# Systems Biology Markup Language (SBML) Level 3 Proposal: Model Composition Features

Andrew Finney  
afinney@cds.caltech.edu  

October 31, 2003

## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Introduction</td>
<td>2</td>
</tr>
<tr>
<td>2 Acknowledgements</td>
<td>2</td>
</tr>
<tr>
<td>3 Overview</td>
<td>2</td>
</tr>
<tr>
<td>4 Submodels</td>
<td>2</td>
</tr>
<tr>
<td>4.1 Libraries of models</td>
<td>4</td>
</tr>
<tr>
<td>5 Instances</td>
<td>4</td>
</tr>
<tr>
<td>5.1 Issue</td>
<td>5</td>
</tr>
<tr>
<td>6 Object References</td>
<td>5</td>
</tr>
<tr>
<td>6.1 Issue</td>
<td>6</td>
</tr>
<tr>
<td>7 Links</td>
<td>6</td>
</tr>
<tr>
<td>7.1 Link Validity</td>
<td>7</td>
</tr>
<tr>
<td>8 Ports</td>
<td>7</td>
</tr>
<tr>
<td>9 Examples using Links</td>
<td>8</td>
</tr>
<tr>
<td>9.1 Model Composition Without Ports</td>
<td>8</td>
</tr>
<tr>
<td>9.2 Model Composition with Ports</td>
<td>10</td>
</tr>
<tr>
<td>10 Direct Links</td>
<td>12</td>
</tr>
<tr>
<td>10.1 Direct links using ports</td>
<td>13</td>
</tr>
<tr>
<td>10.2 Direct links in math expressions</td>
<td>13</td>
</tr>
<tr>
<td>11 Example of Direct links</td>
<td>13</td>
</tr>
<tr>
<td>References</td>
<td>15</td>
</tr>
</tbody>
</table>
1 Introduction

This document describes proposed features for inclusion in Systems Biology Markup Language (SBML) Level 3. This document describes features enabling the composition of models from multiple instances of submodels.

This document is not a definition of SBML Level 3 or part of it. This document simply presents various features which could be incorporated into SBML Level 3 as the Systems Biology community wishes. This document is intended for detailed review by that community and to provoke alternative proposals. This proposal is designed to provide an alternative permutation of ideas proposed by Martin Ginkel (Ginkel, 2002) and Jonathan Webb (Webb, 2003). This proposal is designed with a proposal for arrays (Finney et al., 2003) in mind and is one important motivations for the creation of this proposal.

Throughout this document issues that the author believes will require further discussion have been highlighted.

For brevity the text of this document is with reference to SBML Level 2 (Finney et al., 2002) i.e. features are described in terms of changes to SBML Level 2. In addition for brevity the UML diagrams in this proposal show only new attributes and types for SBML Level 3. For SBML Level 2 types Level 2 attributes are meant to be present in SBML Level 3.

All types proposed in this document will be derived from the SBase type.

2 Acknowledgements

This proposal is based upon the prior proposals made by Martin Ginkel (Ginkel, 2002) and Jonathan Webb (Webb, 2003). This proposal has benefitted from discussions the author had with Martin and Jonathan.

3 Overview

A UML diagram for the proposal is shown in figure [1].

A under this proposal a Model structure would have an optional lists of Model (see section 4), Instance (see section 5), Link (see section 7) and Port (see section 8) structures. Section 6 describes the generic mechanism for referring to objects in models that is used in this proposal. Complete examples of models using these proposed structures is given in section 9.

The new structures attached to Model structures are sufficient for model composition however section 10 describes modifications to existing SBML Level 2 components like SimpleSpeciesReference which enables this proposal to be more flexible. An example is given in section 11 which demonstrates this feature.

4 Submodels

In this proposal a model can contain any number of submodels. A submodel is a full fledged model with its own namespace: object identifiers are only in scope within the immediate enclosing Model structure. These submodels may or may not be part of the actual model description i.e. instances of these submodels may or may not occur in the enclosing model. Instances of these submodels are created through the use of Instance structures (see section 5). A submodel does not have to be included within a composed model. A submodel can be external to the composed model (see section 5).

