# OBI Data Modelling Prototype

James A. Overton james@overton.ca

2013-06-17

This is a prototype of the "value specification" approach to modelling data in OBI/IAO. It is based on discussing during the Philly2013 workshop and on the mailing list, but it does not (yet) reflect a consensus.

James used the new OBI build tool to convert this document into OWL and test it. See: http://obi.svn.sourceforge.net/viewvc/obi/trunk/src/tools/build/

## Motivation

OBI's scope is biomedical investigations, and it must be able to describe the data generated in investigations. Some of the oldest terms in OBI and IAO were designed to describe measurement. More recently, we have recognized the need to describe predictions, simulations, and setting information. Superficially these are very similar to each other. "20g" could occur as:

1. a measurement of the mass of a particular mouse
2. a predicted mass of some future mouse after a treatment
3. an output of a simulation of mouse growth
4. a rule for selecting mice of a certain mass

Despite the superficial similarity, 1-4 are *about* very different things (if they're about anything at all):

1. a particular quality of a particular mouse at a particular time – something that clearly existed
2. a calculated value that does not correspond to any existing particular quality, but may be compared with a particular quality of a particular mouse in the future if the treatment is carried out
3. a calculated value, perhaps about what the mass of mice in general would be like under certain conditions
4. a directive, perhaps about part of a particular plan (now) to select mice in the future

OBI is a realist ontology, where we try to build consensus by being as precise as we can be about what is going on in the world. *Aboutness* is our primary way of distinguishing information content entities, and so we need to be clear in distinguishing measurements from predictions and settings, etc. This pushes our modelling toward greater complexity.

But we also want our modelling to be as simple as possible. We would like to *factor out* the similarities between 1-4, in order to reuse as much of our modelling as possible, and in order to reduce the number of asserted hierarchies we're dealing with.

In the proposal laid out here we distinguish the *structure* from the *content* of an information content entity. We assert a new hierarchy under "information structural entity" that contains "value specifications" such as the "20g" structure, which are not clearly *about* anything. Under "information content entity" we include entities that are clearly about things (in different ways), such a measurement data.

The proposal is designed to add just enough complexity to allow us to model measurements, predictions, and simulations in a similar way, by factoring out the shared structure of a "value specification".

Here are some notes on previous discussions:

- Bjoern's summary and proposal
- ICBO 2012 working session
- Christian's summary and proposal
- Philippe and Alejandra's modelling tests

# Upper Ontology

For clarity, we start from scratch rather than importing an existing ontology. Wherever possible, terms here have the same ID as in their source ontologies.

These are BFO classes and relations that we will need. NOTE: We don't include "dependent continuant", because it's not in BFO2 Graz and doesn't serve a purpose here.

```
Class: obo:BFO_0000001
  Annotations: rdfs:label "entity"

Class: obo:BFO_0000002
  Annotations: rdfs:label "continuant"
  SubClassOf: 'entity'

Class: obo:BFO_0000031
```

```
  Annotations: rdfs:label "generically dependent continuant"
  SubClassOf: 'continuant'

Class: obo:BFO_0000020
  Annotations: rdfs:label "specifically dependent continuant"
  SubClassOf: 'continuant'

Class: obo:BFO_0000004
  Annotations: rdfs:label "independent continuant"
  SubClassOf: 'continuant'

Class: obo:BFO_0000040
  Annotations: rdfs:label "material entity"
  SubClassOf: 'independent continuant'

Class: obo:BFO_0000019
  Annotations: rdfs:label "quality"
  SubClassOf: 'specifically dependent continuant'

ObjectProperty: obo:BFO_0000086
  Annotations: rdfs:label "has quality"

Class: obo:BFO_0000003
  Annotations: rdfs:label "occurrent"
  SubClassOf: 'entity'

Class: obo:BFO_0000015
  Annotations: rdfs:label "process"
  SubClassOf: 'occurrent'

ObjectProperty: obo:BFO_0000057
  Annotations: rdfs:label "participates in"
```

