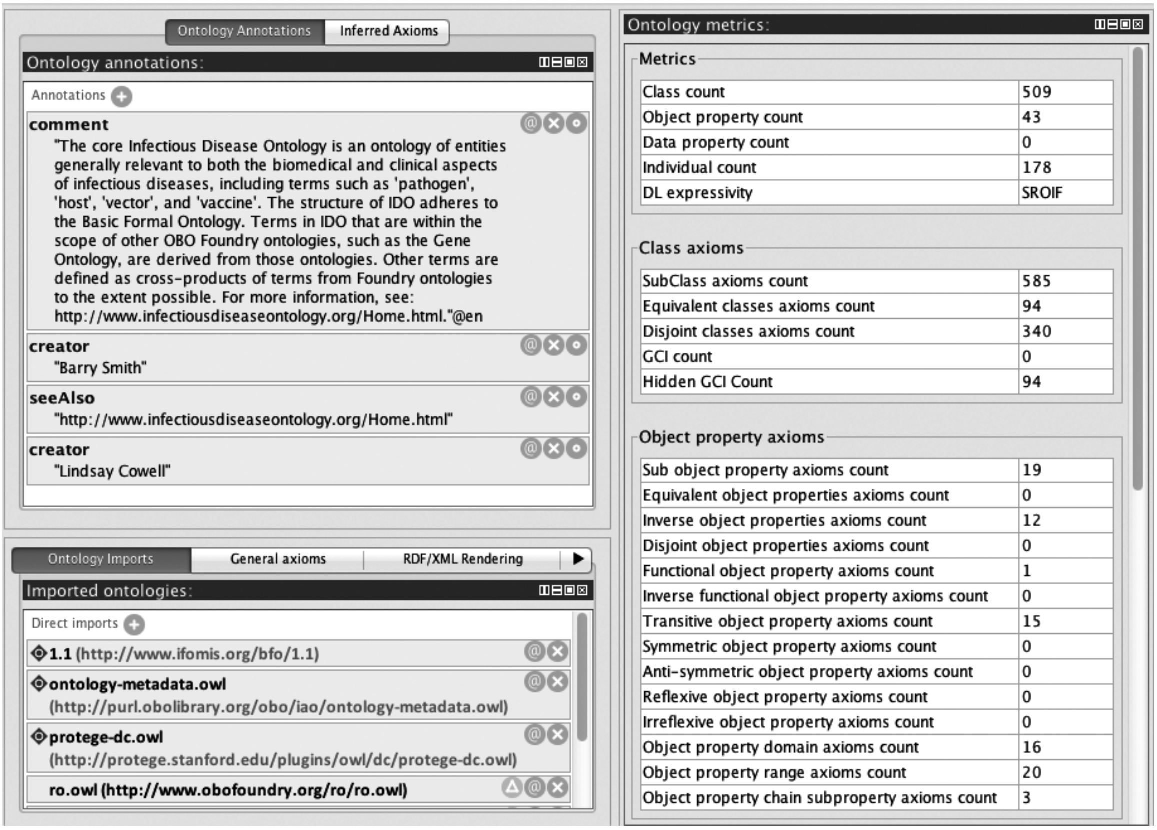
Infectious Disease Ontology (IDO)

The Infectious Disease Ontology (IDO) is an ontology representing over five hundred types or universals relevant to both the biomedical and clinical aspects of infectious diseases. A screenshot of IDO metadata generated by Protégé is shown in figure 8.6.