The following example shows a submodel enclosed in another model. The submodel inner is redundant as no instance of it exists in the outer model.
Figure 1: The types and attributes introduced into SBML by this proposal

```xml
<?xml version="1.0"?>
<sbml xmlns="http://www.sbml.org/sbml/level3" version="1" level="3">
  <model id="outer">
    <listOfCompartments>
      <compartment id="compartmentOne" volume="1"/>
    </listOfCompartments>
    <listOfSpecies>
      <species id="S1" initialAmount="10" compartment="compartmentOne">
        ...
      </species>
    </listOfSpecies>
  </model>
</sbml>
```
4.1 Libraries of models

A SBML stream can be interpreted as a library of models if the top level Model structure only contains a list of submodels. The previous example would not be considered a library.

5 Instances

Under this proposal a model can be composed of instances of submodels through the use of Instance structures. An Instance structure refers to a Model structure. An Instance structure simply represents a copy of that Model structure within the current model.

An Instance structure uses XLink (DeRose et al., 2001) attributes to refer to a submodel inside or outside the stream containing the Instance structure. The href attribute contains an XPointer (DeRose et al., 2002) string that points to either an SBML model document or to a model element. If the pointer refers to an SBML document then the pointer is equivalent to a pointer referring to the top-level model in that document.
document. The type, which must have the value simple, simply indicates the XLink type of Instance. The Instance structure has an id attribute to identify the instance. This identifier exists in the immediate enclosing model’s namespace.

For example the following fragment, if inserted into the previous example, refers to the inner model:

```xml
<listOfInstances>
  <instance
    id="innerA"
    xlink:type="simple"
    xlink:href="#xpointer(/sbml/model/listOfSubmodels/model[@id=%22inner%22])"/>
</listOfInstances>
```

The following fragment is an equivalent reference from another model to the original example contained in a file X.xml:

```xml
<listOfInstances>
  <instance
    id="innerA"
    xlink:type="simple"
    xlink:href="X.xml#xpointer(/sbml/model/listOfSubmodels/model[@id=%22inner%22])"/>
</listOfInstances>
```

The following fragment is a reference from another model to the top-level model contained in a file X.xml:

```xml
<listOfInstances>
  <instance
    id="innerB"
    xlink:type="simple"
    xlink:href="X.xml"/>
</listOfInstances>
```

**5.1 Issue**

We used XLink to refer to submodels because its a standard mechanism for linking XML elements inside and outside a given document. XLink attributes can be interpreted by XML aware tools.

We may wish to significantly restrict the content of the href attribute content to consist only of the form, in BNF

```plaintext
wholeURI ::= URI | XPointer
Xpointer ::= (URI)"#xpointer(/sbml/model" modelreference* ")"
modelReference ::= "/listOfSubmodels/model[@id=%22 SId "%22]"
```

The advantage is that parsers won’t be required to parse and execute the whole XPath syntax which may require SBML streams to be stored in DOM form for access by generic XPointer evaluators. However tools that can interpret the full XPath syntax would still be able to interpret the XLink attributes. This restricted form is unambiguous unlike some potential XPath strings.

**6 Object References**

Each instance structure has an implied content which reflects the content of the referenced Model structure. This section describes an scheme for referencing the structures implied by instances. This scheme is required so that links can be made between models. These links enable a model composed of instances to be literally more than a sum of its parts (see section 7).

The implied content of an instance doesn’t correspond directly to any XML structure. For example consider 2 instances which reference the same submodel. There doesn’t exist anywhere the 2 XML elements that represent the components that are implied by the 2 instances and thus it is not seem appropriate to use XML addressing schemes such as XPointer to reference them. Instead this section proposes a reference scheme specific to SBML Level 3.
The `ObjectRef` structure is used to refer to content implied by instances. The object referenced by a single `ObjectRef` structure depends on the context and the content of the structure's `object` field. The `object` field can refer to components including instances but not models. If the `ObjectRef` is not contained within another `ObjectRef` then the `object` field refers to a component in the immediate enclosing `Model` structure. In any context if the `object` field refers to an instance or a reaction the `ObjectRef` structure must contain a `subobject` attribute i.e. a nested `ObjectRef`.

Consider a `ObjectRef` structure enclosed inside another `ObjectRef` structure where the enclosing structure refers to an instance. The enclosed structure can refer to a component in the submodel referenced by that instance. If this referenced component is an instance then the process continues recursively building up a path to a component.

Consider a `ObjectRef` structure enclosed inside another `ObjectRef` structure where the enclosing structure refers to a reaction. The enclosed structure can refer only to parameters defined within the reaction.

This scheme deliberately cannot create references to instances, models or reactions.

