The Reaction Element in CellML models

What is the Reaction Element?

The flexibility of the CellML language means it can be used to describe many different physiological processes which occur over a range of spatial and temporal scales. Such processes included signal transduction pathways, metabolic pathways, cell division cycles, and electrophysiological phenomena in excitable cells. While it is possible to specify a model purely in terms of equations expressed in MathML, when the CellML language was first defined a reaction element was included. This reaction element was designed to describe individual steps in a reaction pathway or network, and it included a description of the reaction kinetics, the identites of the reactants and products, and also identified any enzyme catalysts or inhibitors. The justification for the inclusion of the reaction element in the CellML specification was because it was felt valuable information about the reaction would be lost if the model was reduced to pure mathematics.

Deprecating and Removing the Reaction Element

However, in practice we found implementing the reaction element in a CellML model often required the equations to be re-written, such that they no longer reflected theose in the original publication. At best this created extra work and some confusion, at worst it broke the model. Consequently we have decided to deprecate the reaction element in CellML1.1.1, and it will be removed completely from the CellML1.2 specification. Reaction information will not be lost because in the future we intend to implement the use of an ontology to label the model variables as reactants, enzymes, products etc., and this information will be stored as metadata.

Rewriting the Reaction Models

In the meantime, we have been left with many models in the CellML repository which include the reaction element. The majority of these have already been re-written and the reaction element has been replaced by simple mathematical equations. As part of this process many of the CellML models have been fixed, such that they are now capable of replicating the results of the published model.

In most cases we found the structure of a model with and without the reaction element was so different it necessitated completely re-writing the model. In this way we were able to translate approximately three quarters of the reaction models in the CellML model repository. Where we were unable to translate the models it was usually because the original published paper did not contain sufficient information. In particular the reaction element supported qualitative models, defined as models which consist solely of information about how the different chemical species in the pathway relate, and contain no mathematics. As such there is no mathematical representation of these models, and we were unable to re-write them without the reaction element.

How to Recode a CellML Model to Remove the Reaction Element

1) Assess whether or not the publication contains a complete, quantitative model with sufficient data to translate it into a working CellML model.
2) Assuming it does, the model is best re-written from scratch, either following the structure outlined in the published paper, or by breaking the model down into a set of differential equations representing the reactions and the changing concetraitons of reactants and products as the sum of these fluxes.
3) Here we highlight, using a simple example, how the reaction element can be removed from the CellML model:

A + B <-----> C

Where A and B are the reactants, C is the product, it's a simple mass action reaction with no catalysts or inhibitors, and the reaction is non-reversible.

With the reaction element:

Each substrate (A, B, C, etc.) is defined in its own component and its changing concentration is defined by a differential equation, calculated as the sum of the imported reaction fluxes:

```
<component cmeta:id="A" name="A">
  <variable units="micromolar" public_interface="out" name="A" initial_value="1.0"/>
  <variable units="flux" public_interface="in" name="delta_A_rxn1"/>
  <variable units="second" public_interface="in" name="time"/>
  <math xmlns="http://www.w3.org/1998/Math/MathML">
    <apply><eq/>
      <apply><diff/>
        <bvar><ci>time</ci></bvar>
        <ci>A</ci>
      </apply>
      <ci>delta_A_rxn1</ci>
    </apply>
  </math>
</component>
```

Each reaction is also defined in its own component, and the *reaction element* asigns a *role* to the imported substates such as *reactant* (A&B) or *product* (C), and states whether or not the reaction is *reversible* (in this example it is). The *stoichiometry* attribute defines the stoichiomentry of a variable relative to the other reaction participants, the *direction* attribute defines which direction the referenced variable serves as reactant or product, while the *delta variable* represents the change in the concentration of the variable due to its involvement in the current reaction.

```
<component cmeta:id="reaction1" name="reaction1">
  <variable units="micromolar" public_interface="in" name="A"/>
  <variable units="micromolar" public_interface="in" name="B"/>
  <variable units="micromolar" public_interface="in" name="C"/>
  <variable units="flux" public_interface="out" name="delta_A_rxn1"/>
  <variable units="flux" public_interface="out" name="delta_B_rxn1"/>
  <variable units="flux" public_interface="out" name="delta_C_rxn1"/>
  <variable units="second_order_rate_constant" name="k1" initial_value="1.0"/>
  <variable units="flux" name="rate"/>

  <reaction reversible="yes">
    <variable_ref variable="A">
      <role stoichiometry="1" direction=”forward”
            delta_variable="delta_A_rxn1" role="reactant"/>
    </variable_ref>
    <variable_ref variable="B">
      <role stoichiometry="1"  direction=”forward”
            delta_variable="delta_B_rxn1" role="reactant"/>
    </variable_ref>
    <variable_ref variable="C">
      <role stoichiometry="1"  direction=”forward”
            delta_variable="delta_C_rxn1" role="product"/>
    </variable_ref>
    <variable_ref variable="rate">
```

```
        <role role="rate">

          <math xmlns="http://www.w3.org/1998/Math/MathML">
            <apply><eq/>
              <ci>rate</ci>
              <apply><minus/>
                <apply><times/>
                  <ci>k1</ci>
                  <ci>A</ci>
                  <ci>B</ci>
                </apply>
                <apply><times/>
                  <ci>k_1</ci>
                  <ci>C</ci>
                </apply>
              </apply>
            </apply>
          </math>
        </role>
      </variable_ref>
    </reaction>
  </component>
```

Without the reaction element:

```
  <component cmeta:id="A" name="A">
    <variable units="micromolar" public_interface="out" name="A" initial_value="1.0"/>
    <variable units="flux" public_interface="in" name="v_rxn1"/>
    <variable units="second" public_interface="in" name="time"/>
    <math xmlns="http://www.w3.org/1998/Math/MathML">
      <apply><eq/>
        <apply><diff/>
          <bvar><ci>time</ci></bvar>
          <ci>A</ci>
        </apply>
        <ci>v_rxn1</ci>
      </apply>
    </math>
  </component>

  <component name="reaction1">
    <variable units="micromolar" public_interface="in" name="A"/>
    <variable units="micromolar" public_interface="in" name="B"/>
    <variable units="micromolar" public_interface="in" name="C"/>
    <variable units="flux" public_interface="out" name="v_rxn1"/>
    <variable units="second_order_rate_constant" name="k1" initial_value="1.0"/>
    <variable units="first_order_rate_constant" name="k_1" initial_value="1.0"/>

    <math xmlns="http://www.w3.org/1998/Math/MathML">
      <apply><eq/>
        <ci>v_rxn1</ci>
        <apply><minus/>
          <apply><times/>
            <ci>k1</ci>
```

```
            <ci>A</ci>
            <ci>B</ci>
          </apply>
          <apply><times/>
            <ci>k_1</ci>
            <ci>C</ci>
          </apply>
        </apply>
      </apply>
    </math>
  </component>
```

It is relatively straight forward to extract the variable information and the mathematical equation from the CellML reaction element, as illustrated above. However, the reaction element also contains a lot of other descriptive information which we do not wish to lose. It is our intention that this information will be captured in the model metadata using OWL ontologies. We are currently working on the best method of implementation for this, and we will describe this process as soon as it has been finalised.