Representing Phenotypes in OWL

Chris Mungall, Georgios Gkoutos, Nicole Washington and Suzanna Lewis

Abstract. Accurate representation of phenotypes using ontologies is important in biology and biomedicine. This paper describes the OWL translation of our methodology for representing phenotypes using ontologies in the OBO Foundry.

1 Introduction: On Phenotypes

In simple terms, a phenotype is a collection of characteristics that arise through the expression of the genes of an organism, in an environment. Some examples of phenotypes are the *red eyes* of a typical fruitfly, *Drosophila melanogaster*; the *length* of a body part such as the tail of a mouse; *high blood pressure* in the arteries of a human, *sensitivity* to a chemical or to light; *reduced mass* of an organ or part of an organism such as bone.

In biology, we are often interested in how variations in genotype and environment can lead to variations in phenotype. There is a wealth of research to draw from, but much of the knowledge is currently encoded in natural language and free text in the literature and biological databases[**?**] - if this knowledge can be captured using ontologies and ontological formalisms we increase the range of computational methods that can be used to analyse this data[**?**].

This paper describes the OWL translation of a formalism for representing phenotypes making use of the ontologies that comprise Open Bio-Ontologies Foundry (http://www.obofoundry.org). Many of these ontologies such as the Gene Ontology[**?**], the Foundational Model of Anatomy[**?**] and the Cell Ontology[**?**] represent *canonical* biology, which is to say the biology of "normal", "typical" or healthy organisms. We use these ontologies as building blocks that can be combined with an ontology of *qualities* to construct descriptions of phenotypes, many of which deviate from canonical biology. This methodology can be used in a variety of different kinds of investigations.

The reader of this paper should be aware that the end-user of the software systems and databases built around the formalism presented do not work directly at level of OWL constructs, or at their OBO-Formalism equivalents. They typically interact via intermediate representations and user interfaces such as Phenote[1] - more details can be found on the phenotype ontology website[2]. We present the OWL version of the formalism in order to give it a grounding in logic, and as a means of leveraging OWL-based tools and reasoning engines, although space dictates we do not discuss these applications.

---

[1] http://www.phenote.org

[2] http://www.phenotypeontology.org

1.1 Conventions Used

We use the Manchester OWL syntax [?] for writing our OWL representations[3]. Most of the OWL descriptions in this paper can also be translated to OBO-Format equivalents; we do not include these here for brevity, details can be found at `http://tinyurl.com/326feg` and the OboInOwl page[4].

Our examples are all drawn from OBO Foundry Ontologies[?] for the most part. OBO ontologies uniquely identify classes via a numeric identifier, but in the examples given we replace these with class labels in order to enhance readability. We do not always specify the ontology used in the examples given, the full list can be found on the OBO Foundry website[5].

2 OWL Representation

The biological and biomedical literature contains descriptions of phenotypes such as the following:

1. *red eye* (a typical or *wild-type* specimen of the fruitfly *Drosophila melanogaster* has red eyes.
2. *curvature* of a wing
3. *high blood pressure*
4. *high glucose concentration in the blood*
5. *short tail*
6. *invaginated cell membrane*

How should we go about representing phenotypes such as these in OWL? We will first review two existing W3C recommendations (instance and class partitions), and then describe our approach.

2.1 Enumeration of Individuals

The W3C Working Group note *Representing Specified Values in OWL: value partitions and value sets*[?] provides guidelines for two different ways of representing what it calls *specified values*: (1) enumerations of individuals and (2) partitions of classes. These "Specified Values" are applicable to organismal phenotypes.

