CellML Specification

Draft — 2 March 2001

4 Mathematics

4.1 Introduction

Model components may contain mathematical expressions that manipulate the values of variables that belong to the current component. These expressions are also free to use (but not change) the values of any other variables declared in the current component. Mathematical expressions must be defined using Mathematical Markup Language (MathML)[1], an XML-based language that encodes the underlying structure of an expression. If it is not possible to describe a particular mathematical expression in MathML or if a model specifically prescribes a certain algorithm, model authors may make use of ECMAScript (formerly Netscape's Javascript) to encode algorithms.

The first version[2] of MathML reached recommendation status at the World Wide Web Consortium (W3C)[3] in April 1998. A second version[4] was released in February 2001. CellML uses the content markup element set from MathML 2.0, which extends the functionality of MathML 1.0 in many key areas. The MathML 2.0 recommendation also deprecates the use of some MathML 1.0 elements. However, CellML processing software is expected to maintain compatibility with equations defined using MathML 1.0.

ECMA[5] is an international industry association that develops standards in information and communication for the European union. ECMA took the scripting language that Netscape[6] developed for adding interactive content to its web browser and developed the formal language specification ECMA-262 (ECMAScript Language Specification)[7].

4.2 Basic Structure

4.2.1 Allowed locations of mathematical expressions

All mathematical expressions defined using MathML must be placed inside a **<math>** element in the MathML namespace. A math element must be a child of a **<component>** element or a **<role>** element (the **<role>** element is a child of the **<variable_ref>** element in a **<reaction>** element).

The CellML syntax for defining scripts using ECMAScript has yet to be determined. It is anticipated that this will involve a **<script>** element in the CellML namespace that may be placed in **<model>**, **<component>** or **<role>** elements.

Mathematical expressions and scripts may manipulate the variables belonging to the component in which they are defined and may use (but not change) the value of variables belonging to other components if necessary. The declaration of variables in CellML is specified in Section 3[8].

[1] http://www.w3.org/Math
[2] http://www.w3.org/TR/1998/REC-MathML-19980407
[3] http://www.w3c.org/
[4] http://www.w3.org/TR/2001/REC-MathML2-20010221/
[5] http://www.ecma.ch/
[6] http://www.netscape.com/
[7] http://www.ecma.ch/ecma1/STAND/ECMA-262.HTM
[8] http://www.cellml.org/public/specification/20010302/cellml_specification.html#sec_model_structure

4.2.2 Allowed mathematical expressions

At the present time, all models defined in CellML consist solely of algebraic equations and ordinary differential equations. However, MathML can specify a wide-range of mathematics, not all of which will necessarily be interpretable by every CellML processor. To encourage interoperability of models created using different processors, the CellML development team intends to define groups of mathematical functionality. CellML processing software could then publish which groups of mathematical functionality it supports.

4.2.3 Definition of scripts

Scripts will be defined using ECMAScript. The CellML development team is still determining how best to incorporate these scripts into CellML.

The use of scripts in CellML is strongly discouraged. CellML is aimed at specifying a model in terms of its most basic governing equations. Wherever possible, mathematical equations should be used to specify the changing behaviour of a model's state variables.

When scripts are used, they are assumed to execute *instantly* with respect to any independent variables, making it possible for the integrator to ignore their execution.

4.3 Examples

The CellML fragment in Figure 6 demonstrates how MathML can be employed within CellML to define mathematical expressions. This fragment is part of the definition of a component that represents the behaviour of the *n* gate from the potassium channel in the Hodgkin-Huxley squid axon model of 1952. The component contains two units definitions, two variable declarations, and a block of MathML that defines the calculation of the `alpha` variable of the *n* gate. This equation has the form:

$$alpha\_n = \frac{0.01(V + 10.0)}{exp(0.1(V + 10.0)) - 1.0}$$

All of the **<mathml:cn>** elements in the equation define the required **cellml:units** attribute, which associates a units definition with the number delimited by the **<cn>** element. The inclusion of units in the equation allows CellML processing software to check that the dimensions on all terms in an equation are consistent, as defined in Section 5[9]

The **<mathml:apply>** element at the top level of the expression defines a **cmeta:id** attribute, which can be used to associate metadata (such as documentation) with the expression as described in Section 8[10].

