SGDI ontology development

VJ Carey stvjc@channing.harvard.edu

April 30, 2008

1 Introduction

This document describes tasks related to ontology development for the SGDI (software for genomic data integration) project. Principal concerns include

- establishment of conventions for describing designs of and samples from microarray experiments;
- establishment of software tools that help implement these conventions;
- maximizing reuse of programatically accessible vocabulary resources, such as those provided by caCORE/EVS;
- employing appropriate standards for metadata structure design and deployment, such as RDF/OWL models and associated XML serializations.

2 Implementation issues

We will distinguish three basic information structures:

- **provenanceStruct:** a container for information regarding the source and maintenance of ontology-related information;
- **nomenclature:** a set of tokens representing terms or concepts, with specified provenance and definitions;
- **ontology:** a nomenclature with a hierarchical structure reflecting semantic relations among the terms.

2.1 Nomenclature example

An example of a nomenclature structure is given here:

```
GDI nomenclature:  KEGGPATH
provenance: source =  ftp://ftp.genome.ad.jp/pub/kegg/pathways/map_title.tab
          : date =  June 30 2004
There are 334 terms.
First 5 terms/definitions
     terms   defs
[1,] "00627" "1,4-Dichlorobenzene degradatio.."
[2,] "00628" "Fluorene degradation"
[3,] "00980" "Metabolism of xenobiotics by c.."
[4,] "00629" "Carbazole degradation"
[5,] "00981" "Insect hormone biosynthesis"
GDI maps:
[1] "LL2KEGGmap.hsa" "LL2KEGGmap.rno"
```

The information on "GDI maps" pertains to usage of the nomenclature in other information structures. In general it will be important to catalog not only the terms and vocabularies in which they are used, but also the substantive data resources in which these vocabulary resources are deployed. For example, in cross-organism inference, it will be useful to be able to identify which data resources use KEGG identifiers to characterize genomic components or gene products. One can then iterate over only these resources, searching for the presence of a given set of KEGG identifiers.

2.2 Ontology example

An example of an ontology is given here. We use the class *taggedHierNomenclature* to emphasize

1. the existence of tags, which are abbreviated tokens that are typically semantically opaque, used for abbreviated reference to terms of interest;
2. the existence of hierarchical semantic relationships among terms;
3. the extension of the formal nomenclature class.

The data resource in this example is the NCI *Thesaurus*, as opposed to the *Metathesaurus*. The thesaurus is made available to support ontological inference in ways that are not straightforward for the metathesaurus at this time.

```
> data(GDI_NCIThesaurus)
> GDI_NCIThesaurus
```

```
GDI hierarchical nomenclature with tags.
First 10 tags:  C191 C192 C193 C194 C195 C196 C197 C198 C199 C200
GDI nomenclature:  NCIThesaurus
provenance: source =  ftp://ftp1.nci.nih.gov/pub/cacore/EVS/Thesaurus_04.06i.FLAT.zip
          : date =  Oct 15 2004
There are 36246 terms.
First 5 terms/definitions
     terms                       defs
[1,] "Combination_Chemotherapy"  "Drug therapy with two or more .."
[2,] "Mesna"                     "2-mercaptoethanesulfonic acid .."
[3,] "_3-Deazauridine"           NA
[4,] "_4-Nitroquinoline-1-Oxide" NA
[5,] "Mercaptopurine"            "An anticancer drug that belong.."
GDI maps:
character(0)
```

There are helper functions that navigate the ontology; at present a true graph is not employed.

```
> mpar <- parents("Mesna", GDI_NCIThesaurus)
> mpar

[1] "Uroprotective_Agent"

> children(mpar, GDI_NCIThesaurus)

[1] "Mesna"   "Dimesna"
```

General regular expression matching can be used:

```
> substring(grep("HER-2", GDI_NCIThesaurus), 1, 70)

                                                                C2215
                          "HER-2_neu_Intracellular_Domain_Peptide_Vaccine"
                                                                C2537
                                              "HER-2_neu_Peptide_Vaccine"
                                                                C2593
                                   "HER-2_neu_Intracellular_Domain_Protein"
                                                                C2637
                                      "MVF-HER-2_628-647-CRL_1005_Vaccine"
                                                                C2658
                 "Anti-CD3_x_anti-HER-2-neu_Bispecific_Monoclonal_Antibody"
                                                               C11686
                             "HER-2-neu-Peptide-Vaccine_Montanide-ISA-51"
```

```
                                                                C11702
                                    "HER-2-Neu-Peptide-Vaccine_Sargramostim"
                                                                C12095
"Anti-CD3-x-Anti-HER-2-neu-BiSpecific-Monoclonal-Antibody_Autologous-Ly"
```

