BIOINFORMATICS

Vol. 19 no. 4 2003, pages 524–531
DOI: 10.1093/bioinformatics/btg015

The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models

M. Hucka[1, 2,*], A. Finney[1, 2], H. M. Sauro[1, 2], H. Bolouri[1, 2, 3], J. C. Doyle[1], H. Kitano[1, 2, 4, 16, 18], and the rest of the SBML Forum: A. P. Arkin[5], B. J. Bornstein[6], D. Bray[7], A. Cornish-Bowden[8], A. A. Cuellar[9], S. Dronov[10], E. D. Gilles[11], M. Ginkel[11], V. Gor[6], I. I. Goryanin[10], W. J. Hedley[9], T. C. Hodgman[10], J.-H. Hofmeyr[12], P. J. Hunter[9], N. S. Juty[10], J. L. Kasberger[5], A. Kremling[11], U. Kummer[13], N. Le Novère[7], L. M. Loew[14], D. Lucio[14], P. Mendes[15], E. Minch[19], E. D. Mjolsness[20], Y. Nakayama[16], M. R. Nelson[17], P. F. Nielsen[9], T. Sakurada[16], J. C. Schaff[14], B. E. Shapiro[6], T. S. Shimizu[7], H. D. Spence[10], J. Stelling[11], K. Takahashi[16], M. Tomita[16], J. Wagner[14] and J. Wang[17]

[1]Control and Dynamical Systems, MC 107-81, California Institute of Technology, Pasadena, CA 91125, USA, [2]ERATO Kitano Symbiotic Systems Project, Tokyo, Japan, [3]University of Hertfordshire, Hertfordshire, UK, [4]The Systems Biology Institute, Tokyo, Japan, [5]University of California, Berkeley, CA, USA, [6]NASA JPL, Pasadena, CA, USA, [7]University of Cambridge, Cambridge, UK, [8]CNRS-BIP, Marseille, France, [9]University of Auckland, Auckland, New Zealand, [10]GlaxoSmithKline, Stevenage, UK, [11]Max-Planck-Institute for Complex Technical Systems, Magdeburg, Germany, [12]University of Stellenbosch, Stellenbosch, South Africa, [13]EML, Heidelberg, Germany, [14]University of Connecticut Health Center, Farmington, CT, USA, [15]Virginia Bioinformatics Institute, Blacksburg, VA, USA, [16]Keio University, Tokyo, Japan, [17]Physiome Sciences Inc., Princeton, NJ, USA, [18]Sony Computer Science Laboratories, Inc., Tokyo, Japan, [19]LION bioscience AG, Heidelberg, Germany and [20]School of Information and Computer Science, University of California, Irvine, CA, USA

Received on May 17, 2002; revised on October 15, 2002; accepted on October 25, 2002

ABSTRACT

Motivation: Molecular biotechnology now makes it possible to build elaborate systems models, but the systems biology community needs information standards if models are to be shared, evaluated and developed cooperatively.

Results: We summarize the Systems Biology Markup Language (SBML) Level 1, a free, open, XML-based format for representing biochemical reaction networks. SBML is a software-independent language for describing models common to research in many areas of computational biology, including cell signaling pathways, metabolic pathways, gene regulation, and others.

Availability: The specification of SBML Level 1 is freely available from http://www.sbml.org/.

Contact: sysbio-team@caltech.edu.

*To whom correspondence should be addressed.

1 INTRODUCTION

Systems biology is characterized by synergistic integration of theory, computational modeling, and experiment (Kitano, 2002). Many contemporary research initiatives demonstrate the growing popularity of this kind of multidisciplinary work (e.g. Abbott, 1999). There now exists a variety of computational tools for the budding systems biologist (see below); however, the diversity of software has been accompanied by a variety of incompatibilities, and this has lead to numerous problems. For example:

- Users often need to work with complementary resources from multiple simulation/analysis tools in the course of a project. Currently this involves manually re-encoding the model in each tool, a time-consuming and error-prone process.
- When simulators are no longer supported, models developed in the old systems can become stranded and unusable. This has already happened on a number of occasions, with the resulting loss of usable models to the community. Continued innovation and development of new software tools will only aggravate this problem unless the issue is addressed.
- Models published in peer-reviewed journals are often accompanied by instructions for obtaining the model definitions. However, because each author may use a different modeling environment (and model representation language), such model definitions are often not straightforward to examine, test and reuse.

