Chapter 1

Trends and Tools for Modeling in Modern Biology

Michael Hucka*
Control and Dynamical Systems, California Institute of Technology, Pasadena, CA 91125, USA

James Schaff
Richard D. Berlin Center for Cell Analysis and Modeling, University of Connecticut Health Center, Farmington, CT 06030, USA

Summary

Computational modeling in biology requires sophisticated software tools. Precise communication and effective sharing of the models developed by researchers requires standard formats for storing, annotating, and exchanging models between software systems. Developing such standards is the driving vision behind the Systems Biology Markup Language (SBML) and several related efforts that we discuss in this chapter. At the same time, such standards are only enablers and ideally should be hidden "under the hood" of modeling environments that provide users with high-level, flexible facilities for working with computational models. As an example of the modern software systems available today, we discuss the Virtual Cell and illustrate its support for typical modeling activities in biology.

* Author for correspondence, e-mail: mhucka@caltech.edu

A. Laisk, L. Nedbal and Govindjee (eds.), Photosynthesis in silico: Understanding Complexity from Molecules to Ecosystems, pp. 3–15.

I. Introduction

Understanding the dynamic processes that are the essence of a living cell stands as one of the most important and most difficult challenges of twenty-first century biology. Today, it is widely appreciated that we can only hope to meet that challenge through the development and application of computational methods (Hartwell et al., 1999; Fraser and Harland, 2000; Arkin, 2001; Tyson et al., 2001; Noble, 2002; Alm and Arkin, 2003; Zerhouni, 2003), particularly the creation of mechanistic, explanatory models illuminating the functional implications of the data upon which they are built.

Models are not substitutes for experiments and data; rather, they are faithful teammates in the process of scientific discovery. A realistic computational model represents a modeler's dynamic understanding of the structure and function of part of a biological system. As the number of researchers constructing realistic models continues to grow, and as the models become ever more sophisticated, they collectively represent a significant accumulation of knowledge about the structural and functional organization of the system. Moreover, using them, the assimilation of new hypotheses and data can be done in a more systematic way because the additions must be fitted into a common, consistent framework. Once properly constructed, the models become a dynamic representation of our current state of understanding of a system in a form that can facilitate communication between researchers and help to direct further experimental investigations (Bower and Bolouri, 2001).

Today's models are large (and growing ever larger) and complex (and getting ever more complex). We are now long past the point of being able to communicate and exchange real-world models effectively by simply summarizing them in written narratives featuring a few equations. The precise communication of computational models between humans and between software is critical to being able to realize modeling's promise. Achieving this requires standardizing the electronic format for representing computational models in a way independent of any particular software – after all, different research goals are often best served by different software tools, yet modelers still need to share their results with their colleagues. At the same time, today's researchers need powerful software environments that offer a range of capabilities to support the creation, analysis, storage and communication of models, all the while hiding the details of the model representation format and providing biological modelers with high-level user interfaces and capabilities matched to the tasks they need to do.

In this chapter, we discuss both standards and software for computational modeling in biology. We summarize the *de facto* standard format, the Systems Biology Markup Language (SBML), as well as ongoing related efforts to standardize the representation of model annotations through MIRIAM (the Minimum Information Requested In the Annotation of biochemical Models) and SBO (the Systems Biology Ontology). As critical as they are, however, such standards are in the end only *enablers*; they are (hopefully) not what users interact with directly. We therefore also discuss software systems, focusing on one in particular, the Virtual Cell, as a way to present typical modeling activities in the context of one of today's most full-featured, interactive modeling environments. The advanced capabilities of systems such as Virtual Cell also help drive further development of SBML and adjunct efforts, and so we close with a summary of present work to extend SBML as well as standardize other areas of modeling and simulation exchange, such as the description of simulations.

Abbreviations: DOI – digital object identifier; MIASE – minimum information about a simulation experiment; MIRIAM – minimum information requested in the annotation of biochemical models; SBGN – systems biology graphical notation; SBML – systems biology markup language; SBO – systems biology ontology; SSA – stochastic simulation algorithm; UML – unified modeling language; URN – uniform resource name; VCell – virtual cell; XML – eXtensible markup language

II. Representing Model Structure and Mathematics

Until the late 1980s, publication of a computational model almost universally involved publishing only the equations and parameter values, usually with some narrative descriptions of how

the model was coded in software and how it was simulated and analyzed. The systems of equations were, with few exceptions, directly implemented in software: in a very direct sense, the program *was* the model. Authors sometimes even wrote their own numerical integration code. This general approach was necessary because of the primitive state of computational platforms and electronic data exchange, and it was fraught with problems. The most significant problem is simply the opportunities for errors that arise when a model must be recapitulated by humans into and back out of natural language form. The degree to which this is a real problem is startling. Curators for databases of published models such as BioModels Database (Le Novère et al., 2006) and JWS Online (Snoep and Olivier, 2003; Olivier and Snoep, 2004), report by personal communication that when they first began operation in the 2000–2004 timeframe, over 95% of published models they encountered had something wrong with them, ranging from typographical errors to missing information (even today, the problem rate is greater than 60%). A second problem is that, when a model is inextricably intertwined with its software implementation, it is difficult to examine and understand the precise details of the actual model (rather than artifacts of its particular realization in software). A third problem is that having to reconstruct a model from a paper is an extremely tall hurdle to fast, efficient and error-free reuse of research results.