When definitions are present, they can be obtained:

```
> getDefs("Mesna", GDI_NCIThesaurus)

[1] "2-mercaptoethanesulfonic acid sodium salt; scavenges glutathione; used as a detoxi
```

3 Building a new ontology

A workflow for building a new ontology is not clearly established at present, but the basic tasks appear to be

1. determine a set of concepts and associated terms;
2. examine existing ontologies for coverage of the terms of interest;
3. if the application can proceed on the basis of the harvesting of a single pre-existing ontology, it may suffice to build a *taggedHierNomenclature* instance based on this ontology;
4. if the application requires a separate ontology or desired concepts are not adequately covered,
   (a) construct a new OWL model for the ontology using Protege;
   (b) deserialize the OWL to an *ontModel* using Rswub;
   (c) create a *taggedHierNomenclature* instance on the basis of lists derivable from the *ontModel* instance.

We'll illustrate this process using some terms related to breast cancer identified by Sridhar. We'll work backwards from a data structure, `SGDIvocab`, currently in *ontoTools* Sridhar looked the terms up in the NCI Metathesaurus and determined that they were covered in some sense, but he did not provide the exact entry matching the intended concept. The terms are

```
[1] "Epithelial_gene_cluster" "ER_Clinical"
[3] "ER"                      "ESR1_Array"
[5] "Differentiation"         "Erb-B2_Clinical"
[7] "PR"                      "HER-2"
[9] "Erb-B2_Array"
```

