models and tacit rule-based models have long been used in product development and safety testing in various engineering disciplines, such as aerospace engineering and electronic circuit design. It can be envisaged that predictive approaches is expected to increase significantly in the coming years in drug discovery too. The extent of *omics*-scale data and the advances in systems technologies to enable comprehension of such large complex data in the form of meaningful biological models are promising to help in this process. The power of systems biology methods is such that it may become possible in the not-too distant future, that a disease could get diagnosed in a clinical setting and characterized at the systems level with precise genotype and phenotype definitions, leading all the way up to predictive quantitative titrations of the available remedies and finally personalized prescriptions.

References

Boran D, Iyengar R (2010) Systems approaches to polypharmacology and drug discovery. Curr Opin Drug Discov Dev 13(3):297–309

Butcher EC, Berg EL et al (2004) Systems biology in drug discovery. Nat Biotechnol 22(10):1253–1259

Forst CV (2006) Host-pathogen systems biology. Drug Discov Today 11:220–227

Glaab E, Baudot A et al (2012) EnrichNet: network-based gene set enrichment analysis. Bioinformatics 28(18): i451–i457

Hood L, Perlmutter RM (2004) The impact of systems approaches on biological problems in drug discovery. Nat Biotechnol 22(10):1215–1217

Ideker T, Krogan NJ (2012) Differential network biology. Mol Syst Biol 8:565

Kitano H (2002) Systems biology: a brief overview. Science 295(5560):1662–1664

Lee DS, Burd H et al (2009) Comparative genome-scale metabolic reconstruction and flux balance analysis of multiple *Staphylococcus aureus* genomes identify novel antimicrobial drug targets. J Bacteriol 191(12):4015–4024

Orth JD, Thiele I et al (2010) What is flux balance analysis? Nat Biotechnol 28(3):245–248

Raman K, Chandra N (2008) *Mycobacterium tuberculosis* interactome analysis unravels potential pathways to drug resistance. BMC Microbiology 2008(2):109

Raman K, Yeturu K et al (2008) TargetTB: a target identification pipeline for *Mycobacterium tuberculosis* through an interactome, reactome and genome-scale structural analysis. BMC Syst Biol 2:109

Rappuoli R, Aderem A (2011) A 2020 vision for vaccines against HIV, tuberculosis and malaria. Nature 473(7348):463–469

Systems Biology Graphical Notation

▶ SBGN

Systems Biology Markup Language (SBML)

Michael Hucka
Computing and Mathematical Sciences,
California Institute of Technology, Pasadena,
CA, USA

Synonyms

SBML

Definition

SBML (the Systems Biology Markup Language) is a representation format, based upon XML, used for communicating and storing computational models of biological processes (Hucka et al. 2003). SBML can represent many different classes of biological phenomena, including metabolic networks, cell-signaling pathways, regulatory networks, disease models, and many others. It does not attempt to be a universal language for quantitative models; rather, SBML's purpose is to serve as a franca lingua for exchanging the essential aspects of a computational model between software systems and databases. It is intended to be used by software and not written by humans directly, although its text-based nature makes it reasonably comprehensible for situations such as debugging and software development, when direct access is useful.

Characteristics

There are many aspects to SBML and its correct use. The following paragraphs can only summarize some notable points; elaborations and further details are available in the SBML specification documents (SBML Team 2011).

General Principles of SBML

SBML has three main purposes:

1. Enable modelers to use multiple software tools without having to rewrite models to conform to every tool's idiosyncratic file format
2. Enable models to be shared and published in a form that other researchers can use even when working with different software environments
3. Ensure the survival of models beyond the lifetime of the software used to create them

The most common and basic form of an SBML model consists of entities (called *species* in SBML) acted upon by processes (called *reactions* in SBML). An important principle is that models are decomposed into explicitly labeled constituent elements, the set of which resembles a verbose rendition of an explicit set of equations. This set of equations can represent both chemical reaction equations (if the model uses reactions) and equations derived from other concepts (again, if the model uses them). The SBML representation deliberately does not express a model directly as (for instance) a set of differential equations or other specific interpretation of the model. This explicit, framework-agnostic decomposition makes it easier for a software tool to interpret the model and translate the SBML form into whatever internal form the tool actually uses.

