



ELSEVIER

Available online at www.sciencedirect.com

SCIENCE @ DIRECT®

Progress in Biophysics and Molecular Biology 90 (2006) 360–377

www.elsevier.com/locate/pbiomolbio

Progress in
Biophysics
& Molecular
Biology

Review

simBio: A Java package for the development of detailed cell models

Nobuaki Sarai*, Satoshi Matsuoka, Akinori Noma

*Department of Physiology and Biophysics, Kyoto University Graduate School of Medicine,
Yoshidakonoe-cho, Sakyo-ku, Kyoto 606-8501, Japan*

Available online 13 June 2005

Abstract

Quantitative dynamic computer models, which integrate a variety of molecular functions into a cell model, provide a powerful tool to create and test working hypotheses. We have developed a new modeling tool, the *simBio* package (freely available from <http://www.sim-bio.org/>), which can be used for constructing cell models, such as cardiac cells (the Kyoto model from Matsuoka et al., 2003, 2004a, b, the LRd model from Faber and Rudy, 2000, and the Noble 98 model from Noble et al., 1998), epithelial cells (Strieter et al., 1990) and pancreatic β cells (Magnus and Keizer, 1998). The *simBio* package is written in Java, uses XML and can solve ordinary differential equations. In an attempt to mimic biological functional structures, a cell model is, in *simBio*, composed of independent functional modules called *Reactors*, such as ion channels and the sarcoplasmic reticulum, and dynamic variables called *Nodes*, such as ion concentrations. The interactions between *Reactors* and *Nodes* are described by the graph theory and the resulting graph represents a blueprint of an intricate cellular system. *Reactors* are prepared in a hierarchical order, in analogy to the biological classification. Each *Reactor* can be composed or improved independently, and can easily be reused for different models. This way of building models, through the combination of various modules, is enabled through the use of object-oriented programming concepts. Thus, *simBio* is a straightforward system for the creation of a variety of cell models on a common database of functional modules.

© 2005 Elsevier Ltd. All rights reserved.

Keywords: Computational cell biology; Mathematical modeling; Detailed cell model

*Corresponding author. Tel.: +81 75 753 4357; fax: +81 75 753 4349.

E-mail address: sarai@card.med.kyoto-u.ac.jp (N. Sarai).

URL: <http://www.card.med.kyoto-u.ac.jp>.

Contents

1. Introduction	361
2. Methods	364
2.1. Define <i>Nodes</i> and <i>Reactors</i>	364
2.2. <i>Reactor</i> as a functional module	365
2.3. Composition of the cell model.	366
2.4. Simulation system	367
3. Results	368
4. Discussion.	374
4.1. Motivation	374
4.2. Model expansion	374
4.3. Model sharing	374
Acknowledgements.	375
References	376

1. Introduction

Although the decoding of the human genome has almost been completed, the complex mechanisms underlying biological activities are still not well understood. One reason is that the function of many proteins and interaction among proteins are still unknown. Databases of static cellular pathways, such as KEGG (Table 1, entry #2), have been constructed to provide a way to abstract knowledge and principles from large scale databases of genomes and proteomes. Furthermore, the field of systems biology, whose aim is to elucidate the dynamic characteristics of a complex system (Kitano, 2002), has emerged. Thus, the creation of dynamic computational models has been long awaited to clarify the physiological functions at the whole-cell level (Noble, 2002).

Recently, the number of computational model studies has increased, especially in the field of electrophysiology, where quantitative data are numerous. Nevertheless, it still takes time to evaluate or combine different models, due to the lack of critical information necessary for executing model programs written in different contexts. To solve this, many tools have been developed, as listed in Table 1. For example, markup languages such as CellML (Table 1, entry #4) or SBML (Table 1, entry #5) for describing cellular processes, a database such as PATIKA (Table 1, entry #3) for the ontology of cell functions, or common simulation environments such as E-Cell (Table 1, entry #13), Virtual Cell (Table 1, entry #27), COR (Table 1, entry #10), and JSim (Table 1, entry #20), to name but a few. In addition, many tools in systems biology are useful for making pathway databases of proteins and/or genes, based on the graph theory, and for the simulation of system behavior. In general, metabolic models contain a restricted number of reactions and a huge variety of reactants. However, in real cells reaction pathways are highly diverse. For example, a cardiac excitation–contraction model (the Kyoto model from Matsuoka et al., 2003, 2004a, b) was made of over 100 types of reaction equations and less than 20 reactants. Thus, it is important to handle the equations of many different disciplines, in addition to reactant databases and graphical pathway editors, when developing a comprehensive cell model.