4.4 Rules for CellML Documents

4.4.1 The <mathml:math> element

1. **Allowed use of the <mathml:math> element**

   - The **<mathml:math>** element must only appear as a child of **<cellml:component>** or **<cellml:role>** elements.

     [In this and subsequent rules, the use of the `mathml` and `cellml` prefixes indicates that elements and attributes are in the MathML and CellML namespaces, respectively.]

[9] http://www.cellml.org/public/specification/20010302/cellml_specification.html#sec_units
[10] http://www.cellml.org/public/specification/20010302/cellml_specification.html#sec_metadata

```
<component
    name="potassium_channel_n_gate"
    xmlns="http://www.cellml.org/2001/03/cellml">

  <units name="per_millisecond">
    <unit prefix="milli" units="second" exponent="-1" />
  </units>

  <units name="millivolt">
    <unit prefix="milli" units="volt" />
  </units>

  <variable name="alpha_n" units="per_millisecond" />
  <variable name="V" public_interface="in" units="millivolt" />

  <math xmlns="http://www.w3.org/1998/Math/MathML">
    <!--
      The alpha rate constant on the n gate is a function of membrane voltage.
    -->
    <apply cmeta:id="alpha_n_calculation"><eq />
      <ci> alpha_n </ci>
      <apply><divide />
        <apply><times />
          <cn cellml:units="per_millisecond"> 0.01 </cn>
          <apply><plus />
            <ci> V </ci>
            <cn cellml:units="millivolt"> 10.0 </cn>
          </apply>
        </apply>
        <apply><minus />
          <apply><exp />
            <apply><times />
              <cn cellml:units="dimensionless"> 0.1 </cn>
              <apply><plus />
                <ci> V </ci>
                <cn cellml:units="millivolt"> 10.0 </cn>
              </apply>
            </apply>
          </apply>
          <cn cellml:units="dimensionless"> 1.0 </cn>
        </apply>
      </apply>
    </apply>
  </math>

</component>
```

FIGURE 6: Part of the definition of a component that represents the behaviour of the *n* gate from the potassium channel in the Hodgkin-Huxley squid axon model of 1952. The component contains two units definitions, two variable declarations, and a block of MathML that defines the calculation of the *alpha* variable of the *n* gate. The **<mathml:apply>** element at the top level of the expression defines a **cmeta:id** attribute, which can be used to associate metadata (such as documentation) with the expression, as described in Section 8.

- The contents of a **<mathml:math>** element must conform to the Mathematical Markup Language (MathML) Version 2.0[11] recommendation from the W3C.

4.4.2 The <mathml:cn> element

1. **Allowed use of the <mathml:cn> element**

   - A **<mathml:cn>** element must define a **cellml:units** attribute.
     [All bare numbers in MathML content markup are enclosed in a **<cn>** element in the MathML namespace.]

2. **Allowed values of the cellml:units attribute**

   - The value of the **cellml:units** attribute must be taken from the standard dictionary of units given in Section 5.2.1[12], or be the value of the **name** attribute on a **<units>** element defined in the current **<component>** or **<model>** element.

4.5 Rules for Processor Behaviour

4.5.1 Order of calculations

Any component may contain multiple blocks of equations and multiple scripts. The order of execution of equations in any block is not necessarily the order in which they appear. All non-differential equations will be calculated before differential equations. This allows the evaluation of parameters before they are input into differential equations. Processing software must determine the appropriate ordering for the calculation of equations across multiple blocks of equations and across the model in general.

Scripts are assumed to execute *instantly* with respect to the independent variable *time*, making it possible for the integrator to ignore their execution.

[11] http://www.w3.org/TR/2000/WD-MathML2-20000328/
[12] http://www.cellml.org/public/specification/20010302/cellml_specification.html#sec_units_cellml_units_dictionary