A software system can read an SBML model description and translate it into its own internal format for model analysis. For example, a software system might provide the ability to simulate the model by constructing differential equations and then perform numerical time integration on the equations to explore the model's dynamic behavior. Or, alternatively, a package might construct a discrete stochastic representation of the model and use a Monte Carlo simulation method such as the Gillespie algorithm.

Another important feature of SBML is that every entity can have machine-readable annotations attached to it. These annotations can be used to express relationships between the entities in a given model and entities in external resources such as online databases. A good example of the value of this is in BioModels Database (Li et al. 2010), where every model is annotated and linked to relevant data resources such as publications, databases of compounds and pathways, controlled vocabularies, and more. With annotations, a model becomes more than simply a rendition of a mathematical construct – it becomes a semantically enriched framework for communicating knowledge.

SBML is sometimes incorrectly assumed to be limited in scope only to biochemical network models because the original publications and early software focused on this domain. In reality, although the central features of SBML are indeed oriented toward representing "reaction-like" processes that act on some entities to generate new or different amounts of other entities, this same formalism serves analogously for many other types of processes; moreover, SBML also supports the direct expression of mathematical formulas and discontinuous events apart from reaction processes, allowing SBML to represent much more than only biochemical reactions.

Structure of SBML

SBML allows models of arbitrary complexity to be represented. Each type of component in a model is described using a specific type of SBML data structure that organizes the relevant information. The data structures determine how the resulting model is encoded in XML. The main data structures in SBML Level 3 Version 1 are the following:

- *Function definition*: A named mathematical function that may be used throughout the rest of a model.
- *Unit definition*: A named definition of a new unit of measurement. Named units can be used in the expression of quantities in a model.
- *Compartment*: A well-stirred container of finite size where species may be located. Compartments may or may not represent actual physical structures.
- *Species*: A pool of entities of the same kind located in a compartment and participating in reactions (processes). In biochemical network models, common examples of species include ions, proteins, and other molecules; however, in practice, an SBML species can be any kind of entity characterizable in terms of an amount.
- *Parameter*: A quantity with a symbolic name. In SBML, the term "parameter" is used in a generic sense to refer to named quantities regardless of whether they are constants or variables in a model. SBML provides the ability to define parameters that are global to a model as well as parameters that are local to a single reaction.
- *Initial Assignment*: A mathematical expression used to determine the initial conditions of a model. This type of object can only be used to define how the value of a variable can be calculated from other values and variables at the start of simulated time.

- *Rule*: A mathematical expression added to the set of equations constructed based on the reactions defined in a model. Rules can be used to define how a variable's value can be calculated from other variables, or used to define the rate of change of a variable. The set of rules in a model can be used with the reaction rate equations to determine the behavior of the model with respect to time.
- *Constraint*: A means of detecting out-of-bounds conditions during a dynamical simulation and optionally issuing diagnostic messages. Constraints are defined by an arbitrary mathematical expression computing a true/false value from model variables, parameters, and constants. An SBML constraint applies at all instants of simulated time; however, the set of constraints in the model should not be used to determine the behavior of the model with respect to time.
- *Reaction*: A statement describing some transformation, transport, or binding process that can change the amount of one or more species. For example, a reaction may describe how certain entities (reactants) are transformed into certain other entities (products). Reactions have associated kinetic rate expressions describing the speed at which reactions occur.
- *Event*: A statement describing an instantaneous, discontinuous change in one or more variables of any type (species, compartment, parameter, etc.) when a triggering condition is satisfied.

Mathematical formulas are represented using a subset of MathML. The SBML specification defines the MathML operators allowed in formulas in SBML, as well as how the identifiers of the various constructs like species and compartment objects are linked with MathML formulas.

Annotations

The constructs in SBML have only limited mathematical semantics. They have no predefined biological or biochemical semantics, and though a human could make inferences when inspecting a given model, software programs are not as competent in that regard. For software, the intended meaning of each model component needs to be made explicit and in a machine-readable form. SBML defines two separate systematic ways of adding annotations to any component of a model.