Moreover, biological functions are expressed at various hierarchical levels (e.g. functional proteins such as ion channels, membrane, intracellular organelles, cells, tissues, whole organs, or

Table 1

Tools for mathematical modeling in biology

Database

- (1) BioPAX: Biological pathways exchange is a collaborative effort to create a data exchange format for biological pathway data. <http://www.biopax.org/>
- (2) KEGG: Kyoto encyclopedia of genes and genomes (Kanehisa and Bork, 2003). Collection of static pathway data from metabolism. <http://www.kegg.org/>
- (3) PATIKA: Pathway analysis tool for integration and knowledge acquisition (Demir et al., 2002). The network of cellular events is stored at the server-side database via graphical user interface. <http://www.patika.org/>

Biological model description language

- (4) CellML is a computer-readable format to store and exchange mathematical models (Lloyd et al., 2004). Biological models are composed of dynamic reactions described in MathML. <http://www.cellml.org/>
- (5) SBML: Systems biology markup language is a computer-readable format for representing models of biochemical reaction networks based on XML (Finney and Hucka, 2003). Dynamic reactions can be documented in the MathML format. <http://www.sbml.org/>

Interface for biological modeling tools

- (6) BioSPICE: A computational infrastructure for systems biology (Garvey et al., 2003) <https://biospice.org/>
- (7) BioUML: Java framework for systems biology (Kolpakov, 2002). It spans the comprehensive range of capabilities including access to databases with experimental data, tools for formalized description of biological systems structure and functioning, as well as tools for their visualization and simulations. <http://www.biouml.org/>
- (8) SBW: Systems biology workbench (Sauro et al., 2003). The SBW is a modular, broker-based, message-passing framework for communication between applications that aid in research in systems biology. <http://sbw.sourceforge.net/>

ODE solver (with pathway modeler)

- (9) CelleratorTM is a Mathematica[®] package designed to facilitate biological modeling via automated equation generation (Shapiro et al., 2003). <http://www.cellerator.info/>
- (10) Cellular Open Resource (COR): A modeling environment to handle cardiac modeling and reaction diffusion problems (Garny et al., 2003). <http://cor.physiol.ox.ac.uk/>
- (11) CESE: Cell electrophysiology simulation environment is a comprehensive framework specifically designed to perform computational electrophysiological simulations. <http://cese.sourceforge.net/>
- (12) DBsolve: Mathematical simulation and analysis of cellular metabolism and regulation (Goryanin et al., 1999). http://biosim.genebee.msu.su/dbdownload5_en.php
- (13) E-Cell: aims to model and reconstruct biological phenomena in silico, and to allow precise whole cell simulation (Tomita et al., 1999). A cell model is composed of variables, processes, and systems. <http://www.e-cell.org/>
- (14) Genomic object net project: a tool for constructing pathway models and simulating pathway mechanisms on hybrid functional Petri net (Matsuno et al., 2003). <http://www.genomicobject.net/>
- (15) Gepasi: simulates the kinetics of systems consisting of biochemical reactions and provides a number of tools to fit models to data, to optimize functions of models, and to perform metabolic control and linear stability analysis (Mendes et al., 2003). <http://www.gepasi.org/>
- (16) iCell: Java simulations of cellular bioelectric activity on the web-server. <http://ssd1.bme.memphis.edu/icell/index.html>
- (17) JAMES: Discrete event simulation for a better understanding of metabolite channeling—A system theoretic approach. <http://www.informatik.uni-rostock.de/FB/Praktik/Mosi/en/research/projects/james.html>
- (18) Jarnac/JDesigner: Jarnac is a language for describing and manipulating cellular system models and can be used to describe metabolic, signal transduction and gene networks, or in fact any physical system which can be described in terms of a network and associated flows. JDesigner is a Win32 application which allows to draw a biochemical network and export the network in the form of SBML. (Sauro et al., 2003) <http://www.cds.caltech.edu/~hsauro/>

