```xml
<?xml version="1.0"?>

<!--
FILE : two_reaction_model_with_encapsulation_raw.xml

CREATED : 18 January 2001

LAST MODIFIED : 22nd July 2002

AUTHOR : Melanie Nelson
         Physiome Sciences, Inc.

MODEL STATUS :  This model conforms to the CellML 1.0 Specification released on
                10th August 2001, and the 16/01/2002 CellML Metadata 1.0
                Specification.

DESCRIPTION : This file contains a CellML description of an extremely simple
  reaction/pathway cellular model. The model and associated markup have been
  created solely for demonstration purposes.

CHANGES:  Altered "concentration_units" to "flux_units" for delta’s in the "total_reaction" c

-->

<!--
  The root element for our CellML model description is <model>. This contains
  a "name" attribute which would be used if the model were to be combined with
  another model at a later date, or if the model is to be referenced in some
  way by another model.

  Two namespaces are declared on the root element. The first sets the default
  namespace for the <model> element and all elements contained within the
  <model> element to the CellML namespace. The second namespace is again the
  CellML namespace, but this time declared with an explicit "cellml" prefix.
  This declaration has document-wide scope, so the "cellml" prefix may be used
  anywhere to move an element or attribute into the CellML namespace.
-->

<model name="two_reaction_model_with_encapsulation" xmlns="http://www.cellml.org/cellml/1.0#"
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:bqs="http://www.cellml
    <!--
      The following RDF block contains metadata that applies to this document
      as a whole, as indicated by the empty about attribute on the
      <rdf:Description> element.
    -->
    <rdf:Description rdf:about="">
      <!--
        The Model Builder Metadata.  The Dublin Core "creator" element is used
        to indicate the person who translated the model into CellML.
      -->
      <dc:creator rdf:parseType="Resource">
        <vCard:N rdf:parseType="Resource">
          <vCard:Family>Nelson</vCard:Family>
          <vCard:Given>Melanie</vCard:Given>
        </vCard:N>
        <vCard:ORG rdf:parseType="Resource">
          <vCard:Orgname>Physiome Sciences</vCard:Orgname>
        </vCard:ORG>
      </dc:creator>
    <!--
      The Creation Date metadata. This is the date on which the model
      was translated into CellML.
    -->
    <dcterms:created rdf:parseType="Resource">
      <dcterms:W3CDTF>2000-11-20</dcterms:W3CDTF>
    </dcterms:created>

    <!--
      The Last Modified Date metadata. This is the date on which
      the model was last changed.
    -->
    <cmeta:modification rdf:parseType="Resource">
      <rdf:value>
        Added metadata.
      </rdf:value>
      <cmeta:modifier rdf:parseType="Resource">
        <vCard:N rdf:parseType="Resource">
          <vCard:Family>Lloyd</vCard:Family>
          <vCard:Given>Catherine</vCard:Given>
          <vCard:Other>May</vCard:Other>
        </vCard:N>
      </cmeta:modifier>
      <dcterms:modified rdf:parseType="Resource">
        <dcterms:W3CDTF>2001-01-22</dcterms:W3CDTF>
      </dcterms:modified>
    </cmeta:modification>


    <!-- The Publisher metadata. -->
    <dc:publisher>Physiome Sciences</dc:publisher>
  </rdf:Description>

  <!--
    The following metadata refers to the model itself, as indicated by the
    reference to the ID "two_reaction_model_with_encapsulation", which is
    declared on the <model> element.