Some areas of biological modeling improved on this situation in the 1990s. The field of computational neuroscience was particularly advanced in this regard, having two freely-available simulation packages, GENESIS (Bower and Beeman, 1995; Bower et al., 2002) and NEURON (Hines and Carnevale, 1997), supported on a variety of operating systems. These simulation platforms made it possible for modelers to distribute abstract definitions of their models and simulation procedures in the form of scripts that could be interpreted automatically by the platform software. The approach vastly improved the reusability of models. However, there remained the limitation that the formats were specific to the simulation package in which they were developed. Whoever wanted to reuse the models had to run the same software in order to reuse the model (assuming they were able to get the necessary files from the model's authors – electronic publishing of models as supplements to journal articles was still rare).

With the surge of interest in computational systems biology at the beginning of this century, software tools evolved one step further with the creation of *application-independent* model description formats such as CellML (Hedley et al., 2001) and SBML (Hucka et al., 2003, 2004). This form of representation is not an algorithm or a simulation script; it is a declarative description of the model structure that is then interpreted and translated by each individual software system into whatever internal format it actually uses. No longer tied to a particular software system, such software-independent formats permit a wider variety of experimentation in algorithms, user interfaces, services, and many other aspects of software tool development, by virtue of allowing multiple software authors to explore different facilities that all use the same input/output representation. In addition, and even more significantly, it enables practical publication of models in public databases.

The Systems Biology Markup Language (SBML; http://sbml.org) has become the de facto standard for this purpose, supported by over 120 software systems at the time of this writing. SBML is a machine-readable lingua franca defined neutrally with respect to software tools and programming languages. It is a model definition language intended for use by software – humans are not intended to read and write SBML directly. By supporting SBML as an input and output format, different software tools can all operate on the identical representation of a model, removing opportunities for errors in translation and assuring a common starting point for analyses and simulations. SBML is defined using a subset of UML, the Unified Modeling Language (Booch et al., 2000), and in turn, this is used to define how SBML is expressed in XML, the eXtensible Markup Language (Bray et al., 1998). Software developers can make use of a number of resources for incorporating SBML support in their applications (Bornstein et al., 2008).

SBML can encode models consisting of biochemical entities (species) linked by reactions to form biochemical networks. An important principle in SBML is that models are decomposed into explicitly-labeled constituent elements, the

set of which resembles a verbose rendition of chemical reaction equations; the representation deliberately does not cast the model directly into a set of differential equations or other specific interpretation of the model. This explicit, modeling-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. The main constructs provided in SBML include the following:

Compartment and *compartment type*: a compartment is a container for well-stirred substances where reactions take place, while a compartment type is an SBML construct allowing compartments with similar characteristics to be classified together.

Species and *species type*: a species in SBML is a pool of a chemical substance located in a specific compartment, while species types allow pools of identical kinds of species located in separate compartments to be classified together.

Reaction: a statement describing some transformation, transport or binding process that can change one or more species (each reaction is characterized by the stoichiometry of its products and reactants and optionally by a rate equation).

Parameter: a quantity that has a symbolic name.

Unit definition: a name for a unit used in the expression of quantities in a model.

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.).

Function: a named mathematical function that can be used in place of repeated expressions in rate equations and other formulae.

Event: a set of mathematical formulae evaluated at a specified moment in the time evolution of the system.

The simple formalisms in SBML allow a wide range of biological phenomena to be modeled, including cell signaling, metabolism, gene regulation, and more. Significant flexibility and power comes from the ability to define arbitrary formulae for the rates of change of variables as well as the ability to express other constraints mathematically.

SBML is being developed in "levels". Each higher level adds richness to the model definitions that can be represented by the language. By delimiting sets of features at incremental stages, the SBML development process provides software authors with stable standards and the community can gain experience with the language definitions before new features are introduced. Two levels have been defined so far, named (appropriately enough) Level 1 and Level 2. The former is simpler (but less powerful) than Level 2. The separate levels are intended to coexist; SBML Level 2 does not render Level 1 obsolete. Software tools that do not need or cannot support higher levels can go on using lower levels; tools that can read higher levels are assured of also being able to interpret models defined in the lower levels. Open-source libraries such as libSBML (Bornstein et al., 2008) allow developers to support both Levels 1 and 2 in their software with a minimum amount of effort.

III. Augmenting Models with Semantic Annotations

The ability to have meaningful exchange of complex mathematical models of biological phenomena turns out to require a deeper level of semantic encoding and knowledge management than is embodied by a format such as SBML, which encompasses only syntax and a limited level of semantics. This realization came early in the context of CellML, whose developers added a standard scheme for metadata annotations soon after CellML was developed (Lloyd et al., 2004). CellML's metadata scheme was adopted by SBML at the beginning of the development of SBML Level 2, but limitations with the scheme later led the SBML community to seek alternatives. These were found in the form of the Systems Biology Ontology (SBO; http://www.ebi.ac.uk/SBO; Le Novère et al., 2006), and the Minimum Information Requested in the Annotation of Biochemical Models (MIRIAM; Le Novère et al., 2005).