IDO consists strictly speaking of a suite of ontologies, each of which extends the general IDO-Core, which itself extends OGMS, which extends BFO in its turn. The extensions in the IDO suite are subdomain-specific ontologies primarily relating to specific infectious pathogens, such as IDO-Brucellosis, IDO-Dengue Fever, IDO-HIV, IDO-Infective Endocarditis, IDO-Influenza, IDO-Malaria, IDO-Staphylococcus Aureus, and IDO-Tuberculosis, as well as the vaccine ontology IDO-Vaccines.[20]

Figure 8.7 illustrates a fragment of the IDO in Protégé while figure 8.8 illustrates the position of the term "pathogen," defined as a material entity with a pathogenic disposition, in the IDO ontology. The circle next to this term in the left-hand column contains a tribar (≡) that indicates that the term is equivalent to some other term that behaves as a formal definition, in this case the following term:

material entity and has disposition some pathogenic disposition



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"

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

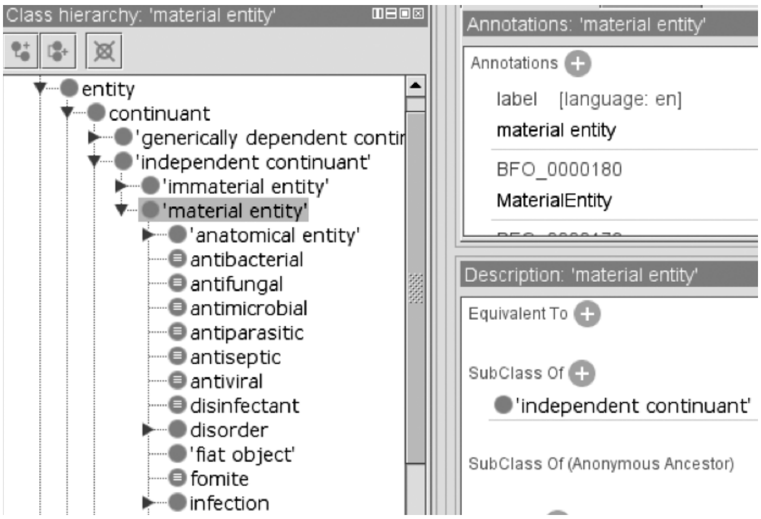
Object property axioms

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

Figure 8.6
IDO metadata

Designating equivalency in this way is also a means by which ontologists may assert synonymy relations between terms in different ontologies.

It is good practice to reuse terms from one domain ontology in another domain ontology, and here, for example, the term "material entity" is taken over from BFO. IDO reuses many terms from other authoritative domain ontologies in formulating its definitions, including "disorder" from OGMS, "molecular entity" from the CHEBI (Chemical Entities of Biological Interest) ontology, and "macromolecular complex" from the Gene Ontology. An illustration of how IDO reuses terms from other ontologies is provided in figure 8.8, which illustrates the rendering of terms (using the Protégé View tab) by alphanumeric identifier.



Class hierarchy: 'material entity'

- entity
  - continuant
    - 'generically dependent contir
    - 'independent continuant'
      - 'immaterial entity'
      - 'material entity'
        - 'anatomical entity'
        - antibacterial
        - antifungal
        - antimicrobial
        - antiparasitic
        - antiseptic
        - antiviral
        - disinfectant
        - disorder
        - 'fiat object'
        - fomite
        - infection

Annotations: 'material entity'

