SBML Level 3 Package Specification

Math V2Extras

Sarah M Keating
skeating@ebi.ac.uk
European Bioinformatics Institute (EMBL-EBI)
Hinxton
Cambridgeshire, GB

Others

Version 1, Release 1

01 April 2017

This is a draft specification for the SBML Level 3 package called "`math-v2extras`". It is not a normative document. Please send feedback to the Package Working Group mailing list at sbml-math@lists.sourceforge.net.

The latest release, past releases, and other materials related to this specification are available at

This release of the specification is available at

Contents

1 Introduction 3
- 1.1 Proposal corresponding to this package specification . . . 3
- 1.2 Package dependencies . . . 3
- 1.3 Document conventions . . . 3

2 Background and context 4

3 Package syntax and semantics 5
- 3.1 Namespace URI and other declarations necessary for using this package . . . 5
- 3.2 The quotient operator . . . 5
  - 3.2.1 Properties of `quotient` . . . 5
- 3.3 The rem operator . . . 6
  - 3.3.1 Properties of `rem` . . . 6
- 3.4 The max operator . . . 6
  - 3.4.1 Properties of `max` . . . 6
  - 3.4.2 Other package considerations . . . 6
- 3.5 The min operator . . . 6
  - 3.5.1 Properties of `min` . . . 7
  - 3.5.2 Other package considerations . . . 7
- 3.6 The implies operator . . . 7
  - 3.6.1 Properties of `implies` . . . 7
- 3.7 The csymbol rateOf . . . 7
  - 3.7.1 Attributes on csymbol rateOf . . . 7
  - 3.7.2 Properties of `rateOf` . . . 7
  - 3.7.3 Determining the `rateOf` for a symbol . . . 8
  - 3.7.4 Special considerations of `rateOf` in SBML Level 3 Core . . . 8

4 Examples 9
- 4.1 Sample . . . 9

A Validation of SBML documents 10
- A.1 Validation and consistency rules . . . 10

Acknowledgments 11

References 12

1 Introduction

The SBML Level 3 Math V2Extras package offers support for the additional math constructs defined in the SBML Level 3 Version 2 Core specification. Using this package will enable users to incorporate these constructs into an SBML Level 3 Version 1 Core document.

1.1 Proposal corresponding to this package specification

This specification for SBML Level 3 Package Specification for Math V2Extras, Version 1 is based on the proposal located at the following URL:

http://sbml.org/images/e/ed/Proposal_multiple_small_additional_math_packages.pdf

1.2 Package dependencies

This package does not depend on any other package.

1.3 Document conventions

UML 1.0 (Unified Modeling Language; Eriksson and Penker 1998; Oestereich 1999) notation is used in this document to define the constructs provided by this package. Colors in the diagrams carry the following additional information for the benefit of those viewing the document on media that can display color:

- *Black*: Items colored black in the UML diagrams are components taken unchanged from their definition in the SBML Level 3 Core specification document.
- *Green*: Items colored green are components that exist in SBML Level 3 Core, but are extended by this package. Class boxes are also drawn with dashed lines to further distinguish them.
- *Blue*: Items colored blue are new components introduced in this package specification. They have no equivalent in the SBML Level 3 Core specification.

The following typographical conventions distinguish the names of objects and data types from other entities; these conventions are identical to the conventions used in the SBML Level 3 Core specification document:

AbstractClass: Abstract classes are never instantiated directly, but rather serve as parents of other classes. Their names begin with a capital letter and they are printed in a slanted, bold, sans-serif typeface. In electronic document formats, the class names defined within this document are also hyperlinked to their definitions; clicking on these items will, given appropriate software, switch the view to the section in this document containing the definition of that class. (However, for classes that are unchanged from their definitions in SBML Level 3 Core, the class names are not hyperlinked because they are not defined within this document.)

Class: Names of ordinary (concrete) classes begin with a capital letter and are printed in an upright, bold, sans-serif typeface. In electronic document formats, the class names are also hyperlinked to their definitions in this specification document. (However, as in the previous case, class names are not hyperlinked if they are for classes that are unchanged from their definitions in the SBML Level 3 Core specification.)

`SomeThing`, `otherThing`: Attributes of classes, data type names, literal XML, and tokens *other* than SBML class names, are printed in an upright typewriter typeface.