A. Systems Biology Ontology (SBO)

The rationale for SBO is to provide controlled vocabularies for terms that can be used to annotate components of a model in SBML (or indeed, any other formal model representation format).

It requires no change to the form of the basic model in SBML; rather, it provides the option to augment the basic model with machine-readable labels that can be used by software systems to recognize more of the semantics of the model. SBO provides terms for identifying 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 species and reactions. Recent versions of SBML Level 2 provide an optional attribute on every element where an SBO term may be attached. Table 1.1 lists the correspondences between major components of SBML and SBO vocabularies.

The relationship implied by the attribute value on an SBML model component is "is a": the thing defined by that SBML component "is an" instance of the thing defined in SBO by indicated SBO term. By adding SBO term references on the components of a 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 be. Those tools can then act on that information. For example, if the SBO identifier SBO:0000049 is assigned to the concept of "first-order irreversible mass-action kinetics, continuous framework", and a given reaction in a model has an SBO attribute with this value, then regardless of the identifier and name given to the reaction itself, a software tool could use this to inform users that the reaction is a first-order irreversible mass-action reaction.

Table 1.1. Correspondence between major SBML components and controlled vocabulary branches in the Systems Biology Ontology (SBO)

SBML component	SBO vocabulary
Model	Interaction
Function definition	Mathematical expression
Compartment type	Material entity
Species type	Material entity
Compartment	Material entity
Species	Material entity
Reaction	Interaction
Reaction's kinetic law	Mathematical expression → Rate law
Parameter	Quantitative parameter
Initial assignment	Mathematical expression
Rule	Mathematical expression
Event	Interaction

As a consequence of the structure of SBO, not only children are versions of the parents, but the mathematical expression associated with a child is a version of the mathematical expressions of the parents. This enables a software application to walk up and down the hierarchy and infer relationships that can be used to better interpret a model annotated with SBO terms. Simulation tools can check the consistency of a rate law in an SBML model, convert reactions from one modeling framework to another (e.g., continuous to discrete), or distinguish between identical mathematical expressions based on different assumptions (e.g., Henri-Michaelis-Menten vs. Briggs-Haldane). Other tools like SBMLmerge (Schulz et al., 2006) can use SBO annotations to integrate individual models into a larger one.

SBO adds a semantic layer to the formal representation of models, resulting in a more complete definition of the structure and meaning of a model. The presence of an SBO label on a compartment, species, or reaction, can also help map SBML elements to equivalents in other standards, such as (but not limited to) BioPAX (http://www.biopax.org) or the Systems Biology Graphical Notation (SBGN, http://www.sbgn.org). Such mappings can be used in conversion procedures, or to build interfaces, with SBO becoming a kind of "glue" between standards of representation.

B. Minimum Information Requested in the Annotation of Biochemical Models (MIRIAM)

While SBO annotations help add semantics, there remains a different kind of impediment to effective sharing and interpretation of computational models. Figure 1.1 illustrates the issue.

When a researcher develops a model, they often use simple identifiers for chemical substances, or at best, only one of a multitude of possible synonyms for the substance. The situation is even worse when it comes to the chemical reaction and other processes: these are often given names such as "R1", "R2", etc., or at best, generic

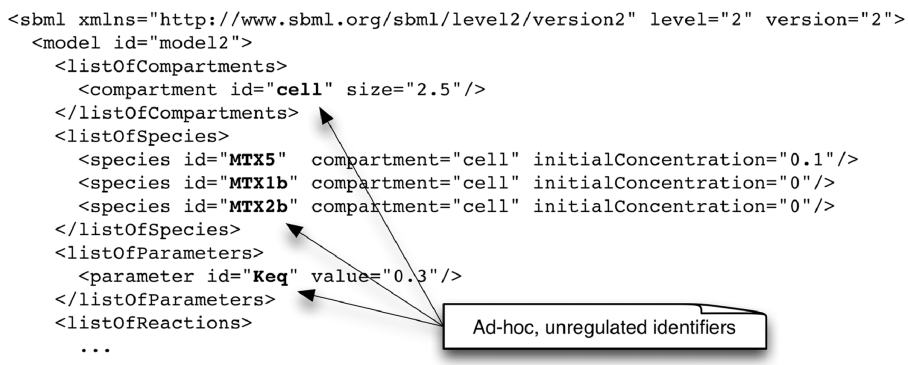

Fig. 1.1. An example fragment of an SBML file. The id fields in the lines above establish the identifiers of entities used in the model. This particular model contains a compartment identified only as "cell"; three biochemical species identified as "MTX5", "MTX1b" and "MTX2b"; and a global parameter (constant) identified as "Keq". These labels presumably have meaning to the creator of the model, but rarely to its readers, and even less so to software tools. Yet, such short identifiers are really what modelers often use in real-life models. It is not in the scope of SBML to regulate or restrict what the identifiers can or should be – a different approach is needed. The solution in use today is to provide a mechanism for augmenting (not replacing) the identifiers with annotations referring to regulated terms in "dictionaries", controlled vocabularies, or entries in databases that provide detailed information about the biological entities to which the identifiers are meant to refer.

terms such as "mass-action" that do not reflect the role of the reaction as a process in the broader model. Searching for models based on useful criteria is next to impossible under these conditions. One could blame modelers for not being more thorough in naming and identifying the elements in their models; one could also blame software tools for not assisting modelers in this process. However, such criticisms would be both futile and misplaced. First, this situation is the reality for thousands of existing models and it is likely to persist into the foreseeable future. Second, different research subfields often have different names for the same chemical species, processes, and other concepts. Who would decide which is most appropriate to use?