The first type of annotation takes the form of references to terms taken from the Systems Biology Ontology (SBO; Le Novère 2006). This set of controlled vocabularies provides terms for identifying such things as common reaction rate expressions, common participant types and roles in reactions, common parameter types and their roles in rate expressions, common modeling frameworks (e.g., "continuous," "discrete," etc.), and common types of biochemical species and reactions. By adding references to SBO terms to components of an SBML model, a software tool can provide additional details using independent, shared vocabularies that can enable other software tools to recognize precisely what the component is meant to represent. For example, if a given reaction in a model has an SBO attribute referencing term SBO:0000049 (which corresponds to "first-order irreversible mass-action kinetics, continuous framework" in SBO), then regardless of the identifier and name given to the reaction in the model, a software tool can look up the SBO term to inform users that the reaction is a first-order irreversible mass-action reaction.

The second type of annotation in SBML is more flexible and wider in scope. Its syntax consists of a structured subelement (the "annotation" subelement) that can be attached to any component in a model and can be used to define a relationship between the SBML component being annotated and the annotation content. The content can be either history information (date created, date modified, author, contact info, etc.) or references to external resources. The external resource can be anything – an entry in an online database, a publication, a part of another model, a term in an ontology, etc. The format of this kind of extended annotation in SBML follows the MIRIAM guidelines. Each annotation is a triplet consisting of (1) a data type, (2) an identifier, and (3) an optional qualifier. The data type is a unique controlled description of the type of the data in annotation and should be recorded as a Uniform Resource Name (URN); the identifier refers/points to a specific datum in whatever source is identified by the data type; and the qualifier serves to refine the nature of the relationship between the model component being annotated and the referred-to datum. Examples of common qualifiers include "is version of," "has part," etc.

SBML Evolution and Growth

The development of SBML is stratified in order to organize architectural changes and versioning. Major editions of SBML are termed *Levels* and represent substantial changes to the composition and structure

of the language. Models defined in lower Levels of SBML can always be represented in higher Levels, though some translation may be necessary. The converse (from higher Level to lower Level) is sometimes also possible, though not guaranteed. The Levels remain distinct; a valid SBML Level 1 document is not a valid SBML Level 2 document. Minor revisions of SBML are termed *Versions* and constitute changes within a level to correct, adjust, and refine language features. Finally, specification documents inevitably require minor editorial changes as its users discover errors and ambiguities. Such problems are corrected in new *Releases* of a given SBML specification.

The latest generation of SBML, which is Level 3, is modular in the sense of having a defined core set of features and optional packages adding features on top of the core. This modular approach means that models can declare which feature-sets they use, and likewise, software tools can declare which packages they support. It also means that the development of SBML Level 3 can proceed in a modular fashion. The development process for Level 3 is designed around this concept.

SBML Level 3 package development is today an ongoing activity, with packages being created to extend SBML in many areas that its core functionality does not directly support. Examples include models whose species have structure and/or state variables, models with spatially nonhomogeneous compartments and spatially dependent processes, and models in which species and processes refer to qualitative entities and processes rather than quantitative ones.

SBML Development Process

SBML uses a community-oriented development approach. For example, technical decisions are made by a group of volunteer editors, with major decisions made as much as possible using electronic voting by the whole SBML community. Much of the development process is defined in a written document (made available on the SBML.org website) that provides guidelines for various aspects of the overall management of SBML development and the SBML community. The following are some of the features of the process:

- The SBML community is organized into the SBML Forum, the SBML Editors, and the SBML Team. The SBML Forum consists of all members of the community who subscribe to the sbml-discuss mailing list, with the list membership acting as a kind of basic voter registration mechanism. The SBML Editors are volunteers who are sufficiently interested in SBML and its continued development that they are willing to spend time in the development, writing, and correction of SBML specification documents. There are five SBML Editors at any given time; they are elected by a majority vote from among the SBML Forum, and they serve 3-year terms, with reelection being possible but consecutive terms being disallowed. The SBML Team are members who are employed to work on SBML-related activities. Their tasks include maintaining the resources that support the SBML community and SBML development in general; developing critical software such as libraries and online facilities; and organizing events and other similar activities.
- Discussions are held publicly as much as possible, usually on the sbml-discuss mailing list and in biannual face-to-face meetings. The public discussions and archives improve transparency, provide a public record of arguments and reasoning, and stimulate the broader community.
- Consensus is sought as much as possible. In situations where a decision appears to have no obvious right or wrong answer on technical grounds alone, the SBML Editors may initiate a public vote on the matter. These votes are typically conducted using an electronic voting system, with the topic and call-for-votes announced on the sbml-discuss@caltech.edu mailing list.

Strengths and Weaknesses of SBML

A few notable strengths of the SBML approach are (1) the use of explicit constructs for representing different facets of a model, (2) the relatively limited number of constructs, and (3) the community-driven development approach. A few notable weaknesses of SBML are (1) the seeming focus on biochemical reaction-based processes and (2) the introduction of syntactic differences between versions.

The use of explicit constructs means that the different aspects of a model in SBML are labeled and characterized explicitly. This facilitates consistent software interpretation of models, because the important features of a model are made explicit – an interpreter does not have to guess at the meaning behind, say, a set of equations (which ones stand for reactions? which ones are other relationships?), and various features such as function definitions are provided in a consistent (if limited) syntax. Moreover, this makes

it more straightforward to take the same model and express it in any of a variety of different mathematical frameworks.

A second strength, the relatively limited number of constructs in SBML, means that it is less work for a software developer to implement tools for working with the format. SBML could have been designed with, for example, a deeper hierarchy of data types, but this was rejected purposefully to limit the complexity of implementations. (However, in fairness, this is not to say that SBML is very simple; there are still quite a few constructs and nuances in their meaning).

Finally, a third strength of SBML is, as mentioned above, the support and involvement of the community in its development and adoption. Specifications and technical decisions are made collectively by a small set of SBML Editors in collaboration with the whole SBML community, and electronic voting is used to reach community-wide consensus on important decisions.

Among the notable weaknesses, the first is SBML's seeming focus on reaction-based processes. The objects and terms in SBML (such as its "Species" and "Reaction" objects) are admittedly couched in biochemical reaction terms, reflecting SBML's origins and history. Many potential users assume SBML is limited to models of this type, but in reality, the same concepts can easily be used in other domains. In hindsight, it is now clear that more neutral terms could have been chosen.

A second weakness is the number of changes between versions within an SBML Level. The changes reflect hard-won experience by the SBML community and especially software developers, so from one perspective they are an understandable consequence of evolution and improvement. However, the changes make it admittedly more difficult for software developers to support all versions of SBML.

Relationships to Other Standardization Efforts

SBML has proven useful for software tools to exchange computational models. Still, by itself it does not provide a complete framework for reproducible modeling. Several related efforts now exist to standardize additional aspects of model exchange.

- *The Systems Biology Ontology* (*SBO*). As mentioned above, SBO provides a set of controlled vocabularies that can be used to annotate a model to make its mathematical semantics more precise (Le Novère 2006).
- *The Minimum Information Requested in the Annotation of biochemical Models* (*MIRIAM*). SBML defines a syntax for how to encode annotations in a MIRIAM-compliant way, but MIRIAM is itself a separate standardization effort applicable to any encoding format, not just SBML. It defines a basic and straightforward annotation scheme. It is also backed by a software resource, the *MIRIAM Services*, that provides a variety of tools for address resolution of references to resources on the Internet.
- *The Simulation Experiment Description Markup Language* (*SED-ML*). This XML-based format for encoding simulation experiments provides a tool-independent way of defining the model(s) to be used, the experimental task(s) to be run, and the result(s) to be produced (Waltemath et al. 2011).

Besides these, there also exist standardization activities for closely related topics. Many have separate sections in this encyclopedia: the *Systems Biology Graphical Notation* (SBGN; Le Novère et al. 2009), the *Biological Pathway Exchange* (BioPAX) *language* (Demir et al. 2010), *NeuroML*, and *CellML*.