  -->
  <rdf:Description rdf:about="#two_reaction_model_with_encapsulation">
    <!-- A human readable name for the model. -->
    <dc:title>A Simple Two Reaction Model With Encapsulation</dc:title>

    <!-- A comment regarding the model. -->
    <cmeta:comment rdf:parseType="Resource">
      <rdf:value>
        Below is a CellML description of a simple two reaction model with                en

        The purpose of this description is to illustrate how CellML can be
        used to model metabolic and signal transduction pathways                         wi
      </rdf:value>
      <!-- The creator of the comment. -->
      <dc:creator>
        <vCard:FN>Catherine Lloyd</vCard:FN>
      </dc:creator>
    </cmeta:comment>
  </rdf:Description>
</rdf:RDF>
```

```
<units name="concentration_units">
  <unit prefix="milli" units="mole" />
  <unit units="litre" exponent="-1" />
</units>

<units name="flux_units">
  <unit units="concentration_units" exponent="1" />
  <unit units="second" exponent="-1" />
</units>

<units name="first_order_rate_constant">
  <unit units="second" exponent="-1" />
</units>

<units name="second_order_rate_constant">
  <unit units="concentration_units" exponent="-1" />
  <unit units="second" exponent="-1" />
</units>

<units name="third_order_rate_constant">
  <unit units="concentration_units" exponent="-2" />
  <unit units="second" exponent="-1" />
</units>


<!--
  The environment component is used to declare variables that are used
  by all or most of the other components. Variables must be declared inside
  of a component element.
-->
<component name="environment">
  <variable name="time" public_interface="out" units="second" />
</component>


<!--
  The first six components correspond to the reactants and products in
  the two reactions in this simple model. The next two components represent
  the elementary reactions in this model: (A + B <-> 2C + D and
  C + E <-> F). The final component represents the total reaction defined
  by this model (A + B + 2E <-> 2F), and encapsulates the intermediate
  reactions and species.

  Note that due to the encapsulation relationship, not all of these
  components are "visible" to each other. Components A, B, E, and F may
  be connected to each other, and components C, D, first_reaction, and
  second_reaction may be connected to each other. However, component A,
  B, E, or F may not be connected to component C or D, since components
  C and D are encapsulated by the total_reaction component. The two
  components representing the elementary reactions, first_reaction and
  second_reaction, are also encapsulated by the total_reaction component,
  and hidden from the unencapsulated components A, B, E, and F. Any
  information that is passed from an unencapsulated component to an
  encapsulated component must be passed through the encapsulating component.
  That is, a variable in an unencapsualted component such as A can not be
  directly mapped to a variable in an encapsulated component such as
  first_reaction. It must first be mapped to a variable in the encapsulating
  component, total_reaction, and the variable in total_reaction mapped
  to the target variable in the encapsualted component.
-->
<component name="A">
  <variable name="A" public_interface="out" units="concentration_units" />
  <variable name="delta_A" public_interface="in" units="flux_units" />
  <variable name="time" public_interface="in" units="second" />

  <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 </ci>
    </apply>
  </math>
</component>

<component name="B">
  <variable name="B" public_interface="out" units="concentration_units" />
  <variable name="delta_B" public_interface="in" units="flux_units" />
  <variable name="time" public_interface="in" units="second" />

  <math xmlns="http://www.w3.org/1998/Math/MathML">
    <apply><eq />
       <apply><diff />
        <bvar><ci> time </ci></bvar>
        <ci> B </ci>
      </apply>
      <ci> delta_B </ci>
    </apply>
  </math>
</component>

<component name="C">
  <variable name="C" public_interface="out" units="concentration_units" />
  <variable name="delta_C_rxn1" public_interface="in" units="flux_units" />
  <variable name="delta_C_rxn2" public_interface="in" units="flux_units" />
  <variable name="time" public_interface="in" units="second" />

  <math xmlns="http://www.w3.org/1998/Math/MathML">
    <apply><eq />
       <apply><diff />
        <bvar><ci> time </ci></bvar>
        <ci> C </ci>
      </apply>
      <apply><plus />
        <ci> delta_C_rxn1 </ci>
        <ci> delta_C_rxn2 </ci>
      </apply>
    </apply>
  </math>
