A FLEXIBLE FRAMEWORK FOR WEB INTERFACES TO IMAGE DATABASES: SUPPORTING USER-DEFINED ONTOLOGIES AND LINKS TO EXTERNAL DATABASES

Josiah Johnston, Arpun Nagaraja, Harry Hochheiser, Ilya Goldberg

Image Informatics and Computational Biology Unit, Laboratory of Genetics, IRP/NIA/NIH
Suite 3000, 333 Cassell Dr, Baltimore, MD 21224. siah@nih.gov, arpun@mit.edu, {hsh, igg}@ nih.gov

ABSTRACT

Vocabularies to describe research findings are needed to effectively use scientific databases. As understanding of data evolves, scientists need tools for extending the vocabularies. A web-based framework addresses these issues in the context of a microscopy image database. It allows definition of simple vocabularies and automatic generation of tools for annotating, displaying, and searching with this vocabulary. The resulting displays of annotated data can be linked to external data sources such as model organism databases. A case study involving publication of in-situ hybridization images cross-linked with the Mouse Gene Index is described, along with preliminary steps in extending the framework to handle fully structured ontologies.

1. INTRODUCTION

1.1. The Open Microscopy Environment (OME)

OME is a data repository and analysis system for light microscopy images [1]. OME specifies an ontology, or well-structured data model, for microscopy images and acquisition data. OME's ontology can be extended through the definition of *semantic types*. OME software is open source, and is available for download at www.openmicroscopy.org. It imports a wide variety of microscopy image formats. Data is stored in a relational database, and accessed through a web interface or other client. For more details, see [2].

1.2. Ontologies

Manual categorization is typically a first step in image analysis. Given a set of images from an experiment, a scientist might want to (for example), describe each image as either being a control or as belonging to one of several alternative experimental conditions. As these groupings are generally not known when software is written, users must be able to define them easily.

Unfortunately, fully-specified ontologies are difficult for casual users to define and use. Ontology languages tend to emphasize flexibility and expressiveness rather than simplicity. Despite this, specification of ontologies still often outpaces development of end-user tools to work with them.

An alternative approach is to provide simple tools for enumerating distinct experimental variables and observations that can be associated with images (Figure 1). While these enumerations do not possess the modeling rigor and completeness associated with formal ontologies, their ease of use makes them more accessible to a wider range of users. This approach has proven successful in so-called "folksonomies", which support lightweight, ad-hoc annotations [3]. Using enumerations defined through a simple web interface, OME's toolkit generates an end-user toolset for annotating, displaying, or searching (ADS) images. These tools can then be customized to enhance usability and features.

1.3. Integration with External Resources

Integration with model organism databases (MODs), genomes browsers, and other external resources can provide further value by placing experimental terms in context. Our specific use case involves a large collection of *in situ* hybridization experiments to determine the expression patterns of mouse genes in embryos and embryonic stem-cells. The goal is to implement an interface for the bench scientist doing the experiments as well as a separate similar interface for public access. In addition, these interfaces should link out to a separate genomics database, such as the NIA Mouse Gene Index (MGI), and conversely, the public interface should be accessible from links in the MGI [4].

2. UNDERLYING WEB ARCHITECTURE

The ADS architecture evolved from the OME web interface. It provides the core set of functionality for individual vocabulary elements. Each component of ADS functionality is implemented by an engine. The behavior of the engine and the resulting UI can be customized using an HTML template or by subclassing the engine. Templates specify

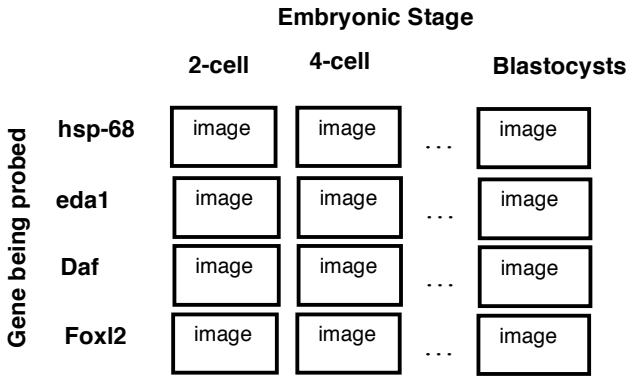

Figure 1: Conceptual view of experimental variables associated with MGI. The set of enumerations is "Embryonic Stage" and "Gene being probed." A third vocabulary element to describe observations (e.g. "Localization of expression") is not shown.

layout, allow injection of links, and request data. Subclasses allow injection of custom code. In all cases, both template and subclass extensions are optional, and generic templates and subclasses are used by default.