For other matters involving the use of UML and XML, this document follows the conventions used in the SBML Level 3 Core specification document.

2 Background and context

Currently SBML restricts the subset of MathML that is deemed 'valid' for use within an SBML model. It was considered that L3 packages that needed additional math would add this as required. However, this limits the use of the additional math to the use of the package - which in some cases may not be necessary. For example the arrays package has suggested including the mathematical operation 'mean' - but it would be perfectly feasible to use 'mean' without using arrays.

Discussions of additional math constructs that should be added to core SBML inevitably introduces difference of opinion as there are varying views on what is considered useful and/or necessary.

It has also been suggested that we relax the restriction and just allow all of MathML to be used within SBML. However, learning from the experience of CellML, this may produce a situation where many software packages do not support all mathematical operations. This can lead to reduced interoperability with some models only being simulatable by one software package.

The conclusion of the discussions was that we produce a number of small math only packages that group together math constructs that would be added by that package e.g. statistical functions, array functions, linear algebra functions. Each package would have its own sbml namespace and use the required attribute to indicate that it has been used in a fashion consistent with the current use of packages.

These packages can be used with all SBML Level 3 versions and thus facilitates modelers in their need to use mathematical constructs whilst maintaining a position where software supports this and we can continue to have the level of interoperability that SBML has currently achieved.

TODO: pkg with l3v2 math

3 Package syntax and semantics

This section contains a definition of the syntax and semantics of the SBML Level 3 Package Specification for Math V2Extras, Version 1. The Math V2Extras package involves the following MathML 2.0 W3C (2000) constructs.

- *arithmetic operators*: `quotient`, `max`, `min`, `rem`
- *logical operators*: `implies`
- *csymbol*: ‘‘`http://www.sbml.org/sbml/symbols/rateOf`’’

3.1 Namespace URI and other declarations necessary for using this package

Every SBML Level 3 package is identified uniquely by an XML namespace URI. For an SBML document to be able to use a given Level 3 package, it must declare the use of that package by referencing its URI. The following is the namespace URI for this version of the Math V2Extras package for SBML Level 3 Core:

“`http://www.sbml.org/sbml/level3/math/v2extras/version1`”

In addition, SBML documents using a given package must indicate whether the package can be used to change the mathematical interpretation of a model. This is done using the attribute `required` on the `<sbml>` element in the SBML document. For the Math V2Extras package, the value of this attribute must be “`true`”, because the use of the Math V2Extras package will change the mathematical meaning of a model.

The following fragment illustrates the beginning of a typical SBML model using SBML Level 3 Version 1 Core and this version of the Math V2Extras package:

```
<?xml version="1.0" encoding="UTF-8"?>
<sbml xmlns="http://www.sbml.org/sbml/level3/version1/core" level="3" version="1"
      xmlns:math_xxx="http://www.sbml.org/sbml/level3/math/v2extras/version1"
      math_xxx:required="true">
```

The use of the package namespace differs from other Level 3 packages. In this package it is only be used to indicate the use of additional constructs. The MathML constructs themselves remain in the MathML namespace in accordance with the current use of MathML within SBML.

TODO: what about the number <-> boolean issue

3.2 The quotient operator

The `quotient` element is a binary arithmetic operator used for division modulo a particular base. When the quotient operator is applied to integer arguments a and b, the result is the "quotient of a divided by b". That is, quotient returns the unique integer q such that a = q b + r. (In common usage, q is called the quotient and r is the remainder.)

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
    <apply>
        <quotient/>
        <ci> a </ci>
        <ci> b </ci>
    </apply>
</math>
```

3.2.1 *Properties of* `quotient`

- binary function
- arguments must be type `int`
- result is `int`

3.3 The rem operator

The `rem` element is a binary arithmetic operator that returns the "remainder" of a division modulo a particular base. When the rem operator is applied to integer arguments a and b, the result is the "remainder of a divided by b". That is, rem returns the unique integer, r such that a = q b+ r, where r < q. (In common usage, q is called the quotient and r is the remainder.)

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
    <apply>
        <rem/>
        <ci> a </ci>
        <ci> b </ci>
    </apply>
</math>
```

3.3.1 Properties of `rem`

- binary function
- arguments must be type `int`
- result is `int`
- rem(a, 0) is undefined

3.4 The max operator

The `max` element is an n-ary arithmentic function used to compare the values of the arguments. It returns the maximum of these values.