</component>

<component name="D">
  <variable name="D" public_interface="out" units="concentration_units" />
  <variable name="delta_D" public_interface="in" units="flux_units" />
  <variable name="time" public_interface="in" units="second" />
```

```
    <math xmlns="http://www.w3.org/1998/Math/MathML">
      <apply><eq />
         <apply><diff />
          <bvar><ci> time </ci></bvar>
          <ci> D </ci>
        </apply>
        <ci> delta_D </ci>
      </apply>
    </math>
  </component>

  <component name="E">
    <variable name="E" public_interface="out" units="concentration_units" />
    <variable name="delta_E" public_interface="in" units="flux_units" />
    <variable name="time" public_interface="in" units="second" />

    <math xmlns="http://www.w3.org/1998/Math/MathML">
      <apply><eq />
         <apply><diff />
          <bvar><ci> time </ci></bvar>
          <ci> E </ci>
        </apply>
        <ci> delta_E </ci>
      </apply>
    </math>
  </component>

  <component name="F">
    <variable name="F" public_interface="out" units="concentration_units" />
    <variable name="delta_F" public_interface="in" units="flux_units" />
    <variable name="time" public_interface="in" units="second" />

    <math xmlns="http://www.w3.org/1998/Math/MathML">
      <apply><eq />
         <apply><diff />
          <bvar><ci> time </ci></bvar>
          <ci> F </ci>
        </apply>
        <ci> delta_F </ci>
      </apply>
    </math>
  </component>

  <component name="first_reaction">
    <!-- These variables are modifiable elsewhere and imported. -->
    <variable name="A" public_interface="in" units="concentration_units" />
    <variable name="B" public_interface="in" units="concentration_units" />
    <variable name="C" public_interface="in" units="concentration_units" />
    <variable name="D" public_interface="in" units="concentration_units" />

    <!-- These variables are modifiable in this component and exported. -->
    <variable name="delta_A" public_interface="out" units="flux_units" />
    <variable name="delta_B" public_interface="out" units="flux_units" />
    <variable name="delta_C" public_interface="out" units="flux_units" />
    <variable name="delta_D" public_interface="out" units="flux_units" />

    <!-- These variables are "private" to this component. -->
    <variable name="k_forward" units="second_order_rate_constant" />
    <variable name="k_reverse" units="third_order_rate_constant" />
    <variable name="r" units="flux_units" />

    <!--
      The <reaction> element is used to indicate which chemical species are
      participating in this reaction, and what role they play in the reaction.
    -->
    <reaction>
      <variable_ref variable="A">
        <role role="reactant" direction="forward" delta_variable="delta_A" stoichiometry="1"
      </variable_ref>
      <variable_ref variable="B">
        <role role="reactant" direction="forward" delta_variable="delta_B" stoichiometry="1"
      </variable_ref>
      <variable_ref variable="C">
        <role role="product" direction="forward" delta_variable="delta_C" stoichiometry="2" /
      </variable_ref>
      <variable_ref variable="D">
        <role role="product" direction="forward" delta_variable="delta_D" stoichiometry="1" /
      </variable_ref>
      <variable_ref variable="r">
        <role role="rate">
          <math xmlns="http://www.w3.org/1998/Math/MathML">
            <apply><eq />
              <ci> r </ci>
              <apply><plus />
                <apply><minus />
                  <apply><times />
                    <ci> k_forward </ci>
                    <ci> A </ci>
                    <ci> B </ci>
                  </apply>
                </apply>
                <apply><times />
                  <ci> k_reverse </ci>
                  <apply><power />
                    <ci> C </ci>
                    <cn cellml:units="dimensionless"> 2.0 </cn>
                  </apply>
                  <ci> D </ci>
                </apply>
              </apply>
            </apply>
          </math>
        </role>
      </variable_ref>
    </reaction>

  </component>

  <component name="second_reaction">
    <!-- These variables are modifiable elsewhere and imported. -->
    <variable name="C" public_interface="in" units="concentration_units" />
    <variable name="E" public_interface="in" units="concentration_units" />
    <variable name="F" public_interface="in" units="concentration_units" />

    <!-- These variables are modifiable in this component and exported. -->
    <variable name="delta_C" public_interface="out" units="flux_units" />