## Running Example

Our running example will involve measurements of a particular mouse named "Mickey" as part of a fictional investigation. Here we describe two universals and two particulars:

```
Class: obo:PATO_0000125
  Annotations: rdfs:label "mass"
  SubClassOf: 'quality'

Individual: mickey
```

```
  Types: 'material entity'
  Annotations: rdfs:label "Mickey",
    rdfs:comment "Mickey is a mouse."
  Facts: 'has quality' 'mass of Mickey'

Individual: mickey-mass
  Types: 'mass'
  Annotations: rdfs:label "mass of Mickey"
```

NOTE: In a previous draft of this document we used the determinable/determinate distinction to sub-class 'mass'. This made the modelling more complicated. Since a BFO OWL representation of determinable/determinate classes have not yet been discussed in detail, we decided to remove them from the current prototype.

- Barry's slides: http://ontology.buffalo.edu/bfo/2013/BFO-2-Smith.ppt
- Bare BFO2 issue: https://code.google.com/p/bfo/issues/detail?id=42

NOTE: We aren't modelling time in this prototype.

We'll focus first on scalar measurements of Mickey's mass. In later versions of this document we'll consider some other cases.

# Information Entities

Under "generically dependent continuant" we distinguish between information content entities (ICEs) that are *about* something, and information entities that are purely structural (ISEs) and not about anything. This is a new distinction put forward by Alan based on ongoing discussions with Barry and Werner.

```
ObjectProperty: obo:IAO_0000136
  Annotations: rdfs:label "is about"

Class: obo:IAO_0000030
  Annotations: rdfs:label "information content entity"
  SubClassOf: 'generically dependent continuant'
  SubClassOf: 'is about' some 'entity'

Class: information-structural-entity
  Annotations: rdfs:label "information structural entity"
  SubClassOf: 'generically dependent continuant'
```

In order to connect ISEs to ICEs we define a new relation. The label is just temporary.

```
ObjectProperty: has-information-structure
  Annotations: rdfs:label "has information structure"
  Domain: 'generically dependent continuant'
  Range: 'information structural entity'
```

### Units of Measurement

Under ISE we include "unit labels" to connect to the Units of Measurement Ontology, and we have an ObjectProperty and a DataProperty to use with them:

```
Class: obo:IAO_0000003
  Annotations: rdfs:label "unit label",
    rdfs:comment "was 'measurement unit label'"
  SubClassOf: 'information structural entity'

ObjectProperty: obo:IAO_0000039
  Annotations: rdfs:label "has unit label",
    rdfs:comment "was 'has measurement unit label'"
```

In OBI we have been modelling specific measurement units as OWL individuals. For our purposes we'll just need the SI unit "gram".

```
Class: obo:UO_0000002
  Annotations: rdfs:label "mass unit label"
  SubClassOf: 'unit label'

Individual: obo:UO_0000021
  Types: 'mass unit label'
  Annotations: rdfs:label "gram"
```

### Value Specifications

Also under ISE we have "value specification". The most important of these is "scalar value specification", which is the pair of a number and a unit.

```
Class: obo:IAO_0000601
  Annotations: rdfs:label "value specification"
  SubClassOf: 'information structural entity'

DataProperty: obo:IAO_0000004
  Annotations: rdfs:label "has value",
    rdfs:comment "was 'has measurement value'"
  Range: xsd:float
```

TODO: explain this relation

```
ObjectProperty: obo:IAO_0000605
  Annotations: rdfs:label "specifies value of"
  SubPropertyOf: 'is about'
```

Given the "unit label" asserted hierarchy, we can create a hierarchy of *defined classes* as needed:

```
Class: obo:IAO_0000602
  Annotations: rdfs:label "scalar value specification"
  EquivalentTo: 'value specification' and
    'has unit label' some 'unit label'

Class: scalar-mass-value-specification
  Annotations: rdfs:label "scalar mass value specification"
  EquivalentTo: 'value specification' and
    'has unit label' some 'mass unit label'
```