Annotations

label [language: en]
material entity

BFO_0000180
MaterialEntity

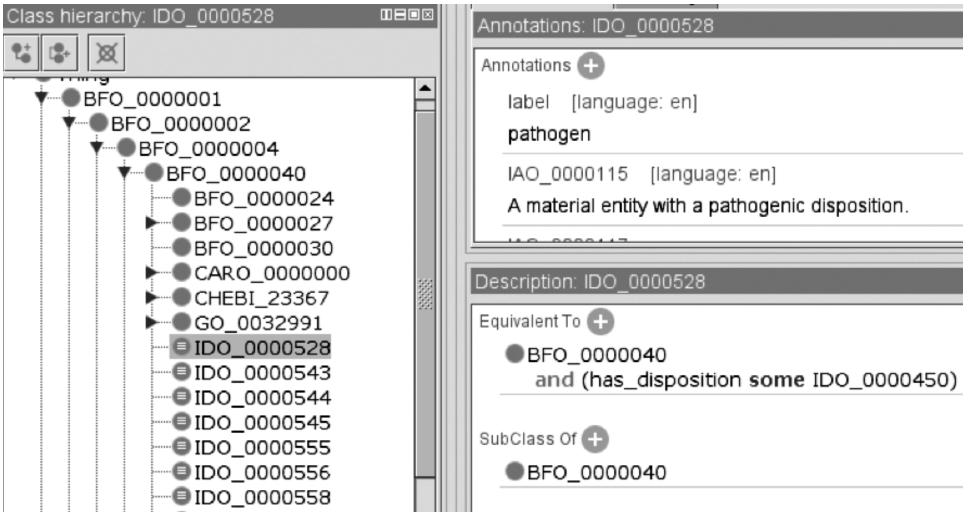
Description: 'material entity'

Equivalent To

SubClass Of
'independent continuant'

SubClass Of (Anonymous Ancestor)

Figure 8.7
Protégé representation of a fragment of IDO



Class hierarchy: IDO_0000528

- BFO_0000001
  - BFO_0000002
    - BFO_0000004
      - BFO_0000040
        - BFO_0000024
        - BFO_0000027
        - BFO_0000030
        - CARO_0000000
        - CHEBI_23367
        - GO_0032991
        - IDO_0000528
        - IDO_0000543
        - IDO_0000544
        - IDO_0000545
        - IDO_0000555
        - IDO_0000556
        - IDO_0000558

Annotations: IDO_0000528

Annotations

label [language: en]
pathogen

IAO_0000115 [language: en]
A material entity with a pathogenic disposition.

Description: IDO_0000528

Equivalent To
BFO_0000040
and (has_disposition **some** IDO_0000450)

SubClass Of
BFO_0000040

Figure 8.8
IDO representation of “pathogen” (IDO_0000528) and its definition

Information Artifact Ontology (IAO)

The Information Artifact Ontology (IAO) is an ontology of information entities based on BFO, emerging out of the need on the part of the developers of the Ontology for Biomedical Investigations (OBI) to categorize the different sorts of information entities that are involved in scientific research, including protocols, databases, experimental logs, published literature, and so forth.[21] IAO concerns the material bearers of information (books, hard drives, photographic prints, traffic signs), information content entities themselves (sentences in a book, XML files on a disk, symbols on a map, 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_translation_of*, and so forth).

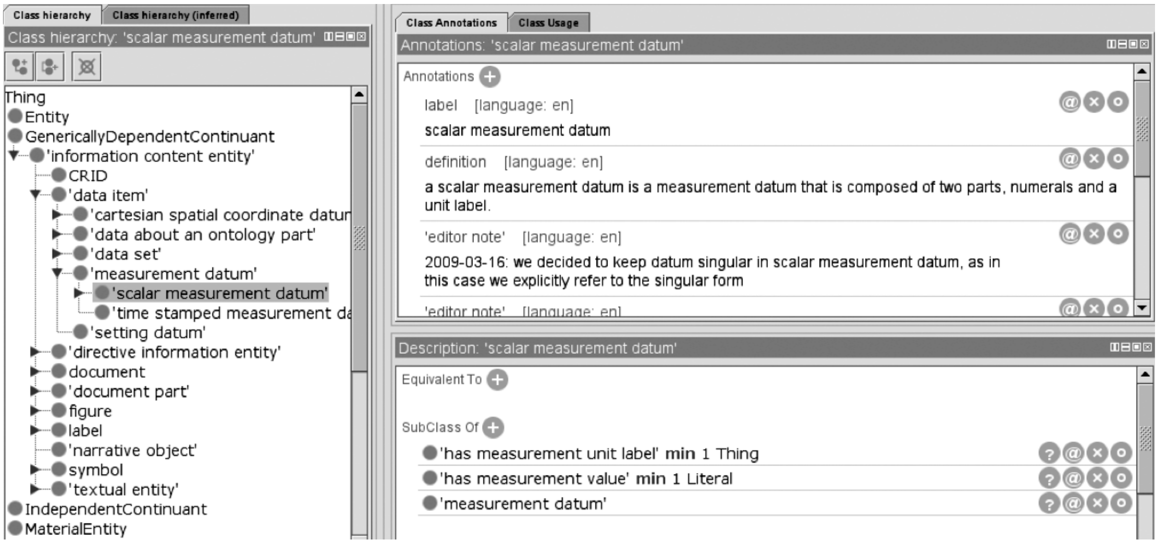
A foundational idea in the IAO is that information content entities are related to other things by being "about" them or denoting them. Information content entities are a subtype of BFO's "generically dependent continuant." The ideas, tables, and figures being communicated right now in this book are examples of information content entities that denote other things such as the OWL and RDF languages, various ontologies, and entities of many other types. The hard copy of the book in your hand is an example of a material bearer of these information content entities.