    <variable name="delta_E" public_interface="out" units="flux_units" />
    <variable name="delta_F" public_interface="out" units="flux_units" />
    <!-- These variables are internal to this component. -->
    <variable name="k_forward" units="second_order_rate_constant" />
    <variable name="k_reverse" units="first_order_rate_constant" />
    <variable name="r" units="flux_units" />

    <reaction>
      <variable_ref variable="C">
        <role role="reactant" direction="forward" delta_variable="delta_C" stoichiometry="1"
      </variable_ref>
      <variable_ref variable="E">
        <role role="reactant" direction="forward" delta_variable="delta_E" stoichiometry="1"
      </variable_ref>
      <variable_ref variable="F">
        <role role="product" direction="forward" delta_variable="delta_F" stoichiometry="2" /
      </variable_ref>
      <variable_ref variable="r">
        <role role="rate">
          <math xmlns="http://www.w3.org/1998/Math/MathML">
            <apply><eq />
              <ci> r </ci>
              <apply><plus />
                <apply><minus />
                  <apply><times />
                    <ci> k_forward </ci>
                    <ci> C </ci>
                    <ci> E </ci>
                  </apply>
                </apply>
                <apply><times />
                  <ci> k_reverse </ci>
                  <ci> F </ci>
                </apply>
              </apply>
            </apply>
          </math>
        </role>
      </variable_ref>
    </reaction>

  </component>

  <!--
    The total_reaction component encapsulates the intermediate reactions and
    intermediate species involved in the total reaction (A + B + 2E <-> 2F).
    It contains no math of its own, but does contain a reaction element, which
    declares the participants in the overall reaction. Note that it would
    technically be possible to derive the information in the total_reaction’s
    reaction element from the reaction elements of the encapsulated
    reactions (first_reaction and second_reaction).
  -->
  <component name="total_reaction">
    <!--
      These variables are imported from unencapsulated (sibling) components
      (public_interface="in"), and passed on to encapsulated components
      (private_interface="out").
    -->
    <variable name="A" public_interface="in" private_interface="out" units="concentration_uni
    <variable name="B" public_interface="in" private_interface="out" units="concentration_uni
```

```
    <variable name="E" public_interface="in" private_interface="out" units="concentration_uni
    <variable name="F" public_interface="in" private_interface="out" units="concentration_uni
    <variable name="time" public_interface="in" private_interface="out" units="second" />

    <!--
      These variables are imported from encapsulated components
      (private_interface="in") and exported to the rest of the network
      (public_interface="out").
    -->
    <variable name="delta_A" public_interface="out" private_interface="in" units="flux_units"
    <variable name="delta_B" public_interface="out" private_interface="in" units="flux_units"
    <variable name="delta_E" public_interface="out" private_interface="in" units="flux_units"
    <variable name="delta_F" public_interface="out" private_interface="in" units="flux_units"

    <!--
      The <reaction> element in the encapsulating reaction component should
      describe the result of all of the encapsulated detail qualitatively.
      For this reason the delta_variable attributes are not used on the
      <role> elements, so that no mathematics is implicitly defined, which
      would duplicate the mathematics that may be defined within the
      encapsulated components. The information here allows software to generate
      the equation and pathway diagrams representations of the total reaction.
    -->
    <reaction>
      <variable_ref variable="A">
        <role role="reactant" direction="forward" stoichiometry="1" />
      </variable_ref>
      <variable_ref variable="B">
        <role role="reactant" direction="forward" stoichiometry="1" />
      </variable_ref>
      <variable_ref variable="E">
        <role role="reactant" direction="forward" stoichiometry="1" />
      </variable_ref>
      <variable_ref variable="F">
        <role role="product" direction="forward" stoichiometry="2" />
      </variable_ref>
    </reaction>

  </component>

  <!--
    The group element creates the encapsulation relationship. The total_reaction
    component encapsulates two intermediate reactions and two intermediate
    species.