1.1 Approach

The current inability to exchange models between different simulation and analysis tools has its roots in the lack of a common format for describing models. To address this, we formed a *Software Platforms for Systems Biology* forum under the auspices of the ERATO Kitano Systems Biology Project (funded by the Japan Science and Technology Corporation and hosted in part at the California Institute of Technology). The forum initially included representatives from the teams developing the software packages *BioSpice* (Arkin, 2001), *Cellerator* (Shapiro and Mjolsness, 2001), *DBsolve* (Goryanin *et al.*, 1999), *E-CELL* (Tomita *et al.*, 2001), *Gepasi* (Mendes, 1997), *Jarnac* (Sauro, 2000), *StochSim* (Morton-Firth and Bray, 1998), and *Virtual Cell* (Schaff *et al.*, 2001), and later grew to include the developers of *ProMoT/DIVA* (Ginkel *et al.*, 2000) and the CellML language at the University of Auckland and Physiome Sciences (Hedley *et al.*, 2001).

The forum decided at the first meeting in April 2000 to develop a simple, XML-based language for representing and exchanging models between simulation/analysis tools: the *Systems Biology Markup Language* (SBML). We chose XML, the eXtensible Markup Language (Bray *et al.*, 1998), because of its portability and increasingly widespread acceptance as a standard data language for bioinformatics (Achard *et al.*, 2001). SBML is formally defined using UML, the Unified Modeling Language (Object Management Group, 2002), and this in turn is used to define a representation in XML. The base definition, *SBML Level 1*, is the result of analyzing common features in representation languages used by several ODE-, DAE- and stochastic-based simulators, and encompasses the minimal information required to support non-spatial biochemical models. Subsequent releases of SBML (termed *levels*) will add additional structures and facilities to Level 1 based on features requested and prioritized by the SBML community. By freezing sets of features in SBML definitions at incremental levels, we hope to provide software authors with stable standards and allow the simulation community to gain experience with the language definitions before introducing new elements.

1.2 Benefits to Biologists

Widespread use of SBML in software packages would benefit users as well as developers, by helping to address the problems of interoperability listed earlier in this introduction. With greater interaction between tools, and a common format for publications and databases, users would be better able to spend more time on actual research rather than on struggling with data format issues. (Note that biologists and other software users are *not* intended to write their models in SBML by hand—it is the software tools that read and write the format.)

2 OVERVIEW OF SBML LEVEL 1

A chemical reaction can be broken down into a number of conceptual elements: reactant species, product species, reactions, stoichiometries, rate laws, and parameters in the rate laws. To analyze or simulate a network of reactions, additional components must be made explicit, including compartments for the species, and units on the various quantities. A definition of a model in SBML simply consists of lists of one or more of these various components:

Compartment: A container of finite volume for well-stirred substances where reactions take place.

Species: A chemical substance or entity that takes part in a reaction. Some example species are ions such as calcium ions and molecules such as ATP.

Reaction: A statement describing some transformation, transport or binding process that can change one or more species. Reactions have associated rate laws describing the manner in which they take place.

Parameter: A quantity that has a symbolic name. SBML provides the ability to define parameters that are global to a model, as well as parameters that are local to a single reaction.

Unit definition: A name for a unit used in the expression of quantities in a model. This is a facility for both setting default units and for allowing combinations of units to be given abbreviated names.