Figure 8.9 illustrates IAO's treatment of the *scalar measurement datum*, which is defined, in conformity with the Aristotelian template, as a measurement datum with two parts: a numeral and a unit label.

Figure 8.10 shows one of IAO's data properties, *has_measurement_value*, which has IAO_0000032: *scalar measurement datum* as its domain, and as its range *float*. Thus *has_measurement_value* is a relation between a *scalar measurement datum* and a floating-point number. This relation is functional, meaning that an individual scalar measurement datum has one, and only one, measurement value of datatype float.

The Emotion Ontology (MFO-EM)

The Emotion Ontology (MFO-EM) is an extension of the Mental Functioning Ontology (MFO) covering mental processes (such as thinking) and dispositions (such as memory) in a BFO framework.[22] The Emotion Ontology itself comprises over 850 terms representing universals in the domain of affective phenomena such as emotions, moods, appraisals, and subjective feelings.[23] Each aspect of the ontology is rooted in BFO; for example, BFO: *occurrent* is utilized when defining *emotion occurrent* and its subtypes


Class hierarchy | Class hierarchy (inferred)

Class hierarchy: 'scalar measurement datum'

Thing
- Entity
- GenericallyDependentContinuant
  - 'information content entity'
    - CRID
    - 'data item'
      - 'cartesian spatial coordinate datur
      - 'data about an ontology part'
      - 'data set'
      - 'measurement datum'
        - 'scalar measurement datum'
        - 'time stamped measurement da
      - 'setting datum'
    - 'directive information entity'
    - document
    - 'document part'
    - figure
    - label
    - 'narrative object'
    - symbol
    - 'textual entity'
- IndependentContinuant
- MaterialEntity

Class Annotations | Class Usage

Annotations: 'scalar measurement datum'

Annotations

label [language: en]
scalar measurement datum

definition [language: en]
a scalar measurement datum is a measurement datum that is composed of two parts, numerals and a unit label.

'editor note' [language: en]
2009-03-16: we decided to keep datum singular in scalar measurement datum, as in this case we explicitly refer to the singular form

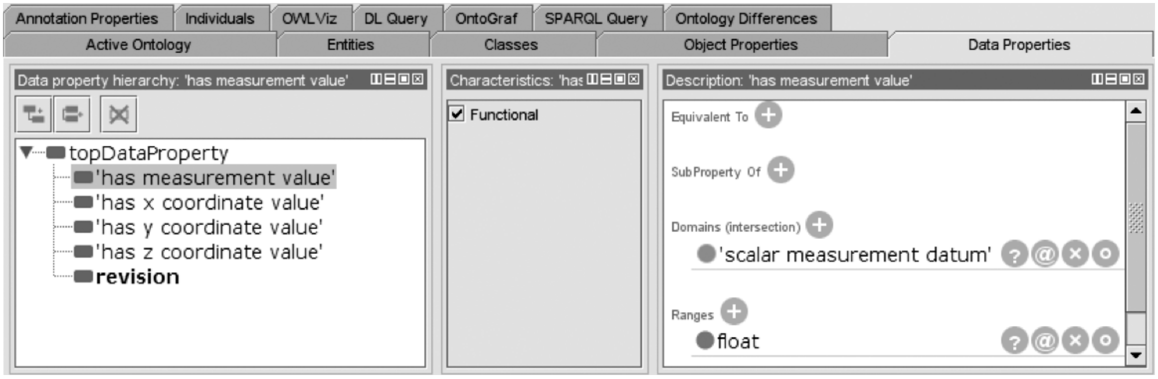
'editor note' [language: en]

