

First...

SBML ODE Solver

- **Continuous Deterministic Simulator**
- **Uses Sundials CVODE**
- **Open source project**
- **Portable**
 - C implementation
 - Windows, Linux
- **Library – SOSlib**
 - API
- **Command Line Application**
- **LPGL**
 - can be used in commercial projects
- **Application note in Bioinformatics**
- **Project started by**
 - Christoph Flamm and Rainer Machne
 - University of Vienna, Austria

SBML ODE Solver: Features

- **Pretty Dam Fast**
 - For ~ 80 variable complex model with events
 - SBML ODE Solver took 5.922 seconds
 - Matlab ODE15s took 9.55 seconds
 - Jarnac took 91.078 seconds
- **Validated against SBML semantic test suite**
- **SBML Features supported:**
 - Reactions
 - Assignment Rules
 - Rate Rules
 - Functions
 - Almost all the MathML subset
- **Partial support for Events**
 - Timestep dependant
 - good enough in practice
- **API**
 - Can optimize model construction in multiple simulation scenarios
 - Can vary variables and parameters during simulation

Future Plans of SBML ODE Solver

- **Optionally use CVODES**
 - sensitivity analysis
- **DAE support with IDA**
- **Clean separation of library and application**
 - Improved API
- **Even longer term**
 - Precise Event Determination
 - Parameter estimation

Draft Specification of SBML Level 2 Version 2

Andrew Finney

Physiomics PLC, UK

This presentation follows
draft specification emailed to
sbml-discuss mailing list
recently

Overview

- Introduction
- Removal of predefined annotation namespaces
- `definitionURL` on `SBase`
- Nested Unit Definitions
- Dimensionless Units
- Substance Mass Derived
- `SpeciesType`
- Constraints
- Initial Assignment Structures
- Reaction Symbols
- Use of `id` Attribute on `SimpleSpeciesReference`
- Dependant Variables semantics
- Extended example
 - putting it all together
- Conclusion

Introduction

- Based on discussion at last forum and on sbml-discuss since then
- Detailed features have been discussed in the majority of cases
- In other cases only requirements were discussed
 - initial structures are presented here
- Ideally we can reach a consensus here which can be confirmed by a web survey

Removal of Pre-defined Annotation Namespaces

- SBML Level 2 Version 1 contain a set of pre-defined namespaces for annotations
 - e.g.
`http://www.sbml.org/2001/ns/cellerator`
- **Removed because:**
 - No one used them
 - Too many applications use SBML to warrant a list

definitionURL on SBase

- SBase structure has new attribute `definitionURL`
 - field name taken to be consistent with MathML
 - contains URI!
 - not necessarily a URL
 - resource reference
 - major application
 - labeling of structures with controlled vocabulary terms
 - formal interpretation
 - 'is a' relationship between SBML object

Example of definitionURL

```
<species  
  id="MAPK"  
  definitionURL=  
  "http://www.foobar.com/protein"/>
```

“MAPK is a protein”

Nested Unit Definitions

- In Level 2 Version 1 all unit definitions have to be composed from fundamental units
 - For example
 - You can define minutes as 60 seconds
 - But not hours as 60 minutes
- We fix this by allowing the **kind** field of **Unit** structures be able to refer to other unit definitions as well as the **unitkind** enumeration

Nested Units Example

```
<unitDefinition id="Minute">
  <listOfUnits>
    <unit kind="second" multiplier="60"/>
  </listOfUnits>
</unitDefinition>
<unitDefinition id="Hour">
  <listOfUnits>
    <unit
      definition="Minute" multiplier="60"/>
  </listOfUnits>
</unitDefinition>
```

Other Minor Units flexibility

- The built in units can be dimensionless
 - substance, length, area, volume, time etc
- Version 2 allows Mass units to be assigned to the substance built in unit
 - To support a few models including:
 - Domach, M. M., Leung, S. K., Cahn, R. E., Cocks, G. G., and Shuler, M. L. (2000). Computer model for glucose-limited growth of a single cell of *Escherichia coli* B/r-A. *Biotechnology and Bioengineering*, 67:827-840.
 - Castellanos, M., Wilson, D. B., and Shuler, M. L. (2004). A modular minimal cell model: Purine and pyrimidine transport and metabolism. *PNAS*, 101:6681-6686.

SpeciesType

- `SpeciesType` structures allow the formal indication that species located in more than one compartment are in fact pools of the same chemical entity type
- In this proposal the `model` class is extended to have a list of species types
- `SpeciesType` structures are entirely optional
- `SpeciesType` structures do not affect any representation of the model dynamics
- `Species` structures still used to declare variables
- `Species` structures can optionally refer to `SpeciesType` structures

SpeciesType and Species

- **Model**
 - new optional list of **SpeciesType** structures
- **SpeciesType**
 - one mandatory field **id**
 - one optional field **name**
- **Species**
 - one new optional **SI** field **speciesType**
- **Semantic Rules**
 - the **speciesType** field must contain the value of a **SpeciesType id** field
 - there cannot be more than one **Species** structure with a given pair of **speciesType** and **compartment** values
 - there can't be more than one pool in a compartment containing a specific type of chemical entity

SpeciesType Example

```
<model id="malate_aspartate_shuttle">
  <listOfCompartments>
    <compartment id="Cytosol"/>
    <compartment id="Mitochondrial_Matrix"/>
  </listOfCompartments>
  <listOfSpeciesTypes>
    <speciesType id="Aspartate"/>
  </listOfSpeciesTypes>
  <listOfSpecies>
    <species
      id="Aspartate_in_Cytosol"
      speciesType="Aspartate"
      compartment="Cytosol"/>
    <species
      id="Aspartate_in_Mitochondrial_Matrix"
      speciesType="Aspartate"
      compartment="Mitochondrial_Matrix"/>
  </listOfSpecies>
</model>
```

Constraint – Concept

- **Concept**

- Define constraints that enable the detection of internal inconsistencies in a model and/or external perturbations of variables and parameters which render a model invalid.
- Simply math expressions that are either true or false given some subset of variables and constant defined in the model.
- Constraints are not structured to facilitate the definition of the time course behaviour of the modelled system but may facilitate other types of analyses e.g. flux balance analysis.