The first method involves subdividing a quality such as *curvature* into owl **Individuals**, such as `flat` and `curved`, represented in Manchester Syntax as:

```
Class: CurvatureValue
EquivalentClass: {flat curved}
Individual: flat TYPE CurvatureValue
Individual: curved TYPE CurvatureValue
```

---

[3] http://www.co-ode.org/resources/reference/manchester_syntax/

[4] http://tinyurl.com/34ddl2

[5] http://obofoundry.org

We may also want to use *differentFrom* statements to state that `flat` and `curved` are distinct individuals:

```
Individual: flat
DifferentFrom: curved
```

We can relate individuals to color hues using an OWL functional property[6]:

```
FunctionalProperty: hasCurvature
Range: CurvatureValue
```

So an instance of a curved fruitfly wing[7] could be represented as:

```
Individual: fly-wing-00001 TYPE Wing
Facts:      hasCurvature VALUE curved
```

or the logically equivalent:

```
Individual: fly-wing-00001 TYPE (Wing THAT hasCurvature VALUE curved)
```

The W3C document notes some problems with this approach, such as the inability to further partition the values. For example, we could not create fiat partitions dividing *curved* into *highly curved*, *mildly curved*, etc[8]. This is a major disadvantage for representing phenotypes, as multiple levels of partitioning may be required to accurately represent the biology. On top of the listed problems, we would also add that this solution is ontologically unsatisfying, as we hold that these “values” correspond to *types* (also known as *universals*) and are instantiated multiple times in nature, and are best represented as owl *Classes*, not owl *Individuals*.

2.2 Value Partitions

The second method described in the W3C technical note involves partitioning qualities into owl *Classes*. We can write the same example in Manchester Syntax as follows:

```
Class: CurvatureValue
EquivalentClass: Flat OR Curved
```

This states that the class CurvatureValue is equivalent to the union of the 2 classes Flat and Curved - this also states the list is closed - there are no other classes that are also CurvatureValues. We could, however, create subclasses of Curved.

In addition we can specify disjointness conditions:

---

[6] for simplicity, we omit treatment of 3D objects that may have surfaces of different curvature, and assume any entity has a single curvature

[7] Represented in the Fly Anatomy Ontology `http://obo.sourceforge.net/cgi-bin/detail.cgi?fly_anatomy`

[8] in this example it may be possible to combine value partitions but this is inherently problematic

```
Disjoint: Flat Curved
```

The functional property linking curvature individuals to individuals with curvature is the same as for the enumeration pattern:

```
FunctionalProperty: hasCurvature
Range: CurvatureValue
```

Manchester Syntax even has a macro feature for conveniently representing value partitions:

```
ValuePartition: Curvature hasCurvature [Flat Curved]
```

This combines the unionOf axioms with the disjointWith axioms into a handy syntactic idiom.

So an instance of a curved fly wing could be represented as:

```
Individual: fly-wing-00001 TYPE (Wing THAT hasCurvature SOME Curved)
```

We can also name the class expression:

```
Class: CurvedWing
EquivalentClass: Wing THAT hasCurvature SOME Curved
```

It is important to note that an important consequence of this pattern is that the existence of each individual of type CurvedWing also entails the existence of a distinct quality individual of type Curved. This is in contrast to the enumeration pattern, in which there is a single quality individual `curved`, which is "shared" by all curved things.

This gives the class partition pattern various advantages over the enumeration of individuals pattern. It allows the quality classes to be further partitioned into more refined classes. We also hold that this pattern is ontologically preferable, and that distinct quality instances, such as my hair color or the shape of a particular organ, exist in reality.

2.3 Qualities and their Bearers

Our approach, which has been dubbed the "EQ" model, is highly similar to the class partition pattern described above, but differs in a few key respects. Our treatment is based on a previously published formal analysis of qualities[**?**], and uses an ontology of qualities called PATO[**?**][**?**].

In our formalisation, a phenotype is one or more *qualities* which are borne by *bearer entities* in some organism[9]. The relation between a quality and its bearer entity is one of *inherence*. An example of an instance of a phenotype is the particular instance of *high curvature inhering in* a particular instance of *wing* in a particular fruitfly.