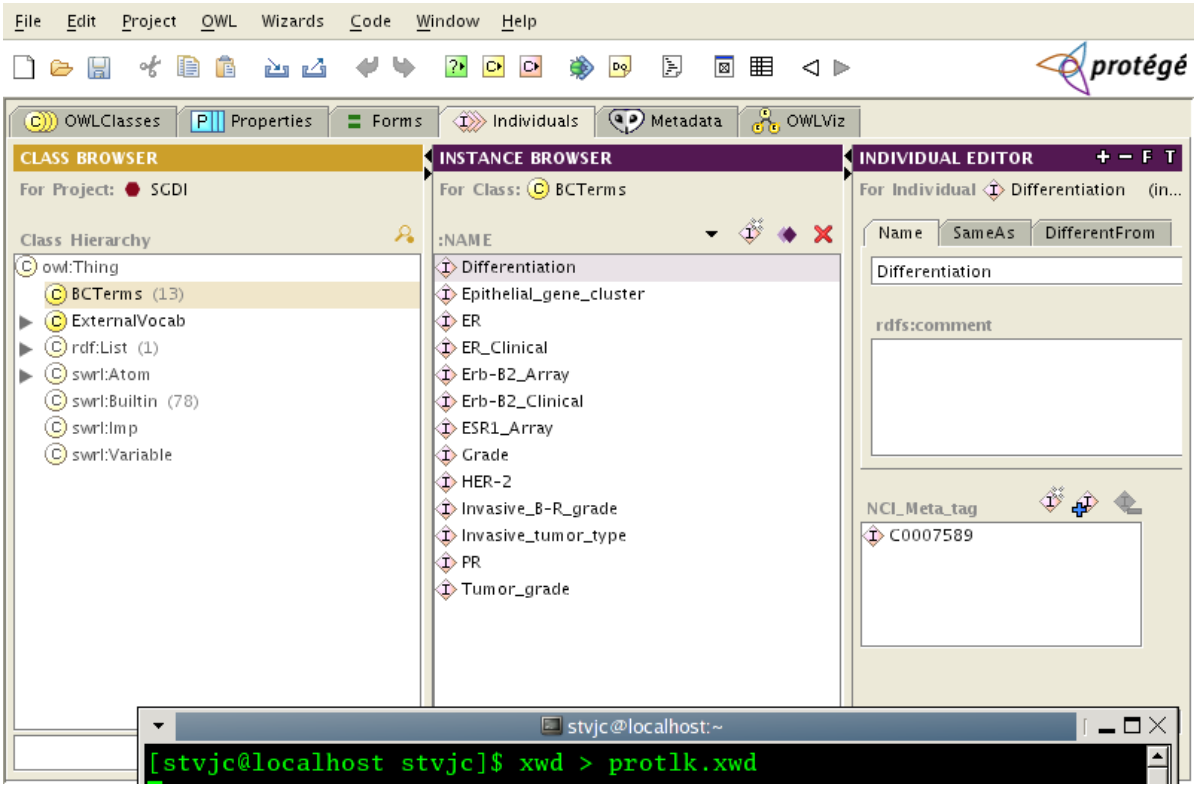

Figure 1: View of protege ontology editor for SGDI vocabulary. Focus on BCTerms.

The exact meanings of these terms is not completely clear. The use of the `_array` suffix has no conventional interpretation that I am aware of; likewise the `_clinical` suffix. We may need to invent new terms and definitions to clarify the intended meaning of these tokens.

We proceed provisionally; the resulting tools may be modified at any time in the future as usage patterns emerge.

3.1 Protege-based management of terms and structure

Figure 1 shows the Protege ontology editor in use to define the BCTerms class. There are 13 instances. Note to the right of the display that there are a variety of fields to be defined, including an RDFS comment field, and an `NCI_Meta_tag` field. Figure 2 shows the editor focused on the formal tags provided by NCI Metathesaurus for terms semantically similar (by informal matching) to the BCTerms of interest.

3.2 Importation of OWL model

The Rswub package is still not ready for distribution, but I will convert OWL to R structures as needed. The ontology model is prescribed by the Jena system from HP.

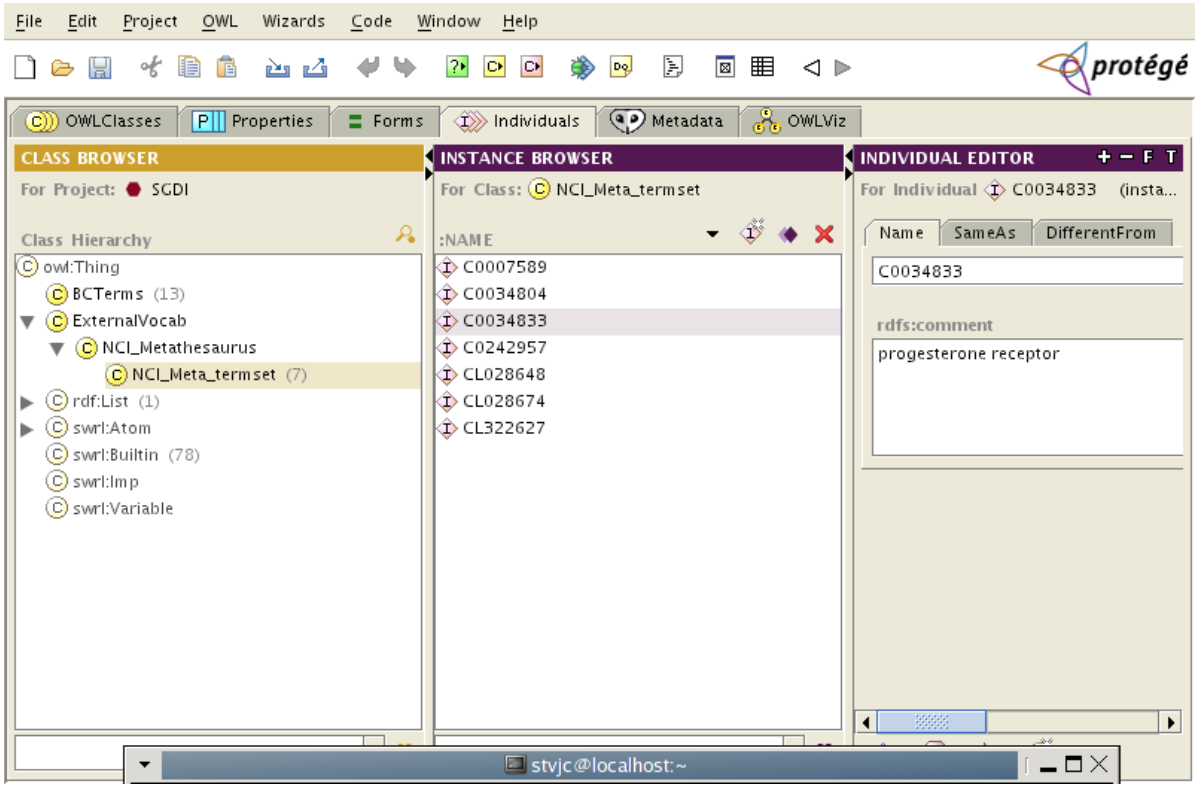

Figure 2: View of protege ontology editor for SGDI vocabulary. Focus on NCI metatags.