In the following examples the outer `ObjectRef` is contained in arbitrarily named field `test`. All these examples are located in the top level model of the model example above. The first example refers to the top-level species `S1`.

```xml
<test object="S1"/>
```

The following example refers to the parameter of the reaction in the top level model

```xml
<test object="reaction_1">
  <subobject object="k3"/>
</test>
```

The remaining example in this section is based on the following instance occurring in the model

```xml
<listOfInstances>
  <instance id="innerA"
    xlink:type="simple"
    xlink:href="#xpointer(/sbml/model/listOfSubmodels/[@id=%22inner%22])"/>
</listOfInstances>
```

The following example refers to the species `S1` contained inside instance `innerA`.

```xml
<test object="innerA">
  <subobject object="S1"/>
</test>
```

### 6.1 Issue

Given the form of model composition given here we could use a simpler form consisting of an attribute containing a sequence of SID strings separated by whitespace. Such scheme would be difficult to extend to incorporate planned Level 3 features such as arrays.

### 7 Links

To enable models to be meaningfully composed linkages between instances need to be created. A model can contain a set of `Link` elements each of which contain 2 `ObjectRef` fields: `from` and `to`. These fields can only refer to ports, species, parameter and compartment components in the top level model and/or implied within instances.

A `Link` structure should interpreted as follows. The object referenced by the `from` field (`from object`) merges with the object referenced by the `to` field (`to object`) to become the same entity. These objects are called the referenced objects. The attribute values of the `to object` are replaced by those of the `from object`. Structures (e.g. `SpeciesReference`) that refer to either object effectively refer to the combined entity. Any assignment
rules that control the value of a from object replace those controlling a to object. The merged entities should obey all the semantic rules of SBML Level 2.

The interpretation of a Link structure referencing a port is described in section 8.

In any list of links in a model an object cannot be referenced by more than one to field in the list. i.e. an object can't be overloaded twice at the same level in the instance hierarchy. However it is possible for an object to be referenced by more than one from field in the list i.e. an object can overload more than one object. An object can't be referenced from a to field and a from field in the same list.

For example given the following instance in the model above.

```xml
<listOfInstances>
  <instance id="innerA"
    xlink:type="simple"
    xlink:href="#xpointer(/sbml/model/listOfSubmodels/model[@id=%22inner%22])"/>
</listOfInstances>
```

We can define a link from species S1 in the top level model to the species X0 in the innerA instance.

```xml
<listOfLinks>
  <link>
    <from object="S1"/>
    <to object="innerA">
      <subobject object="X0"/>
    </to>
  </link>
</listOfLinks>
```

The concentration of the combined object is 10. The combined object is produced by reaction 1 in the top level model and consumed by reaction 1 in instance innerA.

See section 9.1 for a complete example.

### 7.1 Link Validity

It should be obvious that for a composed model to work correctly some restrictions should be imposed on the linkages that can be made. It is not yet clear how far this should be defined as part of the language. Some parts of this section could be regarded as guidance for software that may wish to check the state of models during composition.

#### 7.1.1 Type correspondence

Components of different types (species, compartment or parameter) cannot be linked.

#### 7.1.2 Unit Correspondence

The unit system of SBML Levels 1 and 2 was designed with model composition in mind. Linked components can be checked to ensure that their assigned units match.

### 8 Ports

Although there is no consensus for it in the biochemical network modelling community some researchers believe that submodels need to have well defined interfaces to successfully support model composition. Briefly the arguments for having interfaces are as follows.

- An interface provides a “contract” between a submodel and models composed from that submodel. The contract can be maintained even when the submodel internals are significantly changed.
- The contract can be “implemented” by several alternative submodels with the same interface. These alternative submodels may use different simulation paradigms and/or encode different hypotheses for the modelled phenomena.
The interface facilitates the documentation of the function of the submodel from the perspective of a modeller wishing to reuse the submodel.

The counter argument is that any hierarchy in biochemical networks is only used to structure the human knowledge of the networks. A modeler can’t anticipate all the connections that may be discovered between submodels in those networks. What works in software forward engineering may not work in biochemical network reverse engineering.

This proposal doesn’t enforce the use of interfaces but does allow the definition of model interfaces and the linking to/from objects on an interface. Under this proposal it is valid to simultaneously use and ignore model interfaces.

