Meeting Minutes 2 October 2000

Author:
Warren Hedley (Bioengineering Institute, University of Auckland)
Contributors:
Scott Lett (Physiome Sciences Inc.)
Yi Ge (Physiome Sciences Inc.)
Melanie Nelson (Physiome Sciences Inc.)

1 Summary

Not really a meeting today: just some ideas that Warren floated to Scott, Yi and Melanie for consideration and feedback. This document just summarises those ideas for the benefit of the Auckland team.

2 Script Definition

In an effort to remove any elements associated with scripting from the CellML namespace and data model, and to give the scripting process a bit more credibility, it was suggested by the Auckland team that scripts be declared and executed using the appropriate MathML tags.

Amazingly, MathML does not appear to include any elements specifically aimed at extending MathML functionality by using programming languages. There are also no examples in the specification relating to this problem, although they do show how it is possible to define a function in terms of MathML elements for re-use. With a combination of the semantics, declare and annotation elements however, Warren proposed a scheme that appears to be consistent with the ideas associated with each element type. The definition of an extremely simple function is given in Figure 1.

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
  <semantics>
    <declare nargs="2" type="fn">
      <ci>add</ci> <!-- the function name -->
    </declare>
    <annotation encoding="text/ecmascript">
      function add(a,b) {
        return(a+b);
      }
    </annotation>
  </semantics>
</math>
```

FIGURE 1: An example demonstrating a possible scheme for function defi nition using a MathML-conformant syntax.

The **scope** attribute on the **<declare>** (discussed in Section 4.3.2.8 of the MathML specification) should not be used, as the semantics of its values (`global` and `local`) are not what we want in CellML (local to the enclosing **<math>** element or the parent of the **<declare>** element respectively). In CellML a function may be executed from any math block in the same component as the function is defined. If a function is to be used at multiple locations in a model, it should be included in a class definition, which is

then instantiated wherever it is needed, and the appropriate connections made to each instance to pass in arguments and return the result.

Executing a function is fairly straightforward, and there are numerous examples of this in the MathML 2.0 March 2000 working draft. An adaption of one such example is given in Figure 2.

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
  <apply>
    <eq />
    <ci>c</ci>
    <apply>
      <csymbol>add</csymbol> <!-- the function! -->
      <ci>a</ci>
      <ci>b</ci> <!-- the arguments -->
    </apply>
  </apply>
</math>
```

FIGURE 2: An example demonstrating a scheme for calling the function defined using the syntax shown in Figure 1.

3 XMI

Scott had suggested looking at XMI (an XML serialization of UML) to see if it included an XML serialization of IDL, the Interface Definition Language widely used to declare classes and the associated methods in object-oriented programming. A quick scan of XMI revealed that it includes an incredibly powerful class definition scheme, so powerful in fact that it was immediately dismissed as way too complex for incorporation into CellML.

As anything to do with data modelling has a propensity for inducing states of extreme drowsiness in Warren, he suggested that Melanie have a closer look, to get her opinion.