- **Requirement**

- Constraints must be clearly separated from other structures that are used directly in defining simulation/time course behaviour.
- To facilitate the use of constraints as ‘assertions’ an error string can be optionally associated with the constraints.

- **Example**

- Define quantitatively the assumption in a rate law that the product concentration is much lower than that of the enzyme
 - If the product concentration becomes large enough to render the rate law invalid during a simulation the simulator can notify the user

Constraint Structure

- Model has an optional list of Constraint structures
- A Constraint structure has
 - mandatory `math` boolean MathML field
 - optional `message` HTML field
- No other universal semantic rules
- Semantics only defined loosely in simulation
 - when the constraint becomes false
 - the message may be displayed to the user
 - ideally the simulation time of violation would be reported
- Other analyses may be driven by the constraints

Constraint - Example

```
<constraint>
  <mathml:math>
    <apply>
      <and/>
      <apply><lt/><cn> 1 </cn><ci> S1 </ci>
      </apply>
      <apply><lt/><ci> S1 </ci><cn> 100 </cn>
      </apply>
    </apply>
  </mathml:math>
  <message>
    <xhtml:p>Species S1 is out of range</xhtml:p>
  </message>
</constraint>
```

InitialAssignment

- Allows the calculation of the initial value of a symbol from the initial value of other symbols
 - symbol may be constant or variable
- **Model** contains optional list of **InitialAssignment** structures
- **InitialAssignment** structure contains
 - mandatory **symbol** **id** field
 - mandatory **math** **MathML** field
 - must return numeric result - not boolean!
- **Semantic Rules**
 - **symbol** field must contain the value of a species, compartment or parameter **id** field
 - unlike assignment rules there are no other constraints
 - only executed once with the results applied at t=0

InitialAssignment Example

```
<initialAssignment symbol="x">  
  <mathml:math  
    <apply>  
      <times/>  
        <ci> y </ci>  
        <cn> 2 </cn>  
    </apply>  
  </mathml:math>  
</initialAssignment>
```

Reaction Symbols

- The `id` value of a reaction structure can be used as a symbol value in MathML expressions
- Represents the flux of the reaction
 - the direct result of a rate law
 - substance / time units
- Cannot assign a value to the symbol
 - e.g. can't occur in `variable` field of an `AssignmentRule`
- Other Semantic Rules
 - can't appear in kinetic laws

Reaction Symbol Example

```
<reaction id="foo">...</reaction>
<reaction id="voodoo">..</reaction>
...
<algebraicRule>
  <mathml:math>
    <apply>
      <add/>
      <ci>foo</ci>
      <ci>voodoo</ci>
    </apply>
  </mathml:math>
</algebraicRule>
```

`id` on `SimpleSpeciesReference`

- Create new `id` attribute on `SimpleSpeciesReference`
 - Has type `SId`
- Required by diagram layout proposal
- Value is unique amongst all `id` values declared at the global level
- Optional `name` attribute of type `string` as well
- In addition to existing `species` attribute that refers to a species

Extended Example: A Flux Balance Model

```
<model>
  <listOfCompartments>
    <compartment id="cytosol"/>
    ...
  </listOfCompartments>
  <listOfSpeciesTypes>
    <speciesType id="atp"/>
    ...
  </listOfSpeciesTypes>
  <listOfSpecies>
    <species
      id="atp_in_cytosol"
      compartment="cytosol"
      speciesType="atp"/>
    ...
  </listOfSpecies>
```

Extended Example: A Flux Balance Model [continued]

```
<listOfConstraints>
  <constraint
    definitionURL="http://biomodels.net/sbo/fluxConstraint">
    <mathml:math>
      <apply>
        <and/>
        <apply>
          <gt;<cn>0</cn><ci>Glycolysis</ci></apply>
          <apply>
            <gt;<ci>Glycolysis</ci><infinity/></apply>
        </apply>
      </mathml:math>
    </constraint>
    ...
  </listOfConstraints>
  <listReactions>
    <reaction id="Glycolysis">
      ...
    </reaction>
  </listOfReactions>
</model>
```

Conclusion

- Click to add vote

List of Major Changes

- Removal of predefined annotation namespaces
- `definitionURL` on `SBase`
- Nested Unit Definitions
- Dimensionless Units
- Substance Mass Derived
- `SpeciesType`
- Constraints
- Initial Assignment Structures
- Reaction Symbols
- Use of `Id` Attribute on `SimpleSpeciesReference`
- Dependant Variables semantics

Flux and Variable Bounds

- Its clear that there are non-spatial analyzes of biochemical reaction networks which can be driven by mathematical structures that are not yet part of the SBML standard.
- These analyzes can use the reaction network contained in a SBML model as a starting point yet they require additional information as input.
- Some analyzes of biochemical reaction networks operate on bounds of various properties of a network rather than absolute values or expressions.
- Bounds need to be placed on initial values, variables, parameters and reaction rates.
- See UML diagram in document
- For flux we allow reaction ids to be a symbol
 - In MathML as well