Under this proposal a model’s interface is defined by the list of Port structures contained in the model. A Port structure has an object reference field which refers to a given component within the model (including components inside instances). The other fields on the Port structure are optional id and name fields. If the id attribute is not present then a Port structure simply indicates that the referenced object is part of the defined interface. If the id attribute is present then its value can be used as alternative alias for the referenced object. This alias is for use in object reference structures and in other SId fields. The alias is in the containing model’s component namespace.

The abstract optional boolean attribute on the Port structures indicates that the referenced object is “designed” to be referenced by the to field of a Link structure (it is an abstract port). The abstract field has only a documentation function: tools may or may not wish to raise errors when an abstract port is used as a from object.

A Link structure references a port by using the port’s id field value. The semantics of such a link are identical to a link that references the object referenced by the port.

9 Examples using Links

This section contains complete model examples using links.

9.1 Model Composition Without Ports

Figure 2 shows the important components and relationships of the following example model
Figure 2: The structure of the outer model. The rounded rectangles are compartments. The proper rectangles are model instances. Dashed arrows are links. Solid arrows are reactions.

```xml
<?xml version="1.0"?>
<sbml xmlns="http://www.sbml.org/sbml/level3" version="1" level="3">
  <model id="outer">
    <listOfCompartments>
      <compartment id="compartmentOne" volume="1"/>
    </listOfCompartments>
    <listOfSpecies>
      <species id="S1" initialAmount="10" compartment="compartmentOne"/>
      <species id="X0" initialAmount="0" compartment="compartmentOne"/>
    </listOfSpecies>
    <listOfReactions>
      <reaction id="reaction_1" reversible="false">
        <listOfReactants>
          <speciesReference species="X0" stoichiometry="1"/>
        </listOfReactants>
        <listOfProducts>
          <speciesReference species="S1" stoichiometry="1"/>
        </listOfProducts>
      </reaction>
    </listOfReactions>
  </model>
</sbml>
```

9
Figure 3 shows the structure implied by the outer example model.

![Diagram of the outer model]

**Figure 3: The implied form of the outer model**

### 9.2 Model Composition with Ports

Figure 4 shows the important components and relationships of the following example model. The implied structure of this model is the same as that shown in figure 3.

```xml
<?xml version="1.0"?>
<sbml xmlns="http://www.sbml.org/sbml/level3" version="1" level="3">
  <model id="outer_with_ports">
    <listOfCompartments>
      <compartment id="compartmentOne" volume="1"/>
    </listOfCompartments>
  </model>
</sbml>
```
Figure 4: The structure of the outer_with_ports model

```xml
<species id="S1" initialAmount="10" compartment="compartmentOne">
  <species id="X0" initialAmount="0" compartment="compartmentOne">
    </listOfSpecies>
    <listOfReactions>
      <reaction id="reaction_1" reversible="false">
        <listOfReactants>
          <speciesReference species="X0" stoichiometry="1"/>
        </listOfReactants>
        <listOfProducts>
          <speciesReference species="S1" stoichiometry="1"/>
        </listOfProducts>
      </reaction>
      <listOfReactions>
        <listOfSubmodels>
          <model id="inner">
            <compartment id="compartmentOne" volume="1"/>
          </listOfCompartments>
          <listOfSpecies>
            <species id="S1" initialAmount="0" compartment="compartmentOne">
              <species id="X0" initialAmount="0" compartment="compartmentOne">
                </listOfSpecies>
                <listOfReactions>
                  <reaction id="reaction_1" reversible="false">
                    <listOfReactants>
                      <speciesReference species="X0" stoichiometry="1"/>
                    </listOfReactants>
                    <listOfProducts>
                      <speciesReference species="S1" stoichiometry="1"/>
                    </listOfProducts>
                  </reaction>
                  <listOfReactions>
                    <listOfPorts>
                      <port object="X0"/>
                      <port id="C" object="compartmentOne"/>
                    </listOfPorts>
                  </listOfReactions>
                </model>
              </listOfSubmodels>
            </listOfReactions>
          </listOfReactions>
        </listOfPorts>
      </listOfReactions>
    </listOfPorts>
  </listOfReactions>
</listOfSpecies>
</listOfReactions>
</listOfSubmodels>
</model>
```
10 Direct Links

One of the problems with the proposal as described in the previous sections is that some forms of model composition that ought to be simple are less than straightforward. For example consider the composition of two models together simply by creating a reaction between them (an outer model contains two instances and a reaction). Using the above form we would have to create new species for all the reaction’s products, reactants and modifiers. These species would be linked to the appropriate species inside the instances. A more appropriate approach would be to enable the SimpleSpeciesReference structures of the reaction to reference the species inside the instances directly. This section proposed some modifications to existing structures of SBML to facilitate this kind of direct link.

Under this proposal, as shown in figure ??, types like Species have an additional field of type ObjectRef which can refer to objects inside instances. This field is an alternative to an existing SId field.

On Compartment the ObjectRef field outsideLink is an alternative to the SId field outside. Similarly on Species compartmentLink is an alternative to compartment, on SimpleSpeciesReference speciesLink is an alternative to species and on SingleVariableRule variableLink is an alternative to the variable. In each of the types one and only one of these fields (SId or ObjectRef) can have a value. For example Species structures can contain a value for compartment or compartmentLink but not both. It is not possible for both the SId or ObjectRef fields to be omitted except in the case of the outside and outsideLink fields.

In each case the ObjectRef field can refer to an object of the appropriate type inside any instance, for example Compartment in the case of compartmentLink and outsideLink, Species in the case of speciesLink. Section ?? has an examples of a complete model using these fields.

The following example fragment is of a reaction transforming species f in instance a into species e in instance b.

```xml
<reaction id="reaction_1" reversible="false">
  <listOfReactants>
    <speciesReference stoichiometry="1">
      <speciesLink object="a"/>
      <objectRef object="f"/>
    </speciesReference>
  </listOfReactants>
  <listOfProducts>
    <speciesReference stoichiometry="1">
      <speciesLink object="b"/>
      <speciesReference object="e"/>
    </speciesReference>
  </listOfProducts>
</reaction>
```
10.1 Direct links using ports

These new ObjectRef fields can reference port objects of the appropriate type in which case the direct link is equivalent to a direct link to the object referenced by the port structure.

10.2 Direct links in math expressions

To support direct links in math expressions we we use a SBML specific operator via the the MathML csymbol element. This SBML operator has the URI, http://www.sbml.org/symbols/instanceselector. The instanceselector that takes an 2 arguments: the first is an instance of a submodel the second argument is an object inside that instance.

For example we can extend the previous fragment to contain a simple kinetic law referring to the reactant.

```xml
<reaction id="reaction_1" reversible="false">
  <listOfReactants>
    <speciesReference stoichiometry="1">
      <speciesLink object="a">
        <objectRef object="f"/>
      </speciesLink>
    </speciesReference>
  </listOfReactants>
  <listOfProducts>
    <speciesReference stoichiometry="1">
      <speciesLink object="b">
        <objectRef object="e"/>
      </speciesLink>
    </speciesReference>
  </listOfProducts>
  <kineticLaw>
    <math>
      <apply>
        <times/>
        <cn>0.1</cn>
        <apply>
          <csymbol encoding="SBML"
            definitionURL="http://www.sbml.org/symbols/instanceselector"/>
          <ci>a</ci>
          <ci>f</ci>
        </apply>
      </apply>
    </math>
  </kineticLaw>
</reaction>
```

11 Example of Direct links

The following model consists of two instances of a simple submodel. The submodel contains two species with a reaction between them. The top level model contains a reaction which links the two submodels together. Figure 5 shows the important components and relationships of the model.

13
Figure 5: The structure of the outer2 model.

```xml
<?xml version="1.0"?>
<sbml xmlns="http://www.sbml.org/sbml/level3" version="1" level="3">
  <model id="outer2">
    <listOfReactions>
      <reaction id="reaction_1" reversible="false">
        <listOfReactants>
          <speciesReference stoichiometry="1">
            <speciesLink object="a">
              <subobject object="f"/>
            </speciesLink>
          </speciesReference>
        </listOfReactants>
        <listOfProducts>
          <speciesReference stoichiometry="1">
            <speciesLink object="b">
              <subobject object="e"/>
            </speciesLink>
          </speciesReference>
        </listOfProducts>
        <kineticLaw>
          <math>
            <apply>
              <times/>
              <cn>0.1</cn>
            </apply>
          </math>
        </kineticLaw>
      </reaction>
    </listOfReactions>
    <listOfSubmodels>
      <model id="X">
        <listOfCompartments>
          <compartment id="compartmentOne" volume="1"/>
        </listOfCompartments>
      </model>
    </listOfSubmodels>
  </model>
</sbml>
```
Figure 6 shows the model implied by outer2.

The array proposal contains a more extensive model that combines array and model composition features.
References