Resources for SBML

For software developers who seek to implement support for SBML in their software, as well as modelers working with SBML files, there are many relevant resources available. The following paragraphs summarize some that may be especially useful.

Software

In addition to the hundreds of SBML-aware software systems for biological modeling and other purposes, two classes of important software resources for SBML are software libraries and validation tools.

Developers interested in supporting SBML in their software are encouraged to consider the use of the free, open-source software libraries libSBML (Bornstein et al. 2008) and JSBML (Dräger et al. 2011). LibSBML is written in ISO C and C++ and provides language interfaces for C, C#, C++, Java, MATLAB, Octave, Python, Perl, and Ruby. LibSBML is supported on Linux, Windows, and Mac OS X, and is distributed in both source-code form and as precompiled, ready-to-install libraries. Among its many features are support for all releases of SBML,

unit checking and dimensional analysis, full validation of SBML, and APIs for working with mathematical formulas, annotations, and handling of compressed files. The JSBML library is a pure-Java implementation similar in its API to libSBML; at the time of this writing, it is relatively young and so does not offer as many features as libSBML, but may be more convenient to use for developers who cannot use the Java Native Interface employed by libSBML to provide its Java layer.

A free, online validation system is provided by SBML.org. Users can interact with it directly through a Web-based user interface or through a network programming interface. It provides the ability to upload a model and have it analyzed for conformance to the SBML specifications. The validation system cannot report whether the model is correct (i.e., the models' behavior may be wrong or pointless), but at least it can determine syntactic validity and consistency of the SBML file.

Documentation

The specification documents that define SBML are freely available from SBML.org. In addition, a list of Frequently Asked Questions and Answers is available, along with other help documents and code samples.

The libSBML library described above comes with extensive documentation for many of the supported programming languages. It also comes with sample programs to help developers get started.

Other Resources

There is an Internet MIME type defined for SBML, defined by RFC 3823 (Kovitz 2004).

SBML.org provides a web forum interface to the several SBML-related mailing lists. The list archives contain many years' worth of discussions about SBML, making this a helpful resource when first starting out programming with SBML.

Cross-References

- ▶ Biological Network Model
- ▶ Biological System Model
- ▶ CellML
- ▶ Collaborative and Distributed Biomedical Applications
- ▶ Data Integration
- ▶ Databases for Kinetic Models
- ▶ Information, Biological
- ▶ Interaction Networks
- ▶ Interoperability
- ▶ Knowledge Representation
- ▶ Mathematical Model, Model Theory
- ▶ Metabolic and Signaling Networks
- ▶ Metabolic Flux Analysis
- ▶ Metabolic Model
- ▶ Metabolic Networks
- ▶ Metabolic Networks, Databases
- ▶ Metabolic Networks, Structure and Dynamics
- ▶ Metabolic Pathway Analysis
- ▶ MIRIAM
- ▶ MIRIAM URI
- ▶ Model Repositories
- ▶ Pathway Modeling, Metabolic
- ▶ Pathway, Functional Units
- ▶ SBGN
- ▶ Semantic Web, Interoperability
- ▶ Specialized Metabolic Component Databases
- ▶ Web Service
- ▶ XML
- ▶ XML Schema

References

Bornstein BJ, Keating SM, Jouraku A, Hucka M (2008) LibSBML: an API library for SBML. Bioinformatics 24:880–8801

Demir E, Cary MP, Paley S, Fukuda K, Lemer C, Vastrik I et al (2010) The BioPAX community standard for pathway data sharing. Nat Biotechnol 28:935–942

Dräger A, Rodriguez N, Dumousseau M, Dörr A, Wrzodek C, Le Novère N, Zell A, Hucka M (2011) JSBML: a flexible Java library for working with SBML. Bioinformatics 27:2167–2178

Hucka M, Finney A, Sauro HM, Bolouri H, Doyle JC, Kitano H et al (2003) The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. Bioinformatics 19:524–531