  -->
  <group>
    <relationship_ref relationship="encapsulation" />
    <component_ref component="total_reaction">
      <component_ref component="first_reaction" />
      <component_ref component="second_reaction" />
      <component_ref component="C" />
      <component_ref component="D" />
    </component_ref>
  </group>

  <!--
    The connections define the mappings between variables declared in
    different components. Note the two step mapping of variables declared
    in an unencapsulated component (such as "A") to an encapsulated component
    (such as "first_reaction").
  -->
  <connection>
    <map_components component_1="A" component_2="total_reaction" />
    <map_variables variable_1="A" variable_2="A" />
    <map_variables variable_1="delta_A" variable_2="delta_A" />
  </connection>


  <connection>
    <map_components component_1="B" component_2="total_reaction" />
    <map_variables variable_1="B" variable_2="B" />
    <map_variables variable_1="delta_B" variable_2="delta_B" />
  </connection>


  <connection>
    <map_components component_1="C" component_2="first_reaction" />
    <map_variables variable_1="C" variable_2="C" />
    <map_variables variable_1="delta_C_rxn1" variable_2="delta_C" />
  </connection>


  <connection>
    <map_components component_1="C" component_2="second_reaction" />
    <map_variables variable_1="C" variable_2="C" />
    <map_variables variable_1="delta_C_rxn2" variable_2="delta_C" />
  </connection>


  <connection>
    <map_components component_1="D" component_2="first_reaction" />
    <map_variables variable_1="D" variable_2="D" />
    <map_variables variable_1="delta_D" variable_2="delta_D" />
  </connection>


  <connection>
    <map_components component_1="E" component_2="total_reaction" />
    <map_variables variable_1="E" variable_2="E" />
    <map_variables variable_1="delta_E" variable_2="delta_E" />
  </connection>


  <connection>
    <map_components component_1="F" component_2="total_reaction" />
    <map_variables variable_1="F" variable_2="F" />
    <map_variables variable_1="delta_F" variable_2="delta_F" />
  </connection>


  <connection>
    <map_components component_1="first_reaction" component_2="total_reaction" />
    <map_variables variable_1="A" variable_2="A" />
    <map_variables variable_1="delta_A" variable_2="delta_A" />
    <map_variables variable_1="B" variable_2="B" />
    <map_variables variable_1="delta_B" variable_2="delta_B" />
  </connection>
```

```
<connection>
  <map_components component_1="second_reaction" component_2="total_reaction" />
  <map_variables variable_1="E" variable_2="E" />
  <map_variables variable_1="delta_E" variable_2="delta_E" />
  <map_variables variable_1="F" variable_2="F" />
  <map_variables variable_1="delta_F" variable_2="delta_F" />
</connection>


<!--
  The unencapsulated components A, B, E, and F can receive the "time"
  variable directly from the environment component (also unencapsulated).
  In contrast, the encapsulated components C and D must receive all variables
  from the encapsulating component, "total_reaction", or from other components
  encapsualted by the total_reaction component. Therefore, "time" is passed
  from the environment component to the total_reaction component, and
  then on to the C and D components.
-->
<connection>
  <map_components component_1="A" component_2="environment" />
  <map_variables variable_1="time" variable_2="time" />
</connection>


<connection>
  <map_components component_1="B" component_2="environment" />
  <map_variables variable_1="time" variable_2="time" />
</connection>


<connection>
  <map_components component_1="E" component_2="environment" />
  <map_variables variable_1="time" variable_2="time" />
</connection>


<connection>
  <map_components component_1="F" component_2="environment" />
  <map_variables variable_1="time" variable_2="time" />
</connection>


<connection>
  <map_components component_1="total_reaction" component_2="environment" />
  <map_variables variable_1="time" variable_2="time" />
</connection>


<connection>
  <map_components component_1="C" component_2="total_reaction" />
  <map_variables variable_1="time" variable_2="time" />
</connection>


<connection>
  <map_components component_1="D" component_2="total_reaction" />
  <map_variables variable_1="time" variable_2="time" />
</connection>
```

```
</model>
```