2.1. Annotation

The annotate engine allows creation of one object of any type (an instance of a declared OME Semantic Type or ST). Template extensions allow further formatting. For example, the template defined for the Category Group ST was adjusted to decrease the size of the 'name' data entry field relative to 'description'. Because parameters to the annotate engine can be specified in the URL, it is easy to generate incoming links that populate certain fields. The Category Group display page (figure 2) makes use of this to generate a "Add New Category" link.

Base url to create a new object:
http://localhost/perl2/serve.pl?Page=OME::Web::DBObjCreate

Example: Create a category
…&Type=@Category

Example: Create a category belonging to a Category Group with id 123
…&Type=@Category&CategoryGroup=123

2.2. Display

The display engine comes in two parts. One is accessed through a backend API, and the other is exposed to web browsers. The backend API provides a simple rendering interface. Subclasses of this engine allow injection of code for complex data retrieval or custom manipulation. For example, the Image class has a template that requests a 'thumbnail_url' field. The display engine has a subclass specific for the Image class that derives this url from various parts of the database, and populates that field in the HTML resulting from the template. The Image display subclass also generates links to the annotation engine using a similar mechanism. The display engine sub-class specific for the Dataset object includes control logic to add or remove images.

Example: Render an image as a summary
my $html_snippet = $renderer->render($image, 'summary');

Example: Render an array of images as a list of references
my $html_snippet = $renderer->renderArray($image_list, 'ref_list');

The exposed part of the display engine displays a single object at a time, with each object optionally specifying an object-specific template and an object-specific sub-class of the display engine.

Example: Display a detail view for a single single object:

Base url:
http://localhost/perl2/serve.pl?Page=OME::Web::DBObjDetail

Display a class describing a microscope objective with id 123:
…&Type=@Objective&ID=123

2.3. Search

The search engine allows searches for any type of object. The exposed search fields can be adjusted by changing the underlying template for a given object. Subclasses can specify a default value for search fields of that type (e.g. search for images belonging to the logged in user). So far, that has been the only functionality needing specialization.

The search engine can also be used to select objects. A javascript function that accepts an object type and field name will open a popup window the user can use for selecting objects. By passing an additional url parameter ("Select") to the seach engine, it shifts its functionality to search and select. Clicking the select button updates the calling window, and submits its form.

Base url:
http://localhost/perl2/serve.pl?Page=OME::Web::Search

Example: Search for a Category Group
…&Type=@CategoryGroup

Example: Select one Category Group
…&Type=@CategoryGroup&Select=one

2.4. Summary

These components are interdependent. Figure 2 illustrates how a web page displaying a Category Group links to the Create and Search pages, and recursively uses the Display engine to show Categories within the group. The create and search pages also make use of the other ADS components.

Each component in this ADS set is centered on a single semantic type. To develop domain-specific applications, we need ADS for combinations of types. This motivated development of the enumeration ADS toolkit.

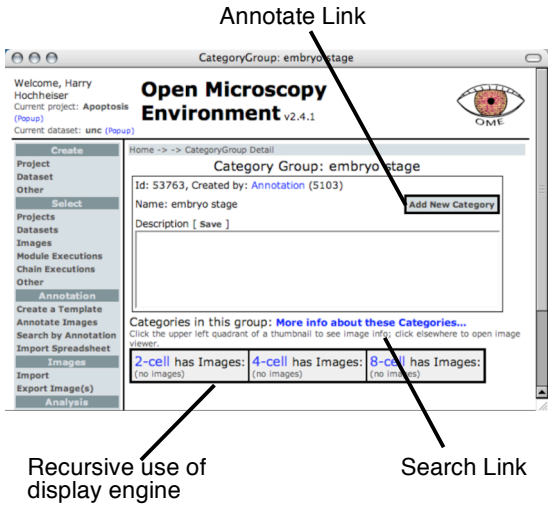

Figure 2: An example of the user interface for the "CategoryGroup" ST, using a CategoryGroup-specific template and display engine.