The most practical solution found so far by the computational systems biology community is to augment models with annotations that provide links between the elements of a model and other (external) data resources and models. However, the potential and power of annotations is largely lost if the format of the annotations is not standardized to the point where different software systems can interpret them in the same way. This was one of the motivations for the development of MIRIAM, a set of guidelines for the Minimum Information Requested In the Annotation of biochemical Models (Le Novère et al., 2005) encoded in a structured representation format such as SBML.

MIRIAM defines both (1) minimum consistency requirements for a model, and (2) a regular and simple annotation scheme for linking a model to its sources and linking model components to external data resources. The goal of the first aspect of MIRIAM is to ensure that a model is reliably attributed to a reference description (which is a document describing or referencing a description of the model, the model's structure, numerical values necessary to instantiate a simulation, and the results to be expected from such a simulation) and is consistent with that description. The requirements apply to the model as a whole and are irrespective of any annotations placed in it. Table 1.2 summarizes these minimal requirements for reference correspondence.

The second broad aspect of the MIRIAM guidelines concerns the annotation content. The requirements for minimum attribution information are summarized in Table 1.3. They simply represent a basic level of information that is deemed to be necessary in order for a model's readers to be able to associate the model with a reference description and a process used to encode the model in the structured format.

Table 1.2. MIRIAM guidelines for minimum consistency of a model

- The model must be encoded in a public, machine-readable format, either standardized (e.g., SBML) or supported by specific applications (e.g., MATLAB).
- The encoded model must comply with the standard in which it is encoded, meaning that the syntax must be correct and the model must pass validation.
- The model must be related to a single description that describes or references results that one can expect to reproduce using the model. If the model is derived from several sources (e.g., several publications), there must exist a single reference description associated with the combined model. Note that MIRIAM does not require the reference description to be published; it must merely be made available to consumers of the model.
- The model's structure must reflect the biological processes listed in the reference description.
- The encoded model must be instantiated in simulation, which implies that quantitative attributes, initial conditions, parameters, etc., must all be defined. (The actual values may be provided as a separate file from the model itself.)
- When instantiated in a suitable environment, the model must be able to reproduce relevant and readily-simulated results given in the reference description, and the results must be quantitatively similar (with any differences being attributable to differences in algorithm roundup errors).

Table 1.3. MIRIAM guidelines for the minimum attribution information to be provided with a model

- A (preferred) name for the model.
- A citation for the reference description. This can be bibliographic information, or a unique identifier (e.g., DOI), or even a URL pointing directly at the description – something to locate and identify the reference description and its authors.
- Name and contact information for the model creator(s).
- Date and time of creation.
- Date and time of last modification.
- A precise statement of the encoded model's terms of distribution. MIRIAM does not require freedom of distribution nor no-cost distribution, only a statement of what the distribution terms are.

As for the manner in which annotations are to be represented, the MIRIAM scheme is simple and does not require a particular format structured – in fact, the annotations can be recorded in something as simple as a separate text file, though whatever method is used, the annotations must always be transferred with the model. Each annotation is a triplet consisting of a data type, an identifier, and 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 (Jacobs and Walsh, 2004). The identifier refers/points to a specific datum in whatever source is identified by the data type. 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. If the qualifier is absent, the assumed relationship is "is".

IV. Connecting Models to Results

SBML, MIRIAM, and related technologies are all meant to be *under the hood*, so to speak, with software systems reading and writing models in SBML form (annotated in MIRIAM-compliant fashion), but ideally without exposing this level of detail to users. In this section, we examine how one particular modeling environment, the Virtual Cell, provides a wide range of modeling facilities while effectively hiding the details of interacting with models in SBML form. The Virtual Cell is an example of the modern trend towards providing powerful, general-purpose modeling environments supporting the whole gamut of tasks that biologist-modelers must do, from importing experimental data, to deriving a family of models from the data, to simulating and analyzing the models, and relating the results of the analyses back to the experimental data.

A. Common Experimental and Modeling Activities

To compare experimental results with the quantitative predictions generated from a biological model, the model must be exercised in a manner consistent with experimental protocols and apparatus. Experimental protocols are standardized procedures for experimental measurements and manipulations. Some protocols can be described simply as ideal processes that directly

perturb only the desired target or are perfect observers of some physiological states or anatomical structures. More realistic notions are routinely considered by the experimentalist who must parse unwanted behaviors and distortions to gain insight into a biological process. They design experiments to reduce the sensitivity of their results to those unwanted artifacts.