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
    <apply>
        <max/>
        <ci> a </ci>
        <ci> b </ci>
        <ci> c </ci>
    </apply>
</math>
```

3.4.1 Properties of `max`

- nary function
- max() is undefined
- arguments must all be comparable if the result is to be well defined

3.4.2 Other package considerations

- **arrays**: When the arguments to the `max` function represent an array of values, the result is the maximum value of the values in the array.

3.5 The min operator

The `min` element is an n-ary arithmentic function used to compare the values of the arguments. It returns the minimum of these values.

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
    <apply>
        <min/>
        <ci> a </ci>
        <ci> b </ci>
```

```
        <ci> c </ci>
    </apply>
</math>
```

3.5.1 Properties of `min`

- nary function
- min() is undefined
- arguments must all be comparable if the result is to be well defined

3.5.2 Other package considerations

- **arrays**: When the arguments to the `min` function represent an array of values, the result is the minimum value of the values in the array.

3.6 The implies operator

The `implies` element is a binary logical function used to represent the boolean relational operator "implies".

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
    <apply>
        <implies/>
        <ci> a </ci>
        <ci> b </ci>
    </apply>
</math>
```

3.6.1 Properties of `implies`

- binary function
- arguments must be type `boolean`
- result is `boolean`

3.7 The csymbol rateOf

SBML uses the MathML `csymbol` element to denote certain built-in mathematical entities without introducing reserved names into the component identifier namespace. Math V2Extras package introduces the `rateOf` `csymbol`.

The intent of this `csymbol` is to provide a means for models to refer to quantities that must naturally be computed as part of doing a dynamical analysis of a model. The *rateOf* `csymbol` is not intended to provide full numerical differentiation capabilities but to represent the instantaneous rate of change, with respect to time, of an entity in the model.

3.7.1 Attributes on csymbol rateOf

- The `encoding` attribute of `csymbol` must be set to "`text`".
- The `definitionURL` should be set to "`http://www.sbml.org/sbml/symbols/rateOf`".

3.7.2 Properties of `rateOf`

- unary function
- argument must be type `SId`
- result is `double`
- rateOf(a) = 0; if `a` is declared "`constant`"

3.7.3 Determining the `rateOf` for a symbol

1. The *rateOf* for a symbol whose `SId` appears as the `variable` of a **RateRule** is the numerical value of that **RateRule**, using the current values of all symbols referenced in the rule's formula.

2. The *rateOf* for the *amount* of a **Species** having attribute `boundaryCondition`="`true`" and appearing in one or more reactions can be calculated from the stoichiometries and **KineticLaw** <math>of every **Reaction** in which the species appears, plus appropriate `conversionFactor` values. If the *species quantity* is in terms of its *concentration*, the rate must be converted by the `size` of the **Compartment** in which it appears, which may itself be changing in time. This can be calculated as follows, where $[x]$ is the concentration of species X, x the amount of species X, and V the size of the compartment in which species X is located:

$$\begin{aligned}\frac{d[x]}{dt} &= \frac{d(x/V)}{dt} \\ &= \frac{1}{V}\cdot\frac{dx}{dt} + x\cdot\frac{d(1/V)}{dt} \\ &= \frac{1}{V}\cdot\frac{dx}{dt} - \frac{x}{V^2}\cdot\frac{dV}{dt}\end{aligned}$$

   When dV/dt is equal to zero, the final term in the last line is zero.

In simulations that progress in a stepwise fashion, such as stochastic simulations, the *rateOf* `csymbol` is still calculated as above, from any appropriate **RateRule** or **KineticLaw**. This effectively means that for stepwise simulations, the *rateOf* indicates the expected average rate of change of the corresponding symbol over time, even when the actual rate of change may be zero or discontinuous.

3.7.4 Special considerations of `rateOf` in SBML Level 3 Core

- The allowable identifiers for use with *rateOf* in SBML Level 3 Core are restricted to those of **Compartment**, **LocalParameter**, **Parameter**, **Species**, and **SpeciesReference** objects in the enclosing model. Note that *rateOf* is not allowed for **Reaction** objects, because their identifiers already represent the rate of change of the reaction, and calculating second derivatives is beyond the scope of this construct. Likewise, there is no sensible meaning to be given to the *rateOf* of a **FunctionDefinition**, **Event**, **Priority**, **Delay**, or other SBML entities.

