Meeting Minutes 1 October 2000

Author:
Warren Hedley (Bioengineering Institute, University of Auckland)
Contributors:
David Bullivant (Bioengineering Institute, University of Auckland)
Melanie Nelson (Physiome Sciences Inc.)
Poul Nielsen (Bioengineering Institute, University of Auckland)

1 Class Definition and L2 XML Re-use

Warren and Melanie had been working for the last few days on the separation between class definition and low-level XML re-use, and Warren had decided against introducing any class definition specific elements into CellML as it would essentially just be duplicating the functionality provided by any low-level XML re-use system. So Warren had updated the April 2000 CRIM documents to better deal with issues raised by CellML class definition, and had included examples of this in both the L2 XML re-use document and the September 28 meeting minutes.

These documents were not greeted with the enthusiasm from the Auckland team that Warren had hoped for (not that anything Warren comes up with has ever been greeted with enthusiasm, but he still hopes). Firstly Poul and David expressed some concern over the terminology introduced in the September 28 minutes, and in particular the term "*class*", which the Auckland team had previously referred to as "*template*", and the fact that this means we have to introduce the term "*ontology-class*" to maintain the independence of the two concepts.

Even more concern was expressed about the use of the low-level re-use mechanisms for the implementation of class definition. Poul and David prefer to think of this in terms of high-level object-oriented terms, and want to see this reflected in the XML. Warren also thinks the same way, but as the functionality would be basically identical, doesn't see the point in introducing new elements. Poul and David then questioned the need for low-level XML re-use at all. Again, since the required functionality is so similar, there is no overhead once you accept that some kind of scheme is needed for the modification of class definitions (i.e., the setting of default variable values.) Warren waits for an alternative proposal from Poul and David.

1.1 Property Sets

The low-level XML re-use scheme is aimed at allowing the document author or software to reduce the size and complexity of CellML documents by allowing the virtual cutting and pasting of CellML data inside the document. However the scheme defined in the 1 October version of the L2 XML re-use document had at least one fatal flaw, which was only highlighted when discussion turned to the relevance of XML re-use and the example of a variable set was brought up.

Currently it has been assumed that the **<define>** element can only contain a single element. When the contents of the **<define>** element are copied, and modifications made to their structure, the XPath expressions that are used to address portions of the copied structure use "." to refer to the top-level element. In the event that the user wanted to define a set of variables, copy them into a component, and then make modifications, it would not be clear what "." referred to, and no good way of addressing into the variable set is obvious as it is unclear where the root of the structure is.

The simplest solution is to add the requirement that we can only ever copy elements that are inside a **<define>** element. (In the current XML re-use document that had not been explicitly stated because it was thought that it might be useful to be able to copy concrete instances as well as the *abstract* instances that are found in **<define>** elements.) In the XPath expressions that are found in modification elements, "." can then be assumed to refer to the **<define>** element itself. If the **element** and **name_ref** attributes

of a **<copy>** have values of `component` and `channel` respectively, these are mapped to the following XPath expressions:

```
//define/component[@name='channel']
//define/copy[@element='component'][@new_name='channel']
```

Another, less attractive, solution to this problem is to add a **<define_group>** element to the low-level XML re-use scheme. The **<define_group>** element may contain more than one element, whereas the **<define>** element may only contain one. When the modeller wishes to insert the content of the group into a document, the **<define_group>** element effectively vanishes. Before this happens however, XPath expressions used in content modification will use the **<define_group>** element as the root node.