Before associating raw experimental data with model predictions, both the raw experimental data and the model predictions are often post-processed into quantities that can be more easily compared. This post-processing can suffice if we assume that measurements and manipulations are ideal and that the experimental intervention is separable from the biological processes. If these assumptions do not hold (e.g., when a fluorescent indicator functionally modifies or sequesters its ligand), changes to the model structure are required to properly represent the interaction of the protocols with the underlying biological system.

Validation of a quantitative model against multiple experiments typically requires the creation of a number of experiment-specific models that must retain a consistent core representing the physiological mechanisms and hypotheses. As new experimental evidence is considered, additional experiment specific models are created. The underlying mechanistic model evolves in a way that corresponds to the entire set of experimental data. During this process it is important to continually reassess the compatibility between the context of each experiment and the current model structure, parameters, and modeling assumptions.

A list of accumulated model assumptions should be maintained explicitly to identify contradictions and track applicability of experimental data. There are explicit modeling assumptions introduced by the physical approximations used within the model (e.g., preconditions for use of a particular kinetic law) as well as those imposed by the modeling framework (e.g., well-mixed compartments ignore spatial variation, deterministic population dynamics ignores stochastic variation). There are also implicit modeling assumptions when deciding which elements can be safely omitted from a model and when introducing functional dependencies. This growing list of assumptions and the collection of experimental data constrain the feasible space of consistent model structures and parameters.

This process of using new experimental data to refine models is important, but the reverse process is even more useful. Experimentalists and modelers working together can use interesting model predictions to suggest new experiments that can help discriminate between alternative model structures (alternative hypotheses) or help to define the boundaries of the model's domain of applicability.

B. Supporting Modeling Activities Through Software Environments

The Virtual Cell Modeling and Simulation framework (VCell; Slepchenko et al., 2003; Moraru et al.; 2008) will be used as an example architecture to describe how modeling tools can support these capabilities. VCell was developed by an interdisciplinary team of engineers, physicists, biologists and mathematicians who were also doing modeling in close collaboration with experimental biologists. Over time, interactions with the growing user community, especially at an intensive annual short course, have helped to keep VCell relevant and usable for modeling experimental biology.

The Virtual Cell supports modeling and simulating reaction networks, diffusive and advective transport, and electrophysiology. The system provides clear conceptual boundaries designed to maximize the reusability of a single mechanistic physiological model in multiple experimental contexts. A VCell "BioModel" is a biological model (as opposed to VCell's equation-based models) containing a single physiological model and multiple "Applications", each of which corresponds to or "applies" the model to an experimental context. Each VCell Application defines its modeling framework (i.e., spatial-deterministic, compartmental-deterministic, and compartmental-stochastic) and captures experimental data, cellular geometry, initial conditions, boundary conditions, knockouts, pseudo steady state approximations, and electrophysiological protocols sufficient for automatically generating an experiment-specific mathematical model.

VCell was developed with an emphasis on spatial modeling where reaction mechanisms have traditionally been described locally, that is, affecting local concentrations at a rate influenced only by local concentrations. When mapped spatially,

these local interactions are defined at each point in the appropriate domain; for example, membrane reactions are defined at each point on the membrane, and membrane/volumetric transport results in a spatially resolved flux density that is coupled to diffusive flux in the volume. When mapped non-spatially, these local interactions are integrated over the domains in which they are defined; for example, flux densities produced from membrane reactions are integrated over the membrane to produce a total flux. Support for spatial models is one of the areas of development for SBML Level 3.

VCell encourages modelers to represent a hypothesized indirect interaction between distant molecular species not as a single reaction, but as a series of local reactions and transport steps. In this way, each of these individual mechanisms must be physically realizable (e.g., reasonable kinetic parameters, concentrations, and transport rates) while combining to produce the desired behavior. While the absolute values of intermediate parameters may be underdetermined, the bounds on these parameters can introduce absolute physical limits on the time scale or sensitivity of an indirect functional relationship.

Modeling an indirect interaction between distant molecular species as a single reaction necessarily omits transport mechanisms or second messengers that act on an appropriately fast time scale. The advantage of these approximations is their simplicity and direct correspondence to some measured quantity without adding additional degrees of freedom; the disadvantage is that it introduces a phenomenological process that will not generalize well and will be insensitive to other parts of your model. Despite this, models containing indirect interactions are quite popular in many research domains. VCell has recently been extended to allow both local and non-local reaction mechanisms, where non-local reactions can only be mapped non-spatially or to molecular species that are constrained to be well-mixed.

For deterministic spatial modeling (partial differential equations), a spatial "Application" maps a user's core physiological model to a three-dimensional cellular geometry (often derived from microscopy images) that supports heterogeneous distributions of processes and molecular species and allows definition of diffusive and advective transport. In addition, all model parameters, initial conditions, boundary conditions and mechanisms can be explicit functions of time and space or derived from user supplied spatiotemporal data (e.g., experimental time series images). The experimental time series can be compared and visualized together with the spatiotemporal simulation results. For deterministic compartmental modeling (differential algebraic equations), the same core physiological model can be mapped to well-stirred compartments and associated with user defined time series datasets for parameter estimation. For stochastic compartmental modeling (Poisson processes) the core physiological model is mapped to jump processes with Poisson distributions and simulated with direct and hybrid solvers, but there is no corresponding experimental data handling at this time.