Description: 'scalar measurement datum'

Equivalent To

SubClass Of
- 'has measurement unit label' **min** 1 Thing
- 'has measurement value' **min** 1 Literal
- 'measurement datum'

Figure 8.9
IAO: *scalar measurement datum*


Annotation Properties | Individuals | OWLViz | DL Query | OntoGraf | SPARQL Query | Ontology Differences
Active Ontology | Entities | Classes | Object Properties | Data Properties

Data property hierarchy: 'has measurement value'

- topDataProperty
  - 'has measurement value'
  - 'has x coordinate value'
  - 'has y coordinate value'
  - 'has z coordinate value'
  - **revision**

Characteristics: 'has

✓ Functional

Description: 'has measurement value'

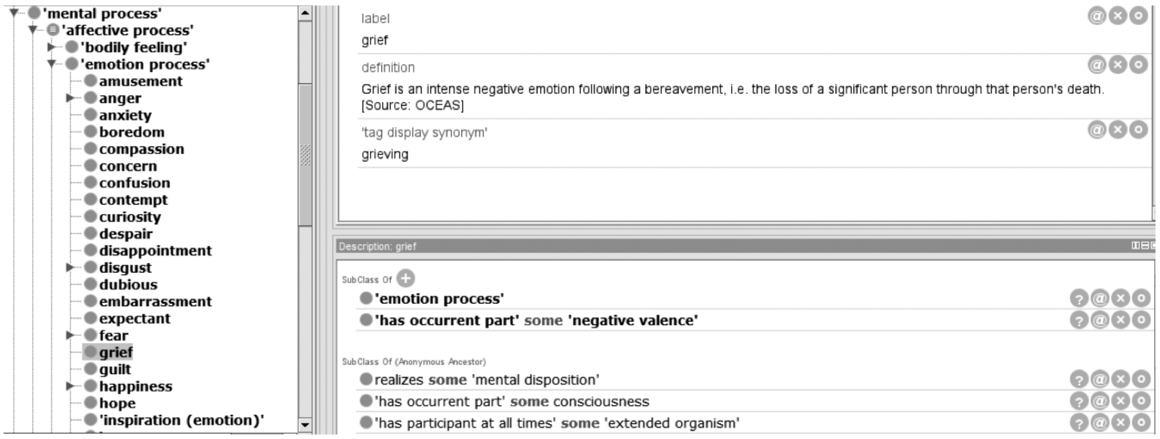
Equivalent To

SubProperty Of

Domains (intersection)
- 'scalar measurement datum'

Ranges
- float

Figure 8.10
IAO's has_measurement_value data property



```
'mental process'
  'affective process'
    'bodily feeling'
    'emotion process'
      amusement
      anger
      anxiety
      boredom
      compassion
      concern
      confusion
      contempt
      curiosity
      despair
      disappointment
      disgust
      dubious
      embarrassment
      expectant
      fear
      grief
      guilt
      happiness
      hope
      'inspiration (emotion)'
```

label
grief

definition
Grief is an intense negative emotion following a bereavement, i.e. the loss of a significant person through that person's death.
[Source: OCEAS]

'tag display synonym'
grieving

Description: grief

SubClass Of
'emotion process'
'has occurrent part' some 'negative valence'

SubClass Of (Anonymous Ancestor)
realizes some 'mental disposition'
'has occurrent part' some consciousness
'has participant at all times' some 'extended organism'

Figure 8.11
MFO-EM: *grief*

anger, *happiness*, and so forth. Similarly, emotion dispositions such as *love* and *hate* are classified in MFO-EM as subtypes of BFO: *disposition*.