Rule: A mathematical expression that is added to the model equations constructed from the set of reactions. Rules can be used to set parameter values, establish constraints between quantities, etc.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <sbml xmlns="http://www.sbml.org/sbml/level1"
3        level="1" version="2">
4    <model name="gene_network_model">
5      <listOfUnitDefinitions>
6          ...
7      </listOfUnitDefinitions>
8      <listOfCompartments>
9          ...
10     </listOfCompartments>
11     <listOfSpecies>
12         ...
13     </listOfSpecies>
14     <listOfParameters>
15         ...
16     </listOfParameters>
17     <listOfRules>
18         ...
19     </listOfRules>
20     <listOfReactions>
21         ...
22     </listOfReactions>
23   </model>
24 </sbml>
```

Fig. 1. The skeleton of a model definition expressed in SBML, showing all possible top-level elements.

A software package can read in a model expressed in SBML and translate it into its own internal format for model analysis. For instance, a package might provide the ability to simulate a model by constructing a set of differential equations representing the network and then performing numerical integration on the equations to explore the model's dynamic behavior.

Figure 1 shows the skeleton of an SBML model description. It exhibits the standard characteristics of an XML data stream (Bray *et al.*, 1998): it is plain text, each element consists of a matched pair of start/end tags enclosed by '<' and '>' characters, some elements can contain attributes of the form *attribute='value'*, and the first line contains a particular sequence of characters (beginning with '<?xml') declaring the rest of the data stream as conforming to the XML encoding standard.

The element `sbml`, beginning on line 2 of Figure 1, encapsulates an SBML model definition. The first attribute, `xmlns`, is required for tools that read XML to be able to verify the syntax of a given definition against the XML Schema for SBML. (This is an aspect of XML parsing that is beyond the scope of this article; interested readers may find more information in books such as that by Skonnard and Gudgin 2001.) The `level` attribute on element `sbml` identifies the SBML *level* in use; currently the only level defined is Level 1, but Level 2 is already under development. The attribute `version` is provided to enable updated versions of a given SBML level to be distinguished.

Inside `sbml`, there must be exactly one subelement: `model`, which itself can have a single optional attribute whose value specifies the name of the model (as shown on line 4). The `model` element can contain several different subelements; each acts as a container for a different kind of component in a model definition. The contents of these `listOf_____` containers are the topic of Section 4.

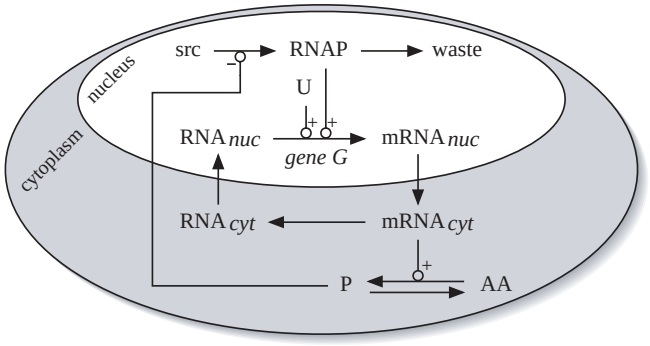

Fig. 2. Schematic diagram of the example model.

3 AN EXAMPLE MODEL

In the following sections, we describe the various components of SBML with the help of a concrete example. It illustrates one application of SBML, but it is by no means the only type of model that can be represented.

Our example is a two-compartment model of a hypothetical single-gene oscillatory circuit in a eukaryotic cell. The model is shown diagrammatically in Figure 2 and the reaction equations for the model are given in Table 1. In this highly simplified model, the nucleus of the cell is represented as one compartment and the surrounding cell cytoplasm as another compartment. Let us suppose that there is a gene G which encodes its own repressor and is transcriptionally activated at a constant rate, V_i, by a ubiquitous transcription factor U. Transcriptional activation involves several enzymatic reactions summarized here as the production of active *RNAP* (from source material, *src*) and its degradation (to *waste*). The transcribed *mRNA* is then transported out of the nucleus and into the cytoplasm, where it is translated into the product (*P*) of the gene G from constituent amino acids (*AA*) and where it is also subject to degradation. *P* travels from the cytoplasm back into the nucleus to repress further transcription of G, but is itself also subject to degradation. Eventually, the concentration of *P* becomes so low that G can be reactivated by U, and the cycle repeats itself.

Table 1. Reactions in the example model. $mRNA_{nuc}$: mRNA in nucleus. $mRNA_{cyt}$: mRNA in cytoplasm. RNA_{cyt}, RNA_{nuc}: RNA constituents. The terms beginning with the letters 'K' and 'V' are parameters given values in Section 4.4.

Reaction	Rate
$src \rightarrow RNAP$	$V_i/(1 + P/K_i)$
$RNAP \rightarrow waste$	$V_{kd} \cdot RNAP$
$RNA_{nuc} \rightarrow mRNA_{nuc}$	$\frac{V_{m1} \cdot RNAP \cdot RNA_{nuc}}{K_{m1} + RNA_{nuc}}$
$mRNA_{nuc} \rightarrow mRNA_{cyt}$	$k_1 \cdot mRNA_{nuc}$
$mRNA_{cyt} \rightarrow RNA_{cyt}$	$\frac{V_{m2} \cdot mRNA_{cyt}}{mRNA_{cyt} + K_{m2}}$
$RNA_{cyt} \rightarrow RNA_{nuc}$	$k_2 \cdot RNA_{cyt}$
$AA \rightarrow P$	$\frac{V_{m3} \cdot mRNA_{cyt} \cdot AA}{AA + K_{m3}}$
$P \rightarrow AA$	$(V_{m4} \cdot P)/(P + K_{m4})$

4 THE COMPONENTS OF SBML

Our goal in this section is to describe SBML in enough detail that readers can gain a general sense for its capabilities. This description summarizes SBML's major elements but omits many details; a detailed definition is presented in the SBML specification (Hucka *et al.*, 2003).

At the outset, we need to elaborate on two data type issues. The first concerns the definitions of basic data types such as `double`, `integer`, etc. Whenever these are used in SBML, they simply refer to the definitions of these data types in XML Schema (Biron and Malhotra, 2000; Thompson *et al.*, 2000). The second issue concerns the allowable syntax of names in name attributes. Names are used throughout SBML to allow different components of a model to have meaningful labels. When an SBML model definition is converted by a simulation/analysis software tool into the tool's native internal form, these names are typically turned into symbols in the software's representation of the model. However, some simulation and analysis tools place restrictions on the characters allowed in symbolic names. To support these packages, names in SBML Level 1 are restricted to character strings having the following syntax: a name is case-sensitive and must begin with either a letter or an underscore ('_') character, followed by any number of letters, digits or underscore characters in any combination. The minimum length for a name is one letter, or one underscore followed by one letter if the first character of the name is an underscore. A 'letter' can be either upper or lower case. Also, though XML permits the use of Unicode characters (Unicode Consortium, 1996), SBML limits the set of characters allowed in names to plain ASCII text characters for compatibility with existing simulation software.

4.1 Compartments

A *compartment* in SBML represents a bounded volume in which species are located. Compartments do not necessarily have to correspond to actual structures inside or outside of a cell, although models are often designed that way. The following fragment of SBML defines the compartments for our example model:

```
<listOfCompartments>
  <compartment name="Cyt" volume="1.5" />
  <compartment name="Nuc" outside="Cyt" />
</listOfCompartments>
```

There is one required attribute for a `compartment` element, `name`, to give it a unique name by which other parts of an SBML model definition can refer to it. A `compartment` can also have an optional `volume` attribute giving the total volume of the compartment. This enables concentrations of species to be calculated in the absence of spatial geometry information. The `volume` attribute defaults to a value of '`1`' (one). The units of volume may be explicitly set using the optional attribute `units`. The value of this attribute must be one of the following: a predefined unit name from Table 2, the term '`volume`' (which, if used, signifies that the default units of volume should be used—see Section 4.5), or the name of a unit defined by a unit definition in the enclosing `model`. If absent, as in the example above, the units default to the value set by the built-in '`volume`'.

The optional attribute `outside` can be used to express containment relationships between compartments. If present, the value of `outside` for a given compartment must be the name of another compartment enclosing it, or in other words, the compartment that is 'outside' of it. This enables the representation of simple topological relationships between compartments, for those simulation systems that can make use of the information (e.g. for drawing simple diagrams of compartments). Although containment relationships are partly taken into account by the compartmental localization of reactants and products, it is not always possible to determine purely from the reaction equations whether one compartment is meant to be located within another. In the absence of a value for `outside`, compartment definitions in SBML Level 1 do not have any implied spatial relationships between each other. (We hope to introduce support for additional spatial characteristics in a future level of SBML.)

As with the other top-level components, compartments are optional in an SBML model definition. If no compartment is defined, the model is assumed to be located within a single compartment of unit volume.

4.2 Species

The species element in SBML is used to represent entities such as ions and molecules that participate in reactions. The following is the list of species for our example:

```
<listOfSpecies>
  <species name="mRNA_nuc" compartment="Nuc"
           initialAmount="0.0032834" />
  <species name="RNA_nuc" compartment="Nuc"
           initialAmount="96.117" />
  <species name="RNAP" compartment="Nuc"
           initialAmount="0.66349" />
  <species name="mRNA_cyt" compartment="Cyt"
           initialAmount="3.8742"/>
  <species name="P" compartment="Cyt"
           initialAmount="22.035" />
  <species name="RNA_cyt" compartment="Cyt"
           initialAmount="0.0054068" />
  <species name="AA" compartment="Cyt"
           initialAmount="90.465" />
  <species name="src" compartment="Nuc"
           initialAmount="1"
           boundaryCondition="true" />
  <species name="waste" compartment="Nuc"
           initialAmount="1"
           boundaryCondition="true" />
</listOfSpecies>
```

The species element has two required attributes: name and initialAmount. The attribute name is required to give each species a unique name in a model. The attribute initialAmount, of type double, is used to define the initial quantity (as a total molar amount, not concentration) of the species in the compartment where it is located. The units of this quantity may be set explicitly using the optional attribute units. The value of units must be one of the following: a predefined unit name from Table 2, the term 'substance' (which, if present, signifies that the default units of quantity should be used—see Section 4.5), or a new unit name defined by a unit definition in the enclosing model. If absent, the units default to the value set by the built-in 'substance'.

The attribute compartment is a string that names the compartment within which the species is located. The attribute can be omitted only if the model does not define any compartments (and thus assumes the default; see Section 4.1); otherwise, each species must have a value for compartment.

The optional attribute boundaryCondition takes on a boolean value to indicate whether the amount of the species is fixed or variable over the course of a simulation. The value of boundaryCondition defaults to a value of 'false', indicating that by default, the amount is not fixed. If the amount of a species is defined as being fixed, it implies that some external mechanism maintains a constant quantity in the compartment throughout the course of a reaction. (The term *boundary condition* alludes to the role of this constraint in a simulation.)

A final optional attribute of species is charge, an integer indicating a charge value (in terms of electrons, not the SI unit Coulombs). This may be useful when the species is a charged ion such as calcium (Ca^{2+}).

4.3 Reactions

A *reaction* represents some transformation, transport or binding process, typically a chemical reaction, that can change one or more chemical species. In SBML, reactions are defined using lists of reactant species and products, their stoichiometric coefficients, and kinetic rate laws. Space limitations permit us to give only one SBML reaction definition as an example:

```
<listOfReactions>
 <reaction name="R1" reversible="false">
   <listOfReactants>
     <species Reference species="src" />
   </listOfReactants>
   <listOfProducts>
     <species Reference species="RNAP"/>
   </listOfProducts>
   <kineticLaw formula="Vi/(1+P/Ki)" />
 </reaction>
 ...

</listOfReactions>
```

The required name attribute gives the reaction a unique name to identify it in the model. The optional attribute reversible takes a boolean value indicating whether the reaction is reversible. If unspecified, the default value is 'true'. An explicit flag is necessary because the kinetic law expression for a reaction is optional. Information about reversibility is useful in certain kinds of analyses such as elementary mode analysis (Schuster *et al.*, 2000).

The optional attribute fast is another boolean attribute in the reaction element; a value of 'true' signifies that the given reaction is a 'fast' one. This may be relevant when computing equilibrium concentrations of rapidly equilibrating reactions. Simulation/analysis packages may choose to use this information to reduce the number of ODEs required and thereby optimize such computations. The default value of fast is 'false'.

The reactants and products of a reaction are identified by references to species using speciesRef elements inside listOfReactants and listOfProducts containers. A speciesRef has one required attribute, species, whose value must be the name of a species defined in the model's listOfSpecies. Stoichiometric numbers for the products and reactants can be specified using two optional attributes on the speciesRef element: stoichiometry and denominator. Both attributes take

positive integers as values, and both have default values of '1' (one). The absolute value of the stoichiometric number is the value of `stoichiometry` divided by `denominator`, and the sign is implicit from the role of the species (i.e. positive for reactants and negative for products). The use of separate numerator and denominator terms allows a simulator to employ rational arithmetic if it is capable of it, potentially reducing round-off errors and other problems during computations. In our example model above, we only needed to use the default values.

Finally, the optional `kineticLaw` element is used to provide a mathematical formula describing the rate at which the reactants combine to form the products. (In general there is no useful default value that can be substituted in place of a missing kinetic law, but the element is optional because certain kinds of network analysis are still possible in the absence of information on reaction kinetics.) The `kineticLaw` element has one required attribute, `formula`, of type `string`, that expresses the rate of the reaction in *substance*/*time* units. The allowable syntax of formula strings is described in the SBML Level 1 specification; it consists of basic operators such as multiplication, addition, exponentiation, etc., as well as a number of predefined functions for common kinetic rate laws.

A `kineticLaw` element can optionally have attributes `substanceUnits` and `timeUnits` to specify the units of substance and time. If these attributes are not used in a given reaction, the units are taken from the defaults defined by the built-in terms '`substance`' and '`time`' of Table 3 in Section 4.5. Although not used in our two-compartment example model, a `kineticLaw` element can also contain zero or more optional `parameter` elements that define new terms used only in the `formula` string.

Readers may wonder why formulas in SBML are not expressed using MathML (W3C, 2000). Although using MathML would be more in the spirit of XML, it would introduce new complexity for software tools. Most contemporary simulation software tools for systems biology represent mathematical formulas simply using text strings. To keep SBML Level 1 simple and maximally compatible with known software, we chose to represent formulas as strings as well. This does not preclude a later level of SBML from introducing the ability to use MathML.

4.4 Parameters

The `parameter` element in SBML is used to associate a name with a floating-point value, so that the name can be used in formulas in place of the value. Here are the parameter definitions for our example:

```
<listOfParameters>
  <parameter name="Vi"  value="10" />
  <parameter name="Ki"  value="0.6"/>
  <parameter name="Vkd" value="1" />
  <parameter name="Vm1" value="50" />
  <parameter name="Km1" value="1" />
  <parameter name="k1"  value="10000" />
  <parameter name="Vm2" value="50" />
  <parameter name="Km2" value="1" />
  <parameter name="k2"  value="10000" />
  <parameter name="Vm3" value="50" />
  <parameter name="Km3" value="80" />
  <parameter name="Vm4" value="50" />
  <parameter name="Km4" value="1" />
</listOfParameters>
```

The `parameter` element has one required attribute, `name`, representing the parameter's name in the model. The optional attribute `value` is of type `double` and determines the numerical value assigned to the parameter. The units on the `value` may be specified by the optional attribute `units`. The string used for `units` must be chosen from one of the following: a predefined unit name from Table 2; one of the three terms '`substance`', '`time`', or '`volume`' (see Section 4.5); or the name of a new unit defined in the list of unit definitions in the enclosing `model`.

Parameters can be defined in two places in SBML: in lists of parameters defined at the top level in a `model`-type structure (in the `listOfParameters` described in Section 2), and within individual reaction definitions (as described in Section 4.3). Parameters defined at the top level are *global* to the whole model; parameters that are defined within a reaction are local to the particular reaction and (within that reaction) *override* any global parameters having the same names.

4.5 Unit Definitions

Although we did not need to define any special units in our example model, SBML does provide a way to define new units and redefine default units.

A unit definition consists of a `name` attribute and an optional `listOfUnits` subelement that in turn contains one or more `unit` elements. For example, the following definition illustrates how an abbreviation named '`mmls`' can be defined for the units mmol l^{-1} s^{-1}:

```
<listOfUnitDefinitions>
  <unitDefinition name="mmls">
    <listOfUnits>
      <unit kind="mole"   scale="-3"/>
      <unit kind="liter"  exponent="-1"/>
      <unit kind="second" exponent="-1"/>
    </listOfUnits>
  </unitDefinition>
  ...
</listOfUnitDefinitions>
```

Table 2. The possible values of `kind` in a `unit` element. All are names of base or derived SI units, except for '`dimensionless`' and '`item`', which are SBML additions. '`Dimensionless`' is needed for cases where a quantity does not have units, and '`item`' is needed to express such things as 'N items' (e.g. '100 molecules'). Although '`Celsius`' is capitalized, for simplicity, SBML requires that these names be treated in a case-insensitive manner. Also, note that the gram and liter/litre are not strictly part of International System of Units (BIPM, 2000); however, they are so commonly used in SBML's areas of application that they are included as predefined unit names.

ampere	henry	lumen	second
becquerel	hertz	lux	siemens
candela	item	meter	sievert
Celsius	joule	metre	steradian
coulomb	katal	mole	tesla
dimensionless	kelvin	newton	volt
farad	kilogram	ohm	watt
gram	liter	pascal	weber
gray	litre	radian	

As this illustrates, SBML uses a compositional approach to defining units. The definition of mmol l^{-1} s^{-1} is constructed by combining a `unit` element representing millimoles with a `unit` element representing liter^{-1} and another `unit` element representing second^{-1}.

The `unit` element has one required attribute, `kind`, whose value must be a name taken from the list of units in Table 2. The optional `exponent` attribute has a default value of '`1`' (one). A unit such as liter^{-1} is obtained by using attributes `kind="liter"` and `exponent="-1"`. Finally, a `unit` element also accepts an optional `scale` field; its value must be an integer used to set the scale of the unit. For example, a unit that has a `kind` value of '`gram`' and a `scale` value of '`-3`' signifies $10^{-3} * \text{gram}$, or milligrams. The default value of `scale` is zero.

There are three special unit names in SBML, listed in Table 3, corresponding to the three types of quantities that play roles in biochemical reactions: amount of substance, volume and time. SBML defines default units for these quantities, all with a default `scale` value of `0`. The various components of a model, such as parameters, can use only the predefined units from Table 2, new units defined in unit definitions, or the three predefined names '`substance`', '`time`', and '`volume`' from Table 3. The latter usage signifies that the units to be used should be the designated defaults. A model may change the default scales by reassigning the keywords '`substance`', '`time`', and '`volume`' in a unit definition.

4.6 Rules

Rules in SBML provide a way to create constraints on variables and parameters for cases in which the constraints cannot be expressed using the reaction components described in Section 4.3. There are two orthogonal dimensions by which rules can be described. First, there are three different possible functional forms, corresponding to the following three general cases (where x is a variable, f is some arbitrary function, and W is a vector of parameters and variables that may include x):

Table 3. SBML's built-in quantities.

Name	Allowable Units	Default Units
`substance`	moles *or* no. of molecules	moles
`volume`	liters	liters
`time`	seconds	seconds

1. left-hand side is zero: $0 = f(W)$
2. left-hand side is a scalar: $x = f(W)$
3. left-hand side is a rate-of-change: $\mathrm{d}x/\mathrm{d}t = f(W)$

The second dimension concerns the role of variable x in the equations above: x can be the name of a compartment (to set its volume), the name of a species (to set its concentration), or the name of a parameter (to set its value).

The approach taken to covering these cases in SBML is to define separate kinds of elements for each of the cases, and to allow these within a single `listOfRules` container within a `model` definition (see Table 1). Each contains a `name` attribute that specifies the quantity being referenced, and a `formula` attribute that holds the right-hand side expression of the rule. For the actual details, we refer readers to the SBML Level 1 specification.

5 STATUS AND FUTURE PLANS

As mentioned above, SBML Level 1 is intended to provide only a basic representation of biochemical reaction networks. Space constraints prevent us from giving a detailed description of SBML here; the full definition is available in a separate document (Hucka *et al.*, 2003). A number of simulation and analysis packages already support SBML Level 1 or are in the process of being extended to support it. At the time of this writing, the tools include: *Cellerator* (Shapiro and Mjolsness, 2001), *DBsolve* (Goryanin *et al.*, 1999), *E-CELL* (Tomita *et al.*, 2001), *Gepasi* (Mendes, 1997), *Jarnac* (Sauro, 2000), *NetBuilder* (Brown *et al.*, 2002), *ProMoT/DIVA* (Ginkel *et al.*, 2000), *StochSim* (Morton-Firth and Bray, 1998), and *Virtual Cell* (Schaff *et al.*, 2001).