Electrophysiological modeling protocols are seamlessly integrated into the VCell "Application" as a Protocol Module. To simulate an electrophysiological experiment, the user selects where to place the patch-clamp electrodes and specifies a waveform for either a current-clamp or voltage-clamp protocol. Then, whenever the mathematical model for that Application is generated (preceding simulation or analysis), the appropriate electrical device (either a current source for current clamp or a voltage source for voltage clamp) is temporarily inserted across the appropriate membrane. This alters the "equivalent circuit" of the model so that the proper set of equations is generated. For either protocol, both the voltage and applied current (sum of capacitive and transmembrane currents) between the electrodes is computed so that the user can directly compare the simulated currents with an experimental recording.

A VCell "Application" adds crucial contextual information, such as initial conditions, reactions, boundary conditions, spatial domain (cellular geometry), the concentrations to hold fixed, and other characteristics, to a mechanistic model. The result completely specifies a mathematical model used to generate simulations. This approach allows several experiment-centric derived models – the Virtual Cell Applications – to be maintained as a single document. The original intent in VCell was to validate a single underlying biological model under several independent experimental conditions. However, some experimental measurements

and manipulations must be modeled by explicitly coupling the experimental processes and the biological processes (e.g., fluorescent calcium indicators also function as significant calcium buffers and so competes for free calcium and changes effective diffusion). Rather than require a modeler to manually incorporate the experimental process directly into a physiological model and thus destroy the model's reusability, it is more flexible to automatically generate the augmented experimentally-focused model as needed. The modeler can define which protocols to include and specify the required parameters (e.g. total fluorescent indicator added, affinity/kinetics with respect to each existing molecule, diffusion rate, bleaching characteristics). A "virtual experiment" will then be a comprehensive, experimentally-focused extension of a modeling application that will provide explicit representations for input data, protocols, measurement processes, and experimental reference data.

Note how this kind of separation between a model and its manipulations in a virtual experiment implies a need for a representation that is separate from SBML – which is precisely the reason why the MIASE project (Minimum Information About a Simulation Experiment; http://www.ebi.ac.uk/compneur-srv/miase/) was initiated. MIASE aims to represent in an application-independent way the common set of information that any modeler needs to provide in order to repeat a numerical simulation experiment derived from a given quantitative model.

Quantitative models should embody mechanistic hypotheses within a consistent theoretical framework. A physics-based framework within a modeling tool provides a scaffold consisting of implicit conservation laws (e.g., mass conservation) that couple physical quantities and processes (e.g. biochemical reactions, patch-clamp electrodes, and measurement processes) to generate a mathematical model. In contrast, an equation-based framework (e.g. model explicitly defines entire system of differential-algebraic equations) requires that all relationships between data and model must be considered explicitly in the form of the model. However, the experimental conditions and apparatus often significantly change the mathematical form of the system (e.g. voltage-clamp electrodes, buffering effects of a fluorescent indicator, over-expression of a fluorescently labeled protein). SBML requires a physics-based framework for reactions with all other processes defined using ancillary equations added directly to the model. VCell extends the supported physical processes that can be described in models to include diffusive and advective transport in space and electrophysiology (e.g. Kirchhoff's Voltage and Current Laws are considered when constructing the equations for electric potential), and "Virtual Microscopy" extensions. These extensions support protocols incorporating fluorescent indicators, fluorescent labels, FRAP, photoactivation, focal stimuli while considering experimental optics.

Table 1.4 provides a summary of how many of the concepts in SBML map to concepts in VCell. Some of the areas of current development in SBML are discussed in the next section.

Table 1.4. SBML components versus their VCell counterparts

SBML component	Location in VCell component
Reactions	Reactions in model
Fast attribute for reaction	Fast reactions in application
Reaction kinetics	Lumped or local kinetics (in model)
Species	Species in model
Species initial conditions	Initial conditions in application
Compartment	Compartment in model
Compartment size	Spatial characteristics in application
Rate rules, algebraic rules, events	Expressions in application (*future version*)
Rate rules for membrane potential	Expressions in application
Spatial package (in development)	Diffusion/advection + geometry (Application)
SBML model	Model + 1 application
Multiple SBML models	BioModel (multiple applications)
MIASE	Simulation

V. Future Directions for Systems Biology Markup Language (SBML)

The needs of advanced modeling environments such as VCell are pushing forward the evolution of SBML and associated standards. The next generation of SBML, called SBML Level 3, will be a modular language based on a core and extension "packages" layered on top of the core. The core will be a minimally-modified Level 2 Version 4. One of the modifications will be a package mechanism that allows a model to declare which additional feature sets (packages) are used by the model. Software tools will be able to use this information to judge whether they can fully interpret a model that they encounter. Advanced software such as VCell will gain the ability to more fully represent classes of models that currently are not supported in SBML Level 2; on the other hand, software tools that do *not* support certain features used in a given model can inform users that only limited functionality can be provided – yet still be able to make *some* use of the SBML model by virtue of its use of core features. There are several SBML Level 3 packages in development today, but here we describe briefly just two, spatial geometry and hierarchical model composition. More information about these and other activities can be found online at the SBML website, http://sbml.org/.

