Meeting Minutes 26 March 2004

Autumn A Cuellar

Word of the day: **use**, defined as 'To put into service or apply for a purpose.'

- **Alan Garny's e-mail**. Alan has requested information regarding the development of the ML's that he can present to a Paris group (no elaboration on who they are or what they do) who are interested in COR [http://cor.physiol.ox.ac.uk/] and CellML. My response included a brief update on AnatML (essentially replaced by an ontological description of how parts of the body relate to other parts of the body) and FieldML (re-created by Shane and David to describe geometry information for use within CMGui; not sure if FieldML as it is will be **use**ful to other groups). I also explained how the immediate focus of CellML has been shifted to tool development and ontology development. Alan requested more information on the ontology development. Matt says he will forward the *Experimental Physiology* paper to me. Has suddenly occurred to me that others might find this paper interesting, as well. Will post citation information (and link if available) on the Publications [../../public/about/publications.html] page.

- **Fedor Kolpakov's e-mail**. Fedor has gotten in touch with me regarding a possible collaboration between his group (BioUML [http://www.biouml.org/]) and ours. They are working on MathML support and eventually will have the ability to parse and edit CellML models and generate the corresponding MATLAB and Java code for simulations. What resources can we contribute? Poul thinks we do not have financial resources to donate. I think possibly a Java CellML API would be **use**ful to not only BioUML but other groups writing biological simulation packages in Java. Can we at least develop one of these? Matt mentions the Java implementation of the CellML API that Nigel Lovell's group (at the University of New South Wales) developed. This inspires a side discussion (see following paragraph) about the point of having so many implementations of the CellML API instead of just one implementation with several bindings. In any case, Matt will look into whether Nigel Lovell intends to make the Java code open source.

- **API situation**. (A lot of this comes from a clarifying e-mail exchange between Matt and I after the meeting.) Shane was asking why there are two different implementations of the CellML API (a C++ [http://cellml.sourceforge.net/] one and a Python one), rather than just one C++ implementation (because that one came first) with the Python bindings. Matt had several explanations for this:

  - First he reminded us of what an API is (from Matt's e-mail):

    > An API can be described in many ways that are essentially the same, for example, an API is a written contract between system developers and application developers, it is not a piece of code; an API is specification of a set of interfaces to access the functionality of lower level services; and so on. You will be familiar with some. SAX and DOM are two APIs for processing XML. While they do attempt to be completely language independent, they do tend to use ideas from particular kinds of languages; for example, DOM is quite Java like in its specification. A DOM implementation in a functional programming language may not have much utility.

  - Secondly, he pointed out that the Python implementation is built around the SAX parser, whereas the C++ implementation is built around a DOM parser.

  - Finally, Matt explained that the C++ implementation isn't really an implementation of the CellML API, because no CellML API specification exists. In Matt's words:

    > The current CellML API is not an API, it's some documentation generated from a C++ library. Therefore, there is no C++ implementation of an API.

Shane and David wonder what is being done to unite these implementations - a valid question considering that the CellML project seems rather disjointed right now. Matt agrees that something should be done, specifically putting together a full API specification, which would include many of the interfaces used in the C++ and/or Python libraries. Matt expects that discussions concerning the full API can get underway when the CellML Tools discussion list is up and running. This process will require input from the authors of all of the implementations.

- **FieldML/ontology databases**. This issue might require a lot of background from e-mails, but using my superhero power Paraphrasing ability, the short of it is: Matt has sent us a link to his demo anatomy ontology interface [http://n3.bioeng5.bioeng.auckland.ac.nz/TestRdfGraph2/]. This is a great step. We keep saying how powerful ontologies are going to be. Shane's caught on, and he's very excited by Matt's interface. He's revving up all engines and is ready to take off with it, so he asked Matt a whole bunch of questions about how to get the interface to the point where we can **use** it - how do we improve the interface, how can we expand it to include new datatypes and links, etc. One of his questions was in regard to getting CellML models linked up with this anatomy ontology (Do I detect a trend in Shane's questions?). He noted that the **`<cmeta:bio_entity>`** metadata in CellML models do not (but probably should) correspond with the terms in the Anatomy ontology (for example, cardiac vs. ventricular myocyte). Shane is interested in using the **`<cmeta:bio_entity>`** tags in building a FieldML database, but he was wondering if it's the right way to annotate his FieldML or is there a better way?

  Matt reveals that the database does not understand ontologies to the point where you can do complex queries. In order to get it to that point, we will have to work through the migration of this ontology environment to one that includes a repository that has a reasoner and useful editing tools that overcome the limitations of the current Protege [http://protege.stanford.edu/] environment. In the meantime, everyone seems to agree that the ontology can still be **use**ful. Shane's list for putting the ontology to use is this:

  1. Plan for syncing what sets of bio-entity to use.
  2. Develop a nicer interface.
  3. Make a list of things we neet to do to increase the ontology's usefulness.

  Addressing the first item of the above list, Matt's suggestion is (though circular, still sensible):

  1. Need to only put bio_entities in code that appear in ontology.
  2. To do that you need a useful ontology.
  3. Go through metadata and look at terms and compare with the ontology list, adding to database where necessary.
  4. To do the previous, we can and should **use** coded scripts to process the metadata.