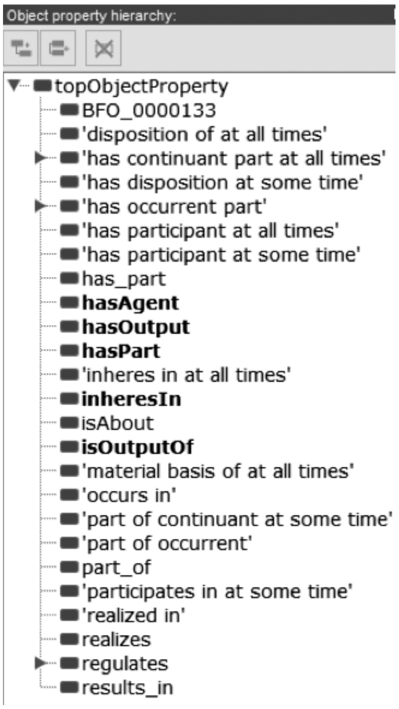
Figure 8.11 provides a fragment of MFO-EM relating to the emotion occurrent: *grief*.

Figure 8.12 shows how MFO-EM, like other ontologies following the principles of the OBO Foundry, makes extensive use of the relations described in chapter 7.

Facilitation of Interoperability

Throughout this book, we have emphasized the advantages that interoperability of information systems brings to a world of increasing quantities of data. BFO promotes such interoperability by allowing the ontologies constructed in its terms to reuse each other's terms, for example, when formulating definitions. This promotes the integration not merely of the ontologies themselves but also of the respective bodies of data annotated in their terms. Queries that cannot be answered when addressed against single bodies of data often yield answers when data are combined through annotation with the same ontologies—so that multiple heterogeneous bodies of data behave as a unified target for ontology-based queries.

In chapter 5, we discussed the 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."[24] Use of BFO and conformity to



topObjectProperty
- BFO_0000133
- 'disposition of at all times'
- 'has continuant part at all times'
- 'has disposition at some time'
- 'has occurrent part'
- 'has participant at all times'
- 'has participant at some time'
- has_part
- **hasAgent**
- **hasOutput**
- **hasPart**
- 'inheres in at all times'
- **inheresIn**
- isAbout
- **isOutputOf**
- 'material basis of at all times'
- 'occurs in'
- 'part of continuant at some time'
- 'part of occurrent'
- part_of
- 'participates in at some time'
- 'realized in'
- realizes
- regulates
- results_in

Figure 8.12
BFO relations used in MFO-EM

the principles found in this book are an integral part of the OBO Foundry approach, and are used also by OBO Foundry candidate ontologies such as OGMS, IAO, MFO, and MFO-EM, and many other ontologies. The result is an expanding virtual framework for navigating through massive amounts of biological and clinical data.

Further Reading on OWL, RDFS, and RDF

Baader, Franz, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge: Cambridge University Press, 2010.

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.

Robinson, Peter N., and Sebastian Bauer. *Introduction to Bio-ontologies*. New York: Chapman and Hall/CRC, 2011.

Zhou, Yujiao, Bernardo Cuenca Grau, Ian Horrocks, Zhe Wu, and Jay Banerjee. “Making the Most of Your Triple Store: Query Answering in OWL 2 Using an RL Reasoner.” In *Proceedings of the 22nd International Conference on World Wide Web (WWW 2013)*, ed. Ian Horrocks, 1569–1580. London: Elsevier, 2013.