

Properties of SBML documents and tools – capabilities

Jörg R. Weimar
Institute of Scientific Computing
Technical University Braunschweig
D-38092 Braunschweig, Germany
j.weimar@jweimar.de

May 26, 2003

1 Introduction

This is a proposal for a recommendation on the usage of a controlled vocabulary for the description of tools and models using the systems biology markup language (SBML).

Different tools for handling SBML files have different capabilities of which constructs in SBML they support. A controlled way of specifying these capabilities is needed to find out if a tool can handle a certain model or class of models. Conversely, models could specify which capabilities they need. Apart from capabilities, specifications on the assumptions that a tool makes are also necessary.

A set of filters could be developed to overcome some limitations of certain tools. Such a filter would need to know the capabilities of the target tool.

An automatic tool could be written to find out which SBML features are used in a model. This tool could also try to make some inferences about the assumptions, but cannot necessarily resolve all assumptions.

2 List of features

A preliminary list of features of SBML that tools might or might not support follows:

2.1 SBML level 1

- Level 1 Version 1
- Level 1 Version 2
- Level 1 Version 3
- rules
- scalar AssignmentRules
- rate AssignmentRules

- AlgebraicRule
- KineticRateLaw
- Formulas
- only Mass Action
- predefined math functions
- predefined rate law functions
- local Parameters
- global Parameters
- Units
- Compartment volume not 1

2.2 SBML level 2

Additional features in SBML level 2:

- Level 2
- Level 2 Version 1
- Level 2 Version 2
- Level 2 Version 3
- Level 2 minus MathML
- logical operators in MathML
- function definition
- ModifierSpeciesReference
- Delays
- Events
- Names (in addition to IDs)
- initial concentration

2.3 SBML level 3

Additional features in SBML level 3:

- submodels
- Instances
- Arrays
- Sparse arrays
- dynamic arrays
- Parameter sets
- initialAssignmentRule
- Ports
- Links to Ports only

2.4 Notes and annotations

- XHTML notes
- Graph layout annotation
-

The above list of features is necessarily incomplete. Some features could be structured hierarchically, some are mutually exclusive, and especially regarding notes and annotations, extension to other features will be necessary.

3 Properties of tools regarding the features

Looking at the list of features, we can try to find how those words could be used in describing a tool or a model.

3.1 Models

For models, one could state that

This model uses **feature**.

or

This model does not use **feature**.

and

This model is created with assumption **feature**.

3.2 Tools

The most important properties for tools are whether a tool can read SBML files and whether the models written by the tool contain (or can contain) a feature. Example sentences would be

This tool can correctly read and process models that use **feature**.

This tool cannot read models that use **feature**.

This tool creates models that may contain **feature**.

The models that this tool creates never contain **feature**.

This tool can optionally create all models without using **feature**.

This tool can convert models with **feature** into models without **feature**.

4 Formalizing

It is desirable to create a vocabulary in which features and properties are orthogonal, in that each feature can be combined with each property and the statement would make sense.

An example list of properties would be: "Never" "Read only", "Read and Write", "Used", "Not used".

A capability of a tool is then a pair of feature and property. This could be formalized as an xml schema, such that the capabilities of a model could be described by the following XML fragment inside an annotation node of the SBML file:

```
<annotation xmlns:cap="http://www.sbml.org/2002/ns/capabilities/">
  </cap:capability cap:feature="Delays" cap:property="Not used">
  </cap:capability cap:feature="Events" cap:property="Used">
</annotation>
```

4.1 RDF

The information about models is a typical example of metadata. Therefore one should investigate whether the Resource Description Format (RDF) can appropriately be used for formalizing the capabilities.

TO BE CONTINUED