Meeting Minutes 1 November

Author:
Warren Hedley (Bioengineering Institute, University of Auckland)
Contributors:
David Bullivant (Bioengineering Institute, University of Auckland)
Poul Nielsen (Bioengineering Institute, University of Auckland)

1 Introduction

In the Auckland meeting on November 1, some discussion was devoted to the specification of initial conditions and boundary conditions on variables. There were also brief conversations about where it was appropriate to provide context-specific shortcuts in place of Warren's low-level XML re-use scheme for element referencing, and advanced grouping functionality.

2 Initial And Boundary Conditions

Examination of a portion of a model description led to some discussion about the **`value`** on the **`<variable>`** element. Poul expressed some concern about the naming and flexibility of this, because it really makes the variable look like a constant. So, in order to make the role of this attribute clear, it will be re-named **`initial_value`**. It should also be explicitly stated in the specification that variables with a public or private interface value of in, may not contain an **`initial_value`** attribute, as the value of these variables is always calculated elsewhere.

CellML will also need a more general mechanism for defining initial and boundary conditions that is consistent with the MathML specification of equations. Unfortunately, the team didn't have time to properly consider the issues or involved, but the problems include:

- in math, it is assumed that all equations hold in every state, unless conditions are specified on an equation
- initial and boundary conditions could thus be specified as conditional equations. This is awkward in MathML: various schemes were proposed in the meeting minutes from October 4 2000[1] for dealing with variables as functions of time, and they were all inelegant.
- alternatively initial and boundary conditions could be associated with the declaration of a variable with specific **`<initial_condition>`** and **`<boundary_condition>`** elements.

3 Element Referencing Shortcuts

Warren had put considerable effort into re-drafting the *Component Re-use and Information Modification* scheme developed in April 2000 by the Auckland team into the new *Low Level XML Re-use* scheme. This is more powerful, elegant and self-consistent, but unfortunately lacks a catchy acronym, so Poul and David hadn't really put a lot of effort into reading this proposal.

Warren expressed some concern about the proliferation of elements like **`<variable_ref>`** and **`<component_`** through CellML, suggesting that the generic **`<reference>`** element from LLXR be used instead — that's what it's there for! The alternative notation is given in Figure 1.

Due to a lack of interest from the majority of the Auckland team, this topic didn't really receive the attention it deserved.

[1] http://www.cellml.org/private/progress_reports/20001004_meeting_minutes.html

```
<component_ref name_ref="my_membrane" />

<reference element="component" name_ref="my_membrane" />
```

FIGURE 1: Warren suggested replacing the **<component_ref>** from CellML with the generic **<reference>** element from LLXR.

4 Advanced Grouping

In the current scheme, the grouping mechanism is used to imply relationships between a group of minor components and possibly a major component. It was suggested that maybe grouping could also be used to associate properties with a number of components at once. There was also talk of grouping by type, i.e., all channels in a model might be grouped together, given a named type and properties associated with them. This might be a good way to specify metadata in one place and associate it with a number of different components or variables.