The goal of supporting *spatial characteristics* in SBML Level 3 is to allow the representation of the geometric features of compartments and the spatial distribution of model quantities and processes. SBML models today are nonspatial: compartments are topological structures only, with dimensionality, size and containment being their only physical attributes. This was partly a conscious design decision, because even today, there are far more nonspatial modeling tools available, and so the SBML development priority reflected that. However, it is clearly insufficient for many potential modeling uses, including the problems described above. The focus for the spatial aspects of SBML will be on supporting at least the following characteristics: (1) the size and shape of physical entities whether compartments or reacting species; (2) the absolute or relative spatial location of reacting species in compartments, for instance in a volume, on membrane surfaces, or along microtubules; (3) the rates of diffusion of species through compartments; and (4) the definition of rate equations and algebraic constraints describing phenomena either at specific locations, or distributed across compartments.

Model composition refers to the ability to include models as submodels inside other models. This requires defining the interfaces between the models and rules for connecting parts of models together. The motivation is to help contain model complexity by allowing decomposition. With this facility in place, users will be able to create reusable models, create libraries of components, etc., and combine them into larger models, much as is done in software development, electronics design, and other engineering fields.

VI. Conclusions

The use of computational modeling is clearly increasing in all areas of biology, from analyzing and extracting understanding from the vast quantities of data saturating researchers today, to designing biological circuits (Church, 2005). One of the most valuable features of computational models is their support of quantitative calculations, allowing researchers not only to test their understanding, but also to explore "what-if" scenarios and make testable predictions about the behavior of the system being studied. This is an essential requirement for being able to understand complicated systems that are replete with feedback mechanisms (the hallmark of biological systems), where the resulting behaviors are rarely predictable through intuitive reasoning alone. Even for the simplest components and systems, it can be impossible to predict such characteristics as sensitivity to exact parameter values without constructing and analyzing a model. Such analyses have shown that some systems are insensitive (e.g. Yi et al., 2000) whereas others are exquisitely sensitive (e.g. McAdams and Arkin, 1999). Computational modeling is thus an extension of the scientific method (Phair and Misteli, 2001; Fall et al., 2002; Slepchenko et al., 2002), providing the means to create precise, unambiguous, quantitative descriptions of biological phenomena that can be used to evaluate hypotheses systematically and to explore non-obvious dynamical

behavior of a biological system (Hartwell et al., 1999; Endy and Brent, 2001; Csete and Doyle, 2002).

The inescapable reality in systems biology is that models (that is to say, hypotheses cast in a computational form) will continue to grow in size, complexity and scope. New tools for gaining greater biological information ensure future revelations will continue to be uncovered at an ever increasing pace. Standardizing on common formats is essential for being able to move forward with increasingly larger-scale research endeavors. As discussed in this chapter, SBML in combination with other standards today permits representing computational models in a way that is independent of any particular software package, operating system, or simulation algorithms. Standardization of this kind removes an impediment to sharing results and permits other researchers to start with an unambiguous representation of their hypotheses and assumptions, thus able to examine it carefully, propose precise corrections and extensions, and apply new techniques and approaches – in short, to do better science.

In part because this standardization has encouraged attempts at collaboration and exchange like never before, limitations in the existing standards such as SBML are being recognized and being addressed. Modern software environments for modeling, such as the Virtual Cell described in this chapter, offer capabilities that exceed what can be represented in SBML Level 2 today. As a result of this and other inspirations, the SBML community is working on SBML Level 3, promising new capabilities for representing such things as spatial geometries and diffusion processes in models. This is one of the beneficial effects of increased collaboration enabled by standardized formats such as SBML: researchers and developers push the standards and encourage their continued evolution and expansion, which in turn encourages more collaboration and development. This feedback loop ensures that the future of computational modeling in biology looks brighter than ever.

References

Alm E and Arkin AP (2003) Biological networks. Curr Opin Struc Biol 13: 193–202

Arkin A (2001) Synthetic cell biology. Curr Opin Biotech 12: 638–644

Booch G, Jacobson I and Rumbaugh J (2000) OMG Unified Modeling Language Specification. Object Management Group, Inc. Needham, MA, USA

Bornstein BJ, Keating SM, Jouraku A and Hucka M (2008) LibSBML: An API library for SBML. Bioinformatics 24: 880–881

Bower JM and Beeman D (1995) The Book of GENESIS: Exploring Realistic Neural Models with the GEneral NEural SImulation System. Springer, Santa Clara, California

Bower JM and Bolouri H (2001) Computational Modeling of Genetic and Biochemical Networks. MIT Press, Cambridge, MA

Bower JM, Beeman D and Hucka M (2002) The GENESIS simulation system. In: Arbib MA (ed) The Handbook of Brain Theory and Neural Networks pp. 475–478. MIT Press, Cambridge, MA

Bray T, Paoli J and Sperberg-McQueen CM (1998) Extensible markup language (xml) 1.0 (w3c recommendation 10-february-1998). http://www. w3.org/TR/