Kovitz B (2004) MIME media type for the systems biology markup language (SBML). RFC 3823, Accessed June, 2011 http://www.rfc-editor.org/rfc/rfc3823.txt

Le Novère N (2006) Model storage, exchange and integration. BMC Neurosci 7:S11–S11

Le Novère N, Hucka M, Mi H, Moodie S, Schreiber F, Sorokin A et al (2009) The systems biology graphical notation. Nat Biotechnol 27:735–741

Li C, Donizelli M, Rodriguez N, Dharuri H, Endler L, Chelliah V et al (2010) BioModels database: an enhanced, curated and annotated resource for published quantitative kinetic models. BMC Syst Biol 4:92

SBML Team (2011) SBML specification documents. Accessed June, 2011 http://sbml.org/Documents/Specifications
Waltemath D, Adams R, Beard DA, Bergmann FT, Bhalla US, Britten R et al (2011) Minimum information about a simulation experiment (MIASE). PLoS Comput Biol 7: e1001122

Systems Biology of Viral Pathogens

▶ Systems Virology

Systems Biology of Virus–Host Interactions

▶ Systems Virology

Systems Biology Ontology

Nick Juty and Nicolas le Novère
EMBL-European Bioinformatics Institute, Wellcome Trust Genome Campus, Hinxton, Cambridgeshire, UK

Synonyms

SBO

Definition

As the field of computational modeling flourished, it became clear that there was a need to provide a means of supplementing model data with additional information to clarify or specify the semantic content of computational models. Furthermore, this additional semantic information needed to be of a standard form to facilitate interoperability and exchange between different model-encoding formats. Orthogonal, structured controlled vocabularies, comprised of commonly used modeling terms and concepts, were created to meet the requirements of the SBML (▶ Systems Biology Markup Language (SBML)), community (Courtot et al. 2011).

The ontology currently consists of seven orthogonal vocabularies that cover the following: "participant role," to describe component roles, such as "substrate"; "systems description parameter," to describe various parameters and constants, such as the "Michaelis constant"; "mathematical expression," to ascribe particular calculus associations between model parameters and variables, such as "mass action rate law"; "modeling framework," to specify the approach or assumptions made in model creation, such as "logical framework"; "physical entity representation," to define the type of the component within the model, such as "macromolecule"; "occurring entity representation," to define processes that take place, such as "transport reaction," and "metadata representation," to specify the different types of metadata that may be incorporated within a model, such as "database cross reference."

A mechanism to directly incorporate SBO terms within SBML models has been available since Level 2 Version 2, using the attribute "sboTerm," and is described in the SBML specification. It is also directly linked to ▶ SBGN, where each glyph is associated with a specific SBO term, facilitating the conversion between SBML and the graphical SBGN representation.

Systems Biology Ontology is a member of the Open Biomedical Ontology effort (OBO; Smith et al. 2007), and hence committed to ontology development by a community-prescribed set of principles. The ontology can be browsed and downloaded in several formats at http://www.ebi.ac.uk/sbo, and is accessible programmatically via Web Services.

References

Courtot M, Juty N, Knüpfer C, Waltemath D, Zhukova A, Dräger A, Dumontier M, Finney A, Golebiewski M, Hastings J, Hoops S, Keating S, Kell DB, Kerrien S, Lawson J, Lister A, Lu J, Machne R, Mendes P, Pocock M, Rodriguez N, Villeger A, Wilkinson DJ, Wimalaratne S, Laibe C, Hucka M, Le Novère N (2011) Controlled vocabularies and semantics in systems biology. Mol Syst Biol 7:543
Smith B, Ashburner M, Rosse C, Bard J, Bug W, Ceusters W, Goldberg LJ, Eilbeck K, Ireland A, Mungall CJ, Leontis N, The OBI Consortium, Rocca-Serra P, Ruttenberg A, Sansone S-A, Scheuermann RH, Shah N, Whetzel PL, Lewis S (2007) The OBO foundry: coordinated evolution of ontologies to support biomedical data integration. Nat Biotechnol 25:1251–1255