Future levels of SBML will add more features requested by the modeling community. The process for feature selection involves a request for proposals from the *Software Platforms for Systems Biology* forum, followed by discussions and votes during subsequent meetings, and finally the drafting of a specification by selected members. Some of the features under discussion for SBML Level 2 are the

introduction of MathML and metadata support. The latter will add a systematic mechanism for recording such information as author and publication references; it will also provide a way to annotate a model with information such as cross-references to biological data sources.

Finally, the project is moving from a primarily Caltech/ERATO-led effort toward a community-led and -maintained model for the future of SBML. We invite all interested parties to join us.

ACKNOWLEDGEMENTS

This work has been supported by: the Japan Science and Technology Corporation's ERATO Kitano Symbiotic Systems Project; an International Joint Research Grant from the NEDO/Japanese Ministry of Economy, Trade and Industry; and the Rice Genome and Simulation Project from the Japanese Ministry of Agriculture.

REFERENCES

Abbott,A. (1999) Alliance of US labs plans to build map of cell signaling pathways. *Nature*, **402**, 219–220.

Achard,F., Vaysseix,G. and Barillot,E. (2001) XML, bioinformatics and data integration. *Bioinformatics*, **17**, 115–125.

Arkin,A.P. (2001) *Simulac & Deduce*. Available via the World Wide Web at http://gobi.lbl.gov/∼aparkin.

Biron,P.V. and Malhotra,A. (2000) XML Schema part 2: Datatypes, Available via the World Wide Web at http://www.w3.org/TR/xmlschema-2/.

Bray,T., Paoli,J. and Sperberg-McQueen,C.M. (1998) Extensible markup language (XML) 1.0, Available via the World Wide Web at http://www.w3.org/TR/1998/REC-xml-19980210.

Brown,C.T., Rust,A.G., Clarke,P.J.C., Pan,Z., Schilstra,M.J., De Buysshcher,T., Griffin,G., Wold,B.J., Cameron,R.A., Davidson,E.H. and Bolouri,H. (2002) New computational approaches for analysis of cis-regulatory networks. *Developmental Biology*, **246**, 86–102.

Bureau International des Poids et Mesures (2000) *The International System of Units (SI) supplement 2000*. Available via the World Wide Web at http://www.bipm.fr/pdf/si-supplement2000.pdf.

Ginkel,M., Kremling,A., Tränkle,F., Gilles,E.D. and Zeitz,M. (2000) Application of the process modeling tool ProMoT to the modeling of metabolic networks. In Troch,I. and Breitenecker,F. (eds), *Proc. of the 3rd MATHMOD*.

Goryanin,I., Hodgman,T.C. and Selkov,E. (1999) Mathematical simulation and analysis of cellular metabolism and regulation. *Bioinformatics*, **15**, 749–758.

Hedley,W.J., Nelson,M.R., Bullivant,D.P. and Nielson,P.F. (2001) A short introduction to CellML. *Phil. Trans. Roy. Soc. London A*, **359**, 1073–1089.

Hucka,M., Finney,A., Sauro,H.M. and Bolouri,H. (2003) Systems Biology Markup Language (SBML) Level 1: Structures and facilities for basic model definitions, Available via the World Wide Web at http://www.sbml.org/.

Kitano,H. (2002) Systems biology: a brief overview. *Science*, **295**, 1662–1664.

Mendes,P. (1997) Biochemistry by numbers: simulation of biochemical pathways with Gepasi 3. *Trends Biochem. Sci.*, **22**, 361–363.

Morton-Firth,C.J. and Bray,D. (1998) Predicting temporal fluctuations in an intracellular signalling pathway. *J. Theor. Biol.*, **192**, 117–128.

Object Management Group (2002) *UML Specification documents available via the World Wide Web at http://www.omg.org/uml/*.

Sauro,H.M. (2000) Jarnac: a system for interactive metabolic analysis. In Hofmeyr,J.-H., Rohwer,J. and Snoep,J. (eds), *Animating the Cellular Map*. Stellenbosch Univ. Press.

Schaff,J., Slepchenko,B., Morgan,F., Wagner,J., Resasco,D., Shin,D., Choi,Y.S., Loew,L., Carson,J., Cowan,A. *et al.* (2001) *Virtual Cell*. Available via the World Wide Web at http://www.nrcam.uchc.edu.

Schuster,S., Fell,D.A. and Dandekar,T. (2000) A general definition of metabolic pathways useful for systematic organization and analysis of complex metabolic networks. *Nat. Biotechnol.*, **18**, 326–332.

Shapiro,B.E. and Mjolsness,E.D. (2001) Developmental simulations with Cellerator. In Yi,T.-M., Hucka,M., Morohashi,M. and Kitano,H. (eds), *Proceedings of the Second International Conference on Systems Biology (ICSB2001)*. Omnipress.

Skonnard,A. and Gudgin,M. (2001) Essential XML Quick Reference, Addison-Wesley.

Thompson,H.S., Beech,D., Maloney,M. and Mendelsohn,N. (2000) *XML Schema part 1: Structures*. Available via the World Wide Web at http://www.w3.org/TR/xmlschema-1/.

Tomita,M., Nakayama,Y., Naito,Y., Shimizu,T., Hashimoto,K., Takahashi,K., Matsuzaki,Y., Yugi,K., Miyoshi,F., Saito,Y. *et al.* (2001) *E-Cell*. Available via the World Wide Web at http://www.e-cell.org/.

Unicode Consortium, (1996) *The Unicode Standard, Version 2.0*. Addison-Wesley Developers Press.

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