---

[9] the phenotype of an organism is influenced by the genetics of that organism, mediated by the environment; description of these crucial factors are omitted for reasons of space

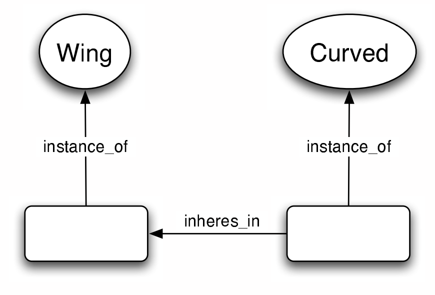

Fig. 1. The quality square: instances are depicted as rectangles, types as circles. Entity instances instantiate entity types (in this case, an eye). quality instances instantiate quality types (here the quality of curved, or high curvature). The quality instance inheres in the entity instance

. The instances are unnamed.

Note that sometimes qualities are referred to as *properties*, both colloquially and in the philosophical literature, but we avoid this usage, due to the contradictory use of the term *property* within the OWL community to denote what we would call a *relation*. Here we also distinguish between *instances*, the particular instances in reality, and the *individuals* that represent them.

All qualities are linked to their bearers by means of the inherence relation, which can be represented in OWL as:

```
FunctionalProperty: inheresIn
Domain: Quality
```

Note that this approach has no need of relations such as hasColor (although these could optionally be added as sub-relations of inheresIn). Rather than creating distinct class partitions for hasColor, hasShape and so on, we have a single subclass hierarchy, with all quality classes ultimately inheriting from the root *Quality* class.

We can describe the fly wild-type phenotype *high curvature wings* using the class expression:

A-1 *Curved* THAT *inheresIn* SOME Wing

We can also describe a *curved wing*:

A-2 *Wing* THAT *hasQuality* SOME Curved

These two expressions are intuitively similar, but crucially distinct, in that the first represents a *quality*, and the second represents the three dimensional object that is the *bearer* of the quality. The existence of an instance of one entails the existence of some instance of the other at any given time. Thus either class is eligible as the description of the phenotype. We choose A-1 as the “normal-form” for phenotypes.

Our approach is most consistent with the second approach in the W3C note above, as we use owl Classes to denote our Quality universals. However, we differ from this approach in that we have no specific need for multiple relations such as hasColor, hasSize, hasShape and so on. Our ontology of qualities is a single hierarchy, with all classes a subclass of the root class *Quality*.

3 Results

3.1 PATO: An Ontology of Qualities

To support the description of phenotypes, we have developed an ontology of qualities called PATO[10]. The ontology is constructed as far as possible according to OBO Foundry[11] principles, with each class in the ontology having a single *is_a* (subclass) parent. The ontology is maintained in OBO format and is available from OBO, but is also available in OWL[12].

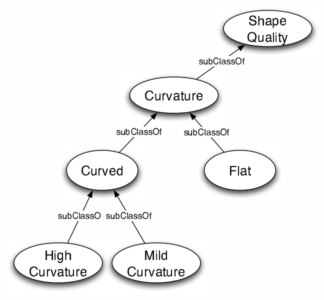

Fig. 2. Depiction of classes representing curvature in PATO

3.2 Granularity of Qualities

The ontology is divided along *granular levels* [**?**] [**?**]. On the one hand we have physical qualities, such as mass, velocity and color; next we have cellular qualities such as *ploidy* (which inheres in a cell or cell nucleus by virtue of the number of chromosomes it has) and *cellular potential* (capability of differentiating into a range of other cell types; finally we have *organismal qualities*. Note that of course cells and organisms have physical qualities such as mass, by virtue of the physical qualities of their subparts. Physics is sufficient to give an account of

---

[10] This originally stood for Phenotype and Trait Ontology. The full name is somewhat misleading as the classes in the ontology represent qualities, rather than full-blown phenotypes

[11] http://www.obofoundry.org

[12] http://purl.org/obo/owl/PATO

physical qualities - however, other qualities are emergent and a description must be given at a higher level of granularity.

Although the ontology is developed with the intent of use for phenotypes in biology and biomedicine, it is our hope that the physical quality sub-hierarchy of PATO will prove useful in other scientific realms.