Church GM (2005) From systems biology to synthetic biology. Mol Sys Biol 1, p. 1, doi:10.1038/msb4100007

Csete ME and Doyle JC (2002) Reverse engineering of biological complexity. Science 295: 1664–1669

Endy D and Brent R (2001) Modelling cellular behaviour. Nature 409: 391–395

Fall C, Marland ES, Wagner JM and Tyson JJ (2002) Computational Cell Biology. Springer, New York

Fraser SE and Harland RM (2000) The molecular metamorphosis of experimental embryology. Cell 100: 41–55

Hartwell LH, Hopfield JJ, Leibler S and Murray AW (1999) From molecular to modular cell biology. Nature 402: C47–C52

Hedley WJ, Nelson MR, Bullivant DP and Nielson PF (2001) A short introduction to CellML. Phil Trans R Soc A 359: 1073–1089

Hines ML and Carnevale NT (1997) The NEURON simulation environment. Neural Comput 9: 1179–1209

Hucka M, Finney A, Sauro HM, Bolouri H, Doyle JC, Kitano H, Arkin AP, Bornstein BJ, Bray D, Cornish-Bowden A, Cuellar AA, Dronov S, Gilles ED, Ginkel M, Gor V, Goryanin II, Hedley WJ, Hodgman TC, Hofmeyr J-H, Hunter PJ, Juty NS, Kasberger JL, Kremling A, Kummer U, Le Novère N, Loew LM, Lucio D, Mendes P, Minch E, Mjolsness ED, Nakayama Y, Nelson MR, Nielsen PF, Sakurada T, Schaff JC, Shapiro BE, Shimizu TS, Spence HD, Stelling J, Takahashi K, Tomita M, Wagner J and Wang J (2003) The Systems Biology Markup Language (SBML): A medium for representation and exchange of biochemical network models. Bioinformatics 19: 524–531

Hucka M, Finney A, Bornstein BJ, Keating SM, Shapiro BE, Matthews J, Kovitz BL, Schilstra MJ, Funahashi A, Doyle JC and Kitano H (2004) Evolving a lingua franca and associated software infrastructure for computational systems biology: The Systems Biology Markup Language (SBML) project. Sys Biol 1: 41–53

Jacobs I and Walsh N (2004) Architecture of the world wide web, volume one: W3c recommendation 15 December 2004. W3C. http://www.w3.org/ TR/webarch/

Le Novère N, Finney A, Hucka M, Bhalla US, Campagne F, Collado-Vides J, Crampin EJ, Halstead M, Klipp E, Mendes P, Nielsen P, Sauro H, Shapiro BE, Snoep JL, Spence HD and Wanner BL (2005) Minimum Information Requested in the Annotation of biochemical Models (MIRIAM). Nat Biotechnol 23: 1509–1515

Le Novère N, Bornstein BJ, Broicher A, Courtot M, Donizelli M, Dharuri H, Li L, Sauro HM, Schilstra MJ, Shapiro BE, Snoep JL and Hucka M (2006) Biomodels database: A free, centralized database of curated, published, quantitative kinetic models of biochemical and cellular systems. Nucleic Acids Res 34: D689–D691

Lloyd CM, Halstead MD and Nielsen PF (2004) CellML: Its future, present and past. Prog Biophys Mol Biol 85: 433–450

McAdams HH and Arkin A (1999) It's a noisy business! Genetic regulation at the nanomolar scale. Trends Genet 15: 65–69

Moraru II, Schaff JC, Slepchenko BM, Blinov ML, Morgan F, Lakshminarayana A, Gao F, Li Y, Loew LM (2008) Virtual Cell modelling and simulation software environment. IET Sys Biol 2(5): 352–362

Noble D (2002) The rise of computational biology. Nat Rev Mol Cell Bio 3: 460–463

Olivier BG and Snoep JL (2004) Web-based kinetic modelling using JWS online. Bioinformatics 20: 2143–2144

Phair RD and Misteli T (2001) Kinetic modelling approaches to *in vivo* imaging. Nat Rev Mol Cell Bio 2: 898–907

Schulz M, Uhlendorf J, Klipp E and Liebermeister W (2006) SBMLmerge, a system for combining biochemical network models. Genome Inform 17: 62–71

Slepchenko BM, Schaff JC, Carson JH and Loew LM (2002) Computational cell biology: Spatiotemporal simulation of cellular events. Annu Rev Bioph Biom 31: 423–441

Slepchenko BM, Schaff JC, Macara I and Loew LM (2003) Quantitative cell biology with the virtual cell. Trends Cell Biol 13: 570–576

Snoep JL and Olivier BG (2003) JWS online cellular systems modeling and microbiology. Microbiology 149: 3045–3047

Tyson JJ, Chen K and Novak B (2001) Network dynamics and cell physiology. Nat Rev Mol Cell Biol 2: 908–916

Yi T-M, Huang Y, Simon MI and Doyle J (2000) Robust perfect adaptation in bacterial chemotaxis through integral feedback control. Proc Natl Acad Sci USA 97: 4649–4653

Zerhouni E (2003) The NIH roadmap. Science 302: 63–64