Here is an example of a particular scalar mass value specification, "20g":

```
Individual: 20g-specification
  Types: 'value specification'
  Annotations: rdfs:label "20g specification"
  Facts: 'has unit label' 'gram',
    'has value' "20"
```

Notice that we only assert that this is a 'value specification', and not that it is specifically a 'scalar mass value specification'. The reasoner will classify it correctly:

```
Fact: 20g specification is a scalar mass value specification
Query: 'scalar mass value specification'
Individuals: include '20g specification'
```

# Measurement Data

Our primary goal is to model measurement data well. We assert this class in OBI:

```
Class: obo:IAO_0000109
  Annotations: rdfs:label "measurement datum",
```

```
    rdfs:comment "We will also model the following:
      is_specified_output_of some
        ('data transformation' or prediction or 'information acquisition')"
  SubClassOf: 'information content entity'
```

Given the asserted hierarchy of value specifications, we can create a *defined hierarchy* of measurement classes.

```
Class: value-measurement-datum
  Annotations: rdfs:label "value measurement datum"
  EquivalentTo: 'measurement datum' and
    ('has information structure' some 'value specification')

Class: scalar-measurement-datum
  Annotations: rdfs:label "scalar measurement datum"
  EquivalentTo: 'measurement datum' and
    ('has information structure' some 'scalar value specification')

Class: scalar-mass-measurement-datum
  Annotations: rdfs:label "scalar mass measurement datum"
  EquivalentTo: 'measurement datum' and
    ('has information structure' some 'scalar mass value specification') and
    ('specifies value of' some 'mass')
```

We define an instance of a measurement datum about Mickey's mass by linking the information structural entity "20g" (using 'has information structure') to the particular mass quality being measured (using 'specifies value of'):

```
Individual: mass-20g-measurement
  Types: 'measurement datum'
  Annotations: rdfs:label "scalar measurement of mass of Mickey"
  Facts: 'has information structure' '20g specification',
    'specifies value of' 'mass of Mickey'
```

We just assert that this is a 'measurement datum', but the reasoner classifies it correctly as a 'scalar mass measurement datum':

```
Fact: mass measurement of Mickey is a scalar mass measurement datum
Query: 'scalar mass measurement datum'
Individuals: include 'scalar measurement of mass of Mickey'
```

# Queries

Modelling the data correctly is important, but we also need to query the data and get useful answers. First of all, we can query for all mass measurements:

```
Fact: mass measurement of Mickey is a mass measurement datum
Query: 'measurement datum' and ('specifies value of' some 'mass')
Individuals: include 'scalar measurement of mass of Mickey'
Subclasses: include 'scalar mass measurement datum'
```

We can create this defined class to capture all mass measurements:

```
Class: mass-measurement-datum
  Annotations: rdfs:label "mass measurement datum"
  EquivalentTo: 'measurement datum' and
    'specifies value of' some 'mass'

Fact: mass measurement of Mickey is a 'mass measurement datum'
Query: 'mass measurement datum'
Individuals: include 'scalar measurement of mass of Mickey'
Subclasses: include 'scalar mass measurement datum'
```

Using SPARQL we can get the subject, value, and units for mass measurements:

```
FACT get subject, value, and units for mass of Mickey measurement
PREFIX obo: <http://purl.obolibrary.org/obo/>
PREFIX : <http://purl.obolibrary.org/obo/obi/test.owl#>
SELECT ?subject ?value ?unit
WHERE {
  ?mass rdf:type obo:PATO_0000125 . # PATO mass
  ?subject obo:BFO_0000086 ?mass_instance . # BFO has quality
  ?measurement rdf:type obo:IAO_0000109 . # IAO measurement datum
  ?measurement obo:IAO_0000605 ?mass_instance . # IAO specifies value of
  ?measurement :has-information-structure ?spec .
  ?spec obo:IAO_0000039 ?unit . # IAO has unit label
  ?spec obo:IAO_0000004 ?value . # IAO has value
}
INCLUDE 'Mickey' "20" 'gram'
```

This query looks fairly complicated. However the only *added* complexity from using "value specification" is the 'has information structure' link:

```
  ?measurement :has-information-structure ?spec .
```

TODO: Add a Turtle example. Because Turtle has a nicer syntax for specifying anonymous entities, it might look cleaner than have OWL individuals for the measurement datum and the value specification.

## Measurement Processes

When modelling measurements, it's also important to be able to trace which processes produced which results. We'll extend the current model to include an assay that measures Mickey.

```
Class: obo:OBI_0000011
  Annotations: rdfs:label "planned process"
  SubClassOf: 'process'

Class: obo:OBI_0000070
  Annotations: rdfs:label "assay"
  SubClassOf: 'planned process'

ObjectProperty: obo:OBI_0000293
  Annotations: rdfs:label "has specified input"
  SubPropertyOf: 'participates in'

ObjectProperty: obo:OBI_0000299
  Annotations: rdfs:label "has specified output"
  SubPropertyOf: 'participates in'

Individual: assay-of-mass-of-mickey
  Types: 'assay'
  Annotations: rdfs:label "assay of mass of Mickey"
  Facts: 'has specified input' 'Mickey',
    'has specified output' 'scalar measurement of mass of Mickey'
```

Of course, this could be more complicated. The assay will involve some protocol, that we have not specified here. It might involve a device with a measurement function, and an investigation agent, etc. Here we show that it is simple to query for the assay, and then it should be straightforward to extend the query to other aspects of the assay.

Now we extend the query above to include the assay that generated the measurement datum:

```
FACT get the assay that measured the mass of Mickey
PREFIX obo: <http://purl.obolibrary.org/obo/>
PREFIX : <http://purl.obolibrary.org/obo/obi/test.owl#>
SELECT ?assay ?subject ?value ?unit
WHERE {
  ?mass rdf:type obo:PATO_0000125 . # PATO mass
  ?subject obo:BFO_0000086 ?mass_instance . # BFO has quality
  ?measurement rdf:type obo:IAO_0000109 . # IAO measurement datum
```

```
  ?measurement obo:IAO_0000605 ?mass_instance . # IAO specifies value of
  ?measurement :has-information-structure ?spec .
  ?spec obo:IAO_0000039 ?unit . # IAO has unit label
  ?spec obo:IAO_0000004 ?value . # IAO has value
  ?assay obo:OBI_0000299 ?measurement . # OBI has specified output
}
INCLUDE 'assay of mass of Mickey' 'Mickey' "20" 'gram'
```

We can query forward from Mickey to the results about him:

```
FACT get the results of assays with Mickey as input
PREFIX obo: <http://purl.obolibrary.org/obo/>
PREFIX : <http://purl.obolibrary.org/obo/obi/test.owl#>
SELECT ?value ?unit
WHERE {
  ?assay obo:OBI_0000293 :mickey . # OBI has specified input
  ?assay obo:OBI_0000299 ?measurement . # OBI has specified output
  ?measurement :has-information-structure ?spec .
  ?spec obo:IAO_0000039 ?unit . # IAO has unit label
  ?spec obo:IAO_0000004 ?value . # IAO has value
}
INCLUDE "20" 'gram'
```

We can query backward from a given measurement datum to the assay and its input:

```
FACT get the input of the assay that generated a particular measurement
PREFIX obo: <http://purl.obolibrary.org/obo/>
PREFIX : <http://purl.obolibrary.org/obo/obi/test.owl#>
SELECT ?assay ?input
WHERE {
  ?assay obo:OBI_0000299 :mass-20g-measurement . # OBI has specified output
  ?assay obo:OBI_0000293 ?input . # OBI has specified input
}
INCLUDE 'assay of mass of Mickey' 'Mickey'
```

If a value specification is unique, then we can query back from it to the assay and inputs:

```
FACT get the input of the assay that generated a particular value specification
PREFIX obo: <http://purl.obolibrary.org/obo/>
PREFIX : <http://purl.obolibrary.org/obo/obi/test.owl#>
SELECT ?assay ?input
WHERE {
```

```
  ?measurement :has-information-structure :20g-specification .
  ?assay obo:OBI_0000299 ?measurement . # OBI has specified output
  ?assay obo:OBI_0000293 ?input . # OBI has specified input
}
INCLUDE 'assay of mass of Mickey' 'Mickey'
```

However, we might want to reuse the same value specification for all our "20g" measurements (which is the approach used in this prototype), in which case this query might not return a unique assay and input. Or we might want to use an anonymous individual (i.e. blank node) for the value specification, which could also make the reverse query difficult. The best idea may be to focus on the measurement datum, then query back to the assay or investigation, and forward to the value specification.

# Predictions

If value specifications were only used for measurement data, then we could simplify this modelling by collapsing the distinction between them. But there are several cases other than measurement where we want to use a value specification. One such case is prediction.

We have just begun to include predictions in OBI, and the details are not yet settled. The goal here is to show one approach that works with value specifications. OBI has a term 'prediction' (OBI_0302910), but for the current purpose we define a more specific term. (For simplicity, we do not specify the input.) we also use a new term for the result of the process, and place it as a child of GDC, because the aboutness of the prediction is not clear. The labels are admitted ugly, but hopefully clear. The modelling parallels the assay above.

```
Class: testable-prediction-generation
  Annotations: rdfs:label "testable prediction generation"
  SubClassOf: 'planned process'
  SubClassOf: 'has specified output' some 'predicted measurement result'

Class: predicted-measurement-result
  Annotations: rdfs:label "predicted measurement result"
  SubClassOf: 'generically dependent continuant'
```

Once we assert 'predicted measurement result' we can define child classes just as we did with 'measurement datum':

```
Class: predicted-measurement-value
  Annotations: rdfs:label "predicted measurement value"
  EquivalentTo: 'predicted measurement result' and
```

```
    ('has information structure' some 'value specification')

Class: predicted-scalar-measurement-value
  Annotations: rdfs:label "predicted scalar measurement value"
  EquivalentTo: 'predicted measurement result' and
    ('has information structure' some 'scalar value specification')

Class: predicted-scalar-mass-measurement-value
  Annotations: rdfs:label "predicted scalar mass measurement value"
  EquivalentTo: 'predicted measurement result' and
    ('has information structure' some 'scalar mass value specification')
```

NOTE: Perhaps we could specify "and ('specifies value of' only 'mass')" for this class.

Then we can instantiate a prediction process and its result. We'll say that the prediction result is "20g", and reuse the same value specification:

```
Individual: prediction-of-mass-of-mice
  Types: 'testable prediction generation'
  Annotations: rdfs:label "testable prediction generation of mass of mice"
  Facts: 'has specified output' 'predicted mass of mice'

Individual: mass-20g-prediction
  Types: 'predicted measurement result'
  Annotations: rdfs:label "predicted mass of mice"
  Facts: 'has information structure' '20g specification'
```

We just assert that this is a 'predicted measurement result', but the reasoner classifies it correctly as a 'predicted scalar mass measurement value':

```
Fact: mass measurement of Mickey is a scalar mass measurement datum
Query: 'scalar mass measurement datum'
Individuals: include 'scalar measurement of mass of Mickey'
```

Using SPARQL we can get the value and units for the prediction. Instead of querying for measurements of mass, here we query for predictions that use mass units:

```
FACT get value and units for predictions of mass
PREFIX obo: <http://purl.obolibrary.org/obo/>
PREFIX : <http://purl.obolibrary.org/obo/obi/test.owl#>
SELECT ?prediction ?value ?unit
WHERE {
  ?unit rdf:type obo:UO_0000002 . # UO mass unit
```

```
  ?prediction rdf:type :predicted-measurement-result .
  ?prediction :has-information-structure ?spec .
  ?spec obo:IAO_0000039 ?unit . # IAO has unit label
  ?spec obo:IAO_0000004 ?value . # IAO has value
}
INCLUDE 'predicted mass of mice' "20" 'gram'
```

## Settings

Settings are another case distinct from measurements where we want to use value specifications. For our current purpose, a setting is part of the instructions for performing an investigation.

Our example will be a material acquisition process involving a selection rule for picking mice for the investigation. We'll assert that the selected mice must weigh less than 20g. First we need to include terms for plans and rules.

```
Class: obo:BFO_0000017
  Annotations: rdfs:label "realizable entity"
  SubClassOf: 'specifically dependent continuant'

ObjectProperty: obo:BFO_0000055
  Annotations: rdfs:label "realizes"
  Domain: 'process'
  Range: 'realizable entity'

ObjectProperty: obo:BFO_0000059
  Annotations: rdfs:label "concretizes"
  Domain: 'specifically dependent continuant'
  Range: 'generically dependent continuant'

Class: obo:OBI_0000260
  Annotations: rdfs:label "plan"
  SubClassOf: 'realizable entity'

Class: obo:IAO_0000033
  Annotations: rdfs:label "directive information entity"
  SubClassOf: 'information content entity'
```

NOTE: Directive information entity might be better as a direct child of GDC, but here we leave it under ICE.

```
Class: obo:OBI_0001755
  Annotations: rdfs:label "selection rule"
```

```
  SubClassOf: 'directive information entity'

Class: obo:OBI_0600008
  Annotations: rdfs:label "acquisition"
  SubClassOf: 'planned process'
  SubClassOf: 'realizes' some ('concretizes' some 'selection rule')

Class: obo:OBI_0600010
  Annotations: rdfs:label "material acquisition"
  SubClassOf: 'acquisition'
  SubClassOf: 'has specified output' some 'material entity'
```

Now we instantiate the material acquisition and the selection rule. The modelling of the rule is very basic, and will need to be improved, but serves the current purpose of connecting the rule to the value specification.

```
Individual: select-small-mice-rule
  Types: 'selection rule'
  Annotations: rdfs:label "select small mice rule",
    rdfs:comment "Select mice with mass less than 20g."
  Facts: 'has information structure' '20g specification'
```

NOTE: We have not provided axioms to constrain the rule to just mice. We have not provided axioms for the “less than” part of the rule. It might be better to say that the rule has a part that has the information structure, but the current modelling is simpler.

```
Individual: select-small-mice-plan
  Types: 'plan'
  Annotations: rdfs:label "select small mice plan"
  Facts: 'concretizes' 'select small mice rule'

Individual: acquisition-of-mice
  Types: 'material acquisition'
  Annotations: rdfs:label "acquisition of mice"
  Facts: 'realizes' 'select small mice plan'
```

Now we can query for processes that realize some concretization of a mass setting, and the value and units:

```
FACT get value and units for predictions of mass
PREFIX obo: <http://purl.obolibrary.org/obo/>
PREFIX : <http://purl.obolibrary.org/obo/obi/test.owl#>
SELECT ?process ?value ?unit
```

```
WHERE {
  ?unit rdf:type obo:UO_0000002 . # UO mass unit
  ?process obo:BFO_0000055 ?plan . # BFO realizes
  ?plan obo:BFO_0000059 ?setting . # BFO concretizes
  ?setting :has-information-structure ?spec .
  ?spec obo:IAO_0000039 ?unit . # IAO has unit label
  ?spec obo:IAO_0000004 ?value . # IAO has value
}
INCLUDE 'acquisition of mice' "20" 'gram'
```

TODO: It will be important to model ranges of values, including the signficant figures in scientific notation.

## Other Cases

TODO: Provide examples of categorical measurements and unstructured measurements.