Table 1 (continued)

-
- (19) JigCell: Model builder, run manager, comparator, and automatic parameter estimator for studying complex biochemical regulatory systems in general, and the cell cycle control system in particular (Allen et al., 2003). <http://jigcell.biol.vt.edu/>
 - (20) JSim: A software environment for scientific modeling that provides tools for development of models, for their runtime control, and for analysis of their behavior. (Butterworth and Bassingthwaite) <http://nsr.bioeng.washington.edu/PLN/>
 - (21) Kinetikit: An interface and utility for developing simulations of chemical kinetics. <http://www.ncbs.res.in/~bhalla/kkit>
 - (22) LabHEART: An interactive computer model of rabbit ventricular myocyte (Puglisi and Bers, 2001).
 - (23) MATLAB: Common software package useful for solving all kinds of mathematical problems provided by MathWorks, MA, USA. <http://www.mathworks.com/>
 - (24) PhysioLab: A comprehensive disease map, a knowledge management infrastructure, a virtual research environment for disease models by Entelos, CA, USA (Michelson, 2003). <http://www.entelos.com/>
 - (25) PySCeS: Python simulator for cellular systems. For a network of coupled reactions it provides a stoichiometric matrix analysis, calculates the time course and steady state, and does a complete control analysis. <http://pysces.sourceforge.net/>
 - (26) ScrumPy: A metabolic modeling package with a mixture of GUI and command-line tools. <http://mudshark.brookes.ac.uk/ScrumPy/>
 - (27) Virtual Cell: A remote user modeling and simulation environment where users can create biological models of various types (Loew and Schaff, 2001). <http://www.nrcam.uchc.edu/>

Pathway analysis tool

- (28) BioMX: The pathways analysis tool that generates and presents metabolic pathways according to user-defined guidelines and multiple layout strategies. <http://www.biomax.de/>
- (29) CADLIVE (Computer-Aided Design of LIVing systEMs): A system for metabolic and gene regulatory networks using GUI and SBML for store models (Kurata et al., 2003). <http://kurata21.bse.kyutech.ac.jp/cadlive/>
- (30) CellDesigner: A process diagram editor for gene-regulatory and biochemical networks (Funahashi et al., 2003). <http://www.celldesigner.org/>
- (31) Cytoscape: a software environment for integrated models of biomolecular interaction networks (Shannon et al., 2003). <http://www.cytoscape.org/>
- (32) PathwayLab: An in silico biochemical pathway analysis tool, enables pharmaceutical R&D to reach their target decisions faster and with higher accuracy, InNetics AB, Sweden. <http://innetics.com/>
- (33) PaVESy: Pathway visualization and editing system (Ludemann et al., 2004). <http://pavesy.mpimp-golm.mpg.de/>

Stochastic simulation

- (34) Simulac Arkin, A.P., Simulac (General purpose software for stochastic simulation of simple prokaryotic genetic and biochemical systems) and Decuce (Software for deduction of chemical network structure from measured time-series). <http://gobi.lbl.gov/~aparkin/Stuff/Software.html>
 - (35) STOCHSIM: A stochastic simulator of (bio)chemical reactions (Le Novere and Shimizu, 2001). <http://www.ebi.ac.uk/~lenov/stochsim.html>
-

integrated body functions, such as the systemic circulation). To take into account the hierarchical network structure of biological functions in software development will therefore promote the scientific understanding of biology, as well as of informatics. In software development, object-oriented concepts, which preserves the analogy between the real world and the structure of the computer program, have become very popular and can easily be applied to compose a hierarchical network model of a dynamic function.