readSWModel returns an anonymous omegahat reference to a Java class instance.

```
> library(Rswub)

> omod <- readSWModel("http://www.biostat.harvard.edu/~carey/SGDI.owl",
+     asURL = TRUE)

> omod

Ontology model (instance of com.hp.hpl.jena.ontology.impl.OntModelImpl )
source: http://www.biostat.harvard.edu/~carey/SGDI.owl
There are 4 named classes.
Base namespace: http://www.owl-ontologies.com/unnamed.owl .
```

The Java-based ontology model object here is completely independent of the tagged hierarchical nomenclature structures which are implemented purely in R. For our purposes, the omod object is just a bridge from OWL to R. omod can be interrogated for the underlying RDF model. This is just a set of triples (subject, predicate, object), and getSplits returns a list of two elements: bypred and bysub.

```
> somod <- getSplits(omod)
> names(somod)
```

We have defined NCI_Meta_tag as a property with domain BCTerms and range NCI_Meta_Termset.

```
> somod$bysub$NCI_Meta_tag
```

The value of the property for each BCTerm instance is:

```
> somod$bypred$NCI_Meta_tag
```

This table is the basis of the tagged nomenclature that we need in the SGDI ontology. The "subj" values are the tokens we are interested in. The "obj" values are the formal identifiers of the NCI Metathesaurus entries that we want to map our tokens to. When CACore is working, we will use these formal identifiers to retrieve detailed definitions, focused keyword matches, etc.

3.3 The nomenclature

```
> data(SGDIvocab)
> SGDIvocab

GDI hierarchical nomenclature with tags.
First 10 tags:  NCIMET:CL028674 NCIMET:C0034804 NCIMET:C0034804 NCIMET:CL322627 NCIMET:
GDI nomenclature:  SGDI
provenance: source =  http://www.biostat.harvard.edu/~carey/sgdi.owl
```

```
          : date =  12/21/2004
There are 9 terms.
First 5 terms/definitions
     terms                      defs
[1,] "Epithelial_gene_cluster" NA
[2,] "ER_Clinical"             NA
[3,] "ER"                      NA
[4,] "ESR1_Array"              NA
[5,] "Differentiation"         NA
GDI maps:
character(0)
```

4 Summary and future work

We have defined a few classes representing nomenclatures and hierarchical nomenclatures. Populating instances of these classes with existing genomic information was illustrated with KEGG and NCI Thesaurus (not metathesaurus!). Search and navigation of these structures is supported, but additional facilities will be required as the workflow clarifies.

When a specific collection of clinical and technical terms is identified, we propose to formally manage the collection using Protege to define an OWL/RDF model. The model can be deserialized to Java and R structures using Rswub, which will be placed in Bioc development in the near future.

The OWL/RDF model has a regimented graphical structure. It is a collection of linked ordered triples with interpretation (subject, predicate, object). When OWL objectProperty status is conferred on an entity (property term), a domain and range can be defined for that property. A benefit of formal specification of domain and range is that misuse of the property can be programmatically forbidden. In our example, the `NCI_Meta_tag` property maps from the BCTerms set to an instance of the `NCI_Meta_termset`. If we ask for the `NCI_Meta_tag` property of any BCTerm instance, we are guaranteed to receive an instance of `NCI_Meta_termset`. An instance of the `NCI_Meta_termset` is composed of a formal NCI Metathesaurus alphanumeric tag, and a brief verbal definition. Ultimately the tag will be used in programmatic interrogation of the metathesaurus for ancillary information such as detailed definition, links to defining and illustrative documents, and so forth.

Use of the OWL technology conveys access to techniques for vocabulary harmonization. Schemas may be defined that establish equivalent properties with different names, or set-theoretically defined combinations of formal classes, and models may be revised using formal inference based on the schema. Logical inference within the OWL model is also supported.