- An object whose `SId` identifier appears as the `variable` of an **AssignmentRule**, or which is calculated from an **AlgebraicRule**, may not be referenced by the *rateOf* `csymbol`. Similarly, it is also not valid to use the *rateOf* `csymbol` to reference a **Species** with a `hasOnlySubstanceUnits` attribute value of "`false`" and whose `compartment` appears as the `variable` of an **AssignmentRule** or whose `size` is calculated from an **AlgebraicRule**. In other words, anything whose value is directly or indirectly determined by an algebraic rule or an assignment rule is excluded.

- In the event of a discontinuity, such as might happen due to an **Event**, a `piecewise` function, the beginning of a time course simulation (i.e., at $t = 0$), or due to a new construct defined in a package, the rate of change is defined as the right-handed *rateOf* for the symbol, that is, the derivative with respect to time of the symbol moving forward in time from the current time, and not the derivative with respect to time from the recent past up until the current time. Thus, the *rateOf* of a symbol will always be calculable from the set of current values of symbols in the model. No **Event** can affect the *rateOf* for a symbol except indirectly.

TODO: any packages that have things that might use rateOf

4 Examples

This section contains examples employing the SBML Level 3 Package Specification for Math V2Extras, Version 1.

4.1 Sample

A Validation of SBML documents

A.1 Validation and consistency rules

This section summarizes all the conditions that must (or in some cases, at least *should*) be true of an SBML Level 3 Version 1 model that uses the Math V2Extras package. We use the same conventions that are used in the SBML Level 3 Version 1 Core specification document. In particular, there are different degrees of rule strictness. Formally, the differences are expressed in the statement of a rule: either a rule states that a condition *must* be true, or a rule states that it *should* be true. Rules of the former kind are strict SBML validation rules—a model encoded in SBML must conform to all of them in order to be considered valid. Rules of the latter kind are consistency rules. To help highlight these differences, we use the following three symbols next to the rule numbers:

☑ A checked box indicates a *requirement* for SBML conformance. If a model does not follow this rule, it does not conform to the Math V2Extras package specification. (Mnemonic intention behind the choice of symbol: "This must be checked.")

▲ A triangle indicates a *recommendation* for model consistency. If a model does not follow this rule, it is not considered strictly invalid as far as the Math V2Extras package specification is concerned; however, it indicates that the model contains a physical or conceptual inconsistency. (Mnemonic intention behind the choice of symbol: "This is a cause for warning.")

★ A star indicates a strong recommendation for good modeling practice. This rule is not strictly a matter of SBML encoding, but the recommendation comes from logical reasoning. As in the previous case, if a model does not follow this rule, it is not considered an invalid SBML encoding. (Mnemonic intention behind the choice of symbol: "You're a star if you heed this.")

The validation rules listed in the following subsections are all stated or implied in the rest of this specification document. They are enumerated here for convenience. Unless explicitly stated, all validation rules concern objects and attributes specifically defined in the Math V2Extras package.

General rules about this package

math-v2extras-10101 ☑ To conform to the Math V2Extras package specification for SBML Level 3, an SBML document must declare "`http://www.sbml.org/sbml/level3/math/v2extras/version1`" as the XMLNamespace that permits the use of the elements detailed in this package in the MathML Namespace. (Reference: SBML Level 3 Package Specification for Math V2Extras, Version 1 Section 3.1 on page 5.)

General rules for MathML content

TODO: add rules for each of the math elements used in this package: type and number of arguments; any specific restrictions etc

Acknowledgments

Work such as this does not take place in a vacuum; many people contributed ideas and discussions that shaped the proposal.

TODO: Add any acknowledgments

This work was funded by the National Institutes of Health (USA) under grant R01 GM070923.

References

Eriksson, H.-E. and Penker, M. (1998). *UML Toolkit.* John Wiley & Sons, New York.

Oestereich, B. (1999). *Developing Software with UML: Object-Oriented Analysis and Design in Practice.* Addison-Wesley.

W3C (2000). W3C's math home page. Available via the World Wide Web at `http://www.w3.org/Math/`.