In this study, we have developed a common biological simulation tool called *simBio*, where functional modules can be independently developed and used to compose larger models by applying object-oriented concepts (Sarai et al., 2003a). We implemented cardiac cell models (the Kyoto model, the LRd model from Faber and Rudy, 2000, and the Noble 98 model from Noble et al., 1998), an epithelial cell model (Strieter et al., 1990) and a pancreatic β cell model (Magnus and Keizer, 1998) into *simBio*. We have designed *simBio* to facilitate the development of detailed models composed of functional modules, and the sharing of models in its entirety or in parts. Thus, the development of a large variety of biological models will be enhanced.

2. Methods

2.1. Define Nodes and Reactors

Synchronized electrical activity is one of the key aspects of cardiac cell function. The minimum requirement for generating an action potential is the presence of both Na^+ and K^+ channels, which allow mainly the permeation of Na^+ and K^+ , respectively (Fig. 1). Here, the functional molecules are the channel proteins, which span the lipid bilayer of the cell membrane and provide selective pathways across the membrane for specific ions, according to their membrane permeability and electro-chemical gradients. Ion channels show conformational transitions between open and closed states (channel gating) responding to either changes in membrane potential (V_m) and/or specific ligand binding. The function of ion channels can be mathematically represented as follows:

$$P_{\text{open}} = f_A(V_m)f_B([\text{Ligand}]), \quad (1)$$

$$V_X = (RT/z_X/F) \ln([X]_o/[X]_i), \quad (2)$$

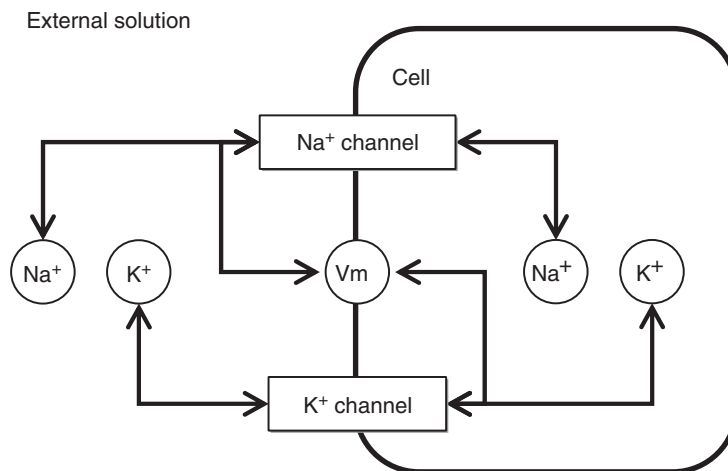


Fig. 1. The structure of the minimum excitable cell model as implemented in *simBio*. The “External solution”, “Cell”, “ Na^+ channel” and “ K^+ channel” are *Reactors* (squares), while the membrane potential (V_m), Na^+ concentrations (Na^+) and K^+ concentrations (K^+) are *Nodes* (circles). The arrows indicate an interaction between functional modules and variables. See text for the definition of *Reactors* and *Nodes*.

$$I = g(V_m - V_X)P_{\text{open}}, \quad (3)$$

$$dV_m/dt = -I/C_m, \quad (4)$$

$$d[X]_i/dt = -I/v_i/z_X/F, \quad (5)$$

$$d[X]_o/dt = I/v_o/z_X/F. \quad (6)$$

At first, the ion channel model determines the probability of the open conformation (P_{open}) using the channel gating equations, which are determined from experiments: f_A is a function of V_m and f_B a function of the ligand concentration (Eq. (1)). Users can define the equation freely or based on the typical Michaelis–Menten kinetics. The reversal potential of ion X (V_X) is calculated from the extra- and intracellular-ion concentrations ($[X]_o$, $[X]_i$) (Eq. (2)), where R is the gas constant, T the absolute temperature, z_X the valence of ion X , and F the Faraday constant. The ion current I is calculated in Eq. (3), where g is the total conductance when all the channels are open. The changes of V_m , $[X]_o$ and $[X]_i$ with time are integrated using Eqs. (4)–(6), where C_m is the membrane capacitance, v_i the cell volume, and v_o the volume of extracellular space.

The time-dependent variables, such as V_m , $[X]_o$ and $[X]_i$, are defined as *Nodes*. Functional modules, such as ion channels, are defined as *Reactors* and consist of several empirical and/or theoretical equations. In Fig. 1, “External solution”, “Cell”, “Na⁺ channel”, and “K⁺ channel” are *Reactors*, while “ V_m ”, “Na⁺” and “K⁺” are *Nodes*. Each arrow pointing from a *Node* to a *Reactor* indicates that the value of this *Node* is fed into the *Reactor*, while an arrow pointing in the other direction indicates that the result of the calculation is integrated into the connected *Node*. The reactions of other molecules such as enzymes can be calculated using the same logic. In summary, the cell function can be reduced to *Nodes* (variables) and *Reactors* (a set of equations) that calculate the time-dependent variation of the *Nodes*.

2.2. Reactor as a functional module

The discovery of new functional molecules and elucidation of molecular functions by experimental biological studies will steadily continue. If these new findings could be easily described and incorporated into an existing object-oriented model, the development of a comprehensive cell model would be speeded up. In the present study, *Reactors* are written in Java (Sun Microsystems Inc., CA, USA, freely available from <http://www.sun.com/>), a well known object-oriented programming language. Java is a cross-platform language, which means that any program written in that language can be executed on a personal computer or a supercomputer, independent of the operating system that it runs, e.g. Microsoft Windows (Microsoft, CA, USA), Mac OS X (Apple, CA, USA), UNIX, Linux. Every *Reactor* is implemented as an individual Java class, which is an individual unit of the object-oriented program, making it easy to create and improve separately, based on the biological hierarchy, e.g. *Reactors* such as the “Na⁺ channel” and “K⁺ channel” classes belong to the “Channel” class. The channel class inherits from the “Membrane Transporter” class, which itself inherits from the “Molecule” class, which in turn inherits from the “Reactor” class (Fig. 2). The “Molecule” and the “Structure” classes are prepared in order to represent biological concepts. The “Membrane Transporter” class adds the ion flux to V_m and the ion concentrations (Eqs. (4)–(6)). The “Channel” class calculates the ion

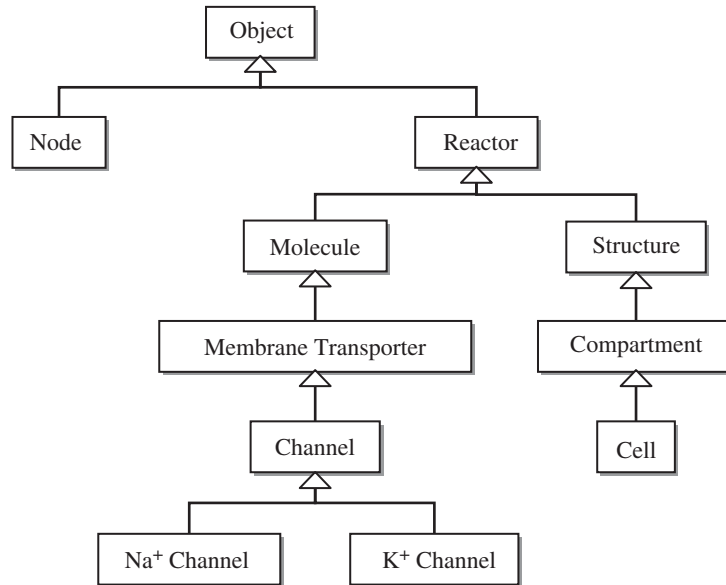


Fig. 2. The classification diagram of the functional modules shown in Fig. 1. Arrows indicate inheritance.

flux (Eqs. (2) and (3)). The “Na⁺ channel” and “K⁺ channel” classes determine the open probability (Eq. (1)).

A cell is a compartment enclosed in a lipid bilayer, which keeps constitutive substances inside, and localizes functional molecules, i.e. ion channels are distributed on the cell membrane and enzymes are anchored on the internal membrane or kept in the cytosol, etc. In *simBio*, this compartmentalization is represented by a class tree (Fig. 2), where the “Cell” class inherits from the “Compartment” class, which itself from the “Structure” class, which in turn inherits from the “Reactor” class. Since the *Reactor* classification bears analogy to biological concepts, collaboration and sharing of knowledge between biologists and information scientists can be advanced.

2.3. Composition of the cell model

A biological system shows a complex hierarchy