3. Enumeration toolkit applied to the Mouse Gene Index (MGI)

The enumeration toolkit provides extensible, auto-generated Annotation, Display, and Search services for sets of enumerations. Its use is illustrated below with a description of the setup procedure for the MGI project. Detailed instructions of these steps are available online. [5]

3.1. Setup

The first step is to define a vocabulary. For clarity, we limit the association of images to two enumerations: Gene and Embryo Stage. There are thousands of gene names, so they are imported from spreadsheets to minimize typing errors. Six embryo stages are being considered, so they are defined using the annotation tools described in 2.1 to create instances of the "Embryo Stage" CategoryGroup, and one instance of Category for each of the specific stages: 2-cell, 4-cell, 8-cell, Morula, and Blastocyst.

The next step is to generate a domain-specific toolset. This entails selecting the set of enumerations (in our case the "Targeted Gene" and "Embryo Stage" Category Groups), and specifying a name for our toolset (in our case, "MGI").

3.2. Annotation

Annotation can be accomplished in two ways. Pre-existing annotations performed outside of OME can be imported from spreadsheets. Images lacking those can be annotated using the tool generated in 3.1. The annotation tool requires the user to select images to annotate then records annotations for the images one-by-one. The annotation tool remembers the settings for the previous image in order to rapidly annotate the selected images with a minimum of user interaction. During the annotation process, the user can also define new terms for each enumeration. This is useful when constructing a list of phenotypes from observations, for example, or when the set of terms is not completely specified during setup.

3.3. Search

The search page contains a list of genes and a list of embryonic stages. The lists are html form elements that allow the user to select one or more items from each list. Selections from these lists displays all images matching all selected criteria. The mouse gene index would link to a search term on this page by appending the category group and category to the base url. Example given below.

Example: URL for MGI-specific search interface

Base URL:

http://localhost/perl2/serve.pl?Page=OME::Web::CG_Search&Template=MGI

Example: show images associated with the gene hsp-68

…&Gene= hsp-68

In this example, gene is the name of a category group and hsp-68 is the name of a category within that group.

3.4. Display

Clicking on an image thumbnail in the search page opens the display page. The image thumbnail is blown up and all its annotations are shown. A desired customization of this display template is to link back to the mouse gene index from the image annotations page. This is done by adjusting the display template to inject the name of the gene category in a search url. e.g http://lgsun.grc.nia.nih.gov/geneindex5/bin /giU.cgi?search_term=foxl2

Now that we have established basic functionality, we can go back and rearrange html blocks in the Annotation and Search templates to make better use of screen space. The auto-generated HTML templates include documentation to make this easy for non-programmers.

4. DISCUSSION

Each component of the enumeration toolkit uses at least one piece of the underlying architecture. Without that base functionality, construction of the enumeration toolkit would have been much more difficult. We believe this illustrates the functional requirements for an ontology environment.

In addition to an extensible data store, an ontology environment needs tools that are transparently generated from data specifications. Tools are needed for middleware, annotation, display, and search. Middleware is object representation of the vocabulary, and is provided by OME's core. These auto-generated classes need to allow injection of code at select points and customization of display. For usability, end users without programming skills need tools

to define and work with a semi-structured vocabulary on the fly. This rudimentary user-defined vocabulary can later form the basis for a fully structured ontology.

The current framework can be used at several levels. Domain experts unfamiliar with programming can define and use a vocabulary. Users with some familiarity with HTML and the ability to follow examples can adjust HTML display templates. Users with some programming experience can customize the modular ADS code for enhanced functionality such as validating vocabulary against model organism databases. Web developers can use this basic ADS set to rapidly implement powerful, light weight front-ends for analysis or domain specific tasks. An example is measurement of DAPI stain to determine stage of cell cycle.