3.3 Relational Qualities

We follow [?] in distinguishing between *monadic* and *relational* qualities. The former are qualities that inhere in and depend on a single entity. The latter depend on some additional entity or entity type. For example, shape is a monadic quality because shapes exist in totality in a single shape-bearing. A quality such as *sensitivity*, by contrast, has a single bearer but is always with respect to some other entity or kind of entity (such as in, for example, sensitivity to sunlight).

Neuhaus et al[?] represent relational qualities in a first-order logic framework using an inherence relation with a higher arity - the additional entity universals are referenced as additional arguments.

As all relations in OWL are binary, we must introduce an additional relation to represent relational qualities - we call this the “towards” relation.

Using this relation we can represent the phenotype *skin photosensitivity to UV light* using the class expression:

A-1 *Sensitive* THAT *towards* SOME UltravioletLight AND *inheresIn* SOME Skin

Note this is ontologically unsatisfactory, as any instance of photosensitivity is with respect to the *universal* UltravioletLight, rather than any one particular instance of a light wave or particle bundle. However, for the purposes of DL reasoning the representation is practical, and a reasoner such as Pellet will correctly reason that A-1 is subsumed by an expression such as A-2:

A-2 *Sensitive* THAT *towards* SOME Radiation AND *inheresIn* SOME Organ

We could instead use the UltravioletLight class as an individual, as in A-3:

A-3 *Sensitive* THAT *towards* VALUE UltravioletLight AND *inheresIn* SOME Skin

However, this has the undesirable consequence of taking us to OWL Full, and will result in omissions in the reasoning process.

3.4 Temporal Durations

Descriptions of phenotypes may be temporally qualified. The qualities in question may only inhere in the bearer entity over some interval - for example, organismal development involves changes in morphology of parts of organisms. Sometimes the quality may only be observed over a certain interval - for example, a particular assay of high blood pressure.

This raises interesting representation issues - all relations in OWL are binary, and there is no simple way to add a temporal index to binary instance-level relations, such as the relation between a quality and its bearer. There are a number of representation patterns for getting around this limitation [**?**] [**?**], such as reifying relations, but unfortunately they are all unsastifactory from an ontological and practical point of view.

Consider an portion of tissue, such as the *optic placode*[13], that changes its shape from flat to curved or invaginated during a specific stage of development. How do we represent this using binary relations?

We are unsatisfied with all the solutions afforded us within the constraints of binary relations - Our current compromise approach would be to represent this kind of anatomical entity as follows:

A-4 *OpticPlacode* THAT *hasQuality* SOME (Flat THAT *during* SOME Stage11) AND *hasQuality* SOME (Invaginated THAT *during* SOME Stage12)

(note that here we have an object-centered representation rather than quality-centered, using the hasQuality relation, the inverse of inheresIn)

This approach is unsatisfactory because it involves stating artefactual classes such as “flat during stage 11”. This approach also causes problems when reasoning, as illustrated below:

Assume also that Invaginated and Flat are disjoint (a reasonable assumption, given some fiat[**?**] quality boundary), and subclasses of Shape

```
Class: Invaginated
SubClassOf: Shape
DisjointFrom: Flat
```

Now we add the restriction that all organism parts have exactly one shape, as follows[14]:

```
Class: OrganismPart
SubClassOf: hasQuality EXACTLY 1 Shape
```

However, if flat and invaginated are disjoint classes, and we declare that each organism part can only assume a single shape, this will result A-4 being equivalent to the empty set, an undesirable result.

An alternate representational paradigm involves representing “time-slices” of three dimensional objects (such as an organism part) at different times. These slices can then be related via binary relations to qualities that obtain over intervals. Each slice can then be linked back to the representation of the three dimensional object. This approach is sometimes called the *perdurantist* or *4-dimensionalist* approach, and the time-slices are sometimes called *histories* or *snapshots*.