4.1. From enumerations to structured ontologies

Although enumerations are adequate for simple display of images and associated annotations, a more structured annotation would more closely reflect the actual relationships between genes, probes, and images. In reality, an image of an *in situ* experiment is associated with a gene only indirectly. The experiment makes use of a nucleic acid probe for the hybridization, which usually spans a sub-region of a gene. Many probes can be used to track the expression pattern of a given gene, and they may not yield equivalent results because they may vary in specificity. More specifically, hybridization signal is obtained by using an "anti-sense" probe for a given gene, with the "sense" probe used as a control.. Although the "sense" and "anti-sense" probes correspond to the same sub-region of a gene, they are expected to yield different results. This experimental setup therefore benefits from a "Type" specifier associated with a probe, where Type is either "sense" or "anti-sense". This can be achieved using the ad hoc vocabulary definition described in this work by specifying a new "Probe Type" Category Group, consisting of "sense" and "anti-sense" categories. However, a more accurate data model would define the probe type as a property of a probe object rather than a property of a specific image.

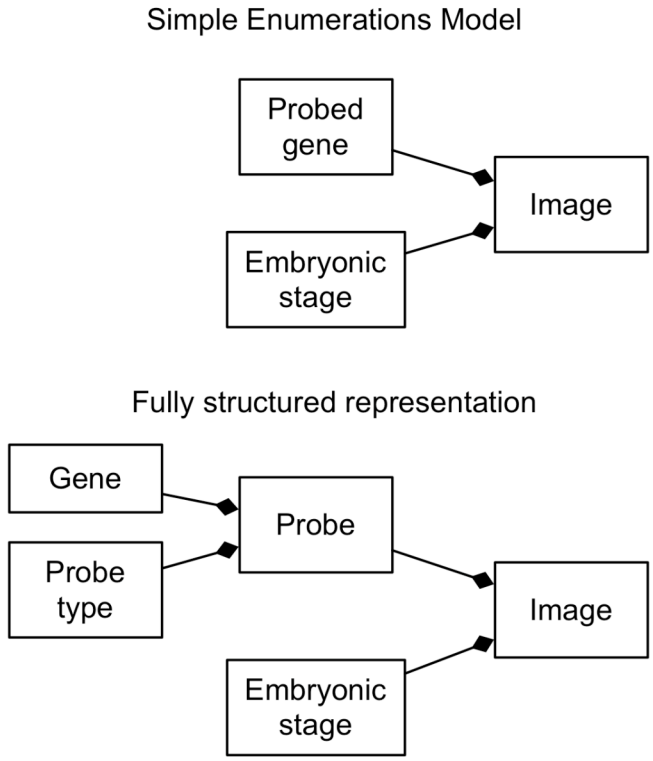

Figure 3: Class diagrams of semi-structured enumerations model and a fully structured representation. A line with a diamond on one side indicates a one-to-many relationship. For example, there is a one-to-many relationship between Probed gene and Image.

As you can see, the experimental system can only be approximated through these simple enumerations. Full description necessitates a more structured ontology. Tools to support this are in development. Currently, we have developed a tool for annotation (similar in function to the one described above) that can be used for arbitrary structured types (i.e., Semantic Types), and we are actively developing tools for Display and Search.

5. REFERENCES

[1] J.R. Swedlow, I. Goldberg, E. Brauner, and P.K. Sorger,. "Informatics and Quantitative Analysis in Biological Imaging." *Science* AAAS, 300(5616) 100-102, April 4, 2003.

[2] I.G. Goldberg, et al., "The Open Microscopy Environment (OME) Data Model and XML File: Open Tools for Informatics and Quantitative Analysis in Biological Imaging." *Genome Biology*, Biomed Central, London, pp. 6:R47, 2005.

[3] A. Mathes. "Folksonomies: Cooperative classification and communication through shared metadata." Computer Mediated Communication, LIS590CMC (Doctoral Seminar), Graduate School of Library and Information Science, University of Illinois Urbana-Champaign, December 2004.

[4] A. Sharov, et al., "Transcriptome Analysis of Mouse Stem Cells and Early Embryos" *PLOS Biology* 1(3), pp. 410-419, December 2003.

[5] http://www.openmicroscopy.org/custom-annotations/