---

[13] the part of an organism that will develop into the eye

[14] the intent is to specify a Qualified Cardinality Constraint, as is allowed in OWL-1.1, but the authors are not sure if this is the correct for in the Manchester syntax

For example, we can represent an optic placode whose shape changes between flat and invaginated during a stage of development using the following class expression:

A-5 *OpticPlacode* THAT *hasTimeSlice* SOME *(Slice* THAT *hasQuality* SOME Flat AND *during* Stage11) AND hasTimeSlice SOME (Slice THAT *hasQuality* SOME Invaginated AND *during* Stage12)

However, this solution is unsatisfactory from both an ontological and a practical perspective. On the one hand we must introduce additional artefacts, that have no place in upper ontologies such as BFO[**?**] and DOLCE. Our commitment to OBO Foundry principles leads us to reject the existence of unusual entities such as time-slices. In addition, the time slice paradigm must be anticipated in all ontologies used, such that for example the domain and range of relations can be 3D objects or time-slices of 3D objects.

At best, the above construction is unwieldy and will require specific tool support if these constructs are to be generally useful.

Another approach is to reify the relation in question, in this case *inheresIn* , modeling it with an OWL class rather than an OWL property. We omit a discussion of this approach for reasons of space, but we find this solution even more unsatisfactory.

An extension to OWL allowing would simplify this problem immensely. For example, if relations could take a third temporal argument as illustrated in the following expression:

A-6 *Organism* THAT *(hasQuality* SOME *Flat* AT *Stage11)* AND *(hasQuality* SOME *Invaginated* AT *Stage12)*

The proposed `AT` operator could be used to index any relation at the class or instance level in OWL. We recognise that temporal extensions to DL expressivity would introduce many challenging problems, but we believe that this issue must be addressed sooner rather than later in order to accurately represent biology.

One possibility is for 3-ary temporal expression such as the above to be "compiled down" to standard binary relation oriented description logics bu automatically introducing artefactual entities such as time-slices "behind the scenes".

4 Discussion

Representation of and reasoning over phenotypes is of huge importance to the life sciences and biomedicine. Our methodology for describing phenotypes through the composition of *quality* classes with classes of *quality-bearers* via the *inherence* relation works well for a variety of phenotypes, and promotes the reuse of ontologies in a modular way. The ontology of qualities, PATO, can be used across a variety of organisms for phenotypes across multiple levels of granularity.

We find the ability to compose classes using description logic type expressions useful, and is a representational technique the general life sciences and biomedical informatics and database communities would find useful in general. In particular, the elegance of Manchester OWL Syntax renders complex ontological representations accessible to ordinary mortals[15]. This is a significant result for taking OWL out of the ivory tower into the real world. To illustrate this, here is a class expression involving multiple levels of nesting utilising classes from multiple ontologies to represent the phenotype *high permeability of mitochondrial cristae in the axons of pyramidal cells in the CA1 hippocampal field*:

A-1 *HighPermeability* THAT *inheresIn* SOME *(MitochondrialCristae* THAT *partOf* SOME (Axon THAT partOf SOME PyramidalCell THAT *partOf* SOME CA1Field))

The ability to attach these class expressions to named phenotypes, such as those that may be found in the Mammalian Phenotype Ontology[**?**] and the Plant Trait Ontology[**?**] will be important for comparing data across organisms, and we have commenced an effort to create these expressions[16].

We have found that limiting expressions to binary relations has negative implications for ontologically sound and intuitive representations of biology, particularly with respect to time.

For lack of space we did not address various other important topics such as the representation of experimental assays and quantitative measurements. We also avoided discussion of representation on the complex relationship between genotypes, environments and phenotypes. However, we note that the methodology employed in this paper for representing phenotypes is also applicable to representing environment types.

5 Acknowledgements

The ideas in this paper arose through discussions with many individuals, including Fabian Neuhaus, Barry Smith, Alan Rector, Nigam Shah, William Bug, Mark Gibson, John Day-Richter, Michael Ashburner and the members of the FlyBase consortium, Monte Westerfield and the members of ZFIN Consortium, Pankaj Jaiswal and the participants on the obo-phenotype mailing list.

References

1. M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock. Gene ontology: tool for the unification of biology. the gene ontology consortium. *Nat Genet*, 25(1):25–9, 2000. 1061-4036 Journal Article.

---

[15] of course, the end-user does not see Manchester Syntax or any other OWL-oriented syntax, they work with intermediate representations and user interfaces such as Phenote

[16] http://tinyurl.com/2o7fnh

2. M. Ashburner, C. J. Mungall, and S. E. Lewis. Ontologies for biologists: a community model for the annotation of genomic data. *Cold Spring Harb Symp Quant Biol*, 68:227–35, 2003. 0091-7451 Journal Article Review Review, Tutorial.
3. J. Bard, S.Y. Rhee, and M. Ashburner. An ontology for cell types. *Genome Biology*, Manuscript submitted, 2004.
4. R. Drysdale. Phenotypic data in flybase. *Brief Bioinform*, 2(1):68–80, 2001. 1467-5463 Journal Article.
5. G. V. Gkoutos, E. C. Green, A. M. Mallon, J. M. Hancock, and D. Davidson. Building mouse phenotype ontologies. *Pac Symp Biocomput*, pages 178–89, 2004. Journal Article.
6. G. V. Gkoutos, E. C. Green, A. M. Mallon, J. M. Hancock, and D. Davidson. Using ontologies to describe mouse phenotypes. *Genome Biol*, 6(1):R8, 2005. 1465-6914 Journal Article.
7. Pierre Grenon. Temporal qualification and change with first-order binary predicates. In *International Conference on Formal Ontology in Information Systems (FOIS 2006)*, Baltimore, Maryland (USA), 2006.
8. Pierre Grenon, Barry Smith, and Louis Goldberg. Biodynamic ontology: applying BFO in the biomedical domain. *Stud Health Technol Inform*, 102:20–38, 2004.
9. Purvesh Khatri and Sorin Draghici. Ontological analysis of gene expression data: current tools, limitations, and open problems. *Bioinformatics*, 21(18):3587–3595, Sep 2005.
10. Anand Kumar, Barry Smith, and Daniel D. Novotny. Biomedical informatics and granularity: Conference papers. *Comp. Funct. Genomics*, 5(6-7):501–508, 2004.
11. M.Horridge, N.Drummond, J.Goodwin, A.Rector, R.Stevens, and H.Wan. The manchester owl syntax. *OWL: Experience and Directions 2006*, 2006.
12. F. Neuhaus, P. Grenon, and B. Smith. A formal theory of substances, qualities, and universals. In A. Varzi and L. Vieu, editors, *Formal Ontology in Information Systems (FOIS?04)*, page 49?59, Turin, Italy, 2004. IOS Press.
13. Alan Rector. Representing specified values in OWL: “value partitions” and “value sets”. W3C note, W3C, May 2005. http://www.w3.org/TR/2005/NOTE-swbp-specified-values-20050517.
14. Alan Rector and Natasha Noy. Defining n-ary relations on the semantic web. W3C note, W3C, April 2006. http://www.w3.org/TR/2006/NOTE-swbp-n-aryRelations-20060412/.
15. Alan Rector, Jeremy Rogers, and Thomas Bittner. Granularity, scale and collectivity: when size does and does not matter. *J. of Biomedical Informatics*, 39(3):333–349, 2006.
16. C. Rosse and J.L.V. Mejino. A reference ontology for bioinformatics: The foundational model of anatomy. *Journal of Biomedical Informatics 36:478-500.*
17. Barry Smith and Achille C. Varzi. Fiat and bona fide boundaries: Towards on ontology of spatially extended objects. In *Spatial Information Theory*, pages 103–119, 1997.
18. Cynthia L Smith, Carroll-Ann W Goldsmith, and Janan T Eppig. The Mammalian Phenotype Ontology as a tool for annotating, analyzing and comparing phenotypic information. *Genome Biol*, 6(1):R7, 2005.
19. Yukiko Yamazaki and Pankaj Jaiswal. Biological ontologies in rice databases. An introduction to the activities in Gramene and Oryzabase. *Plant Cell Physiol*, 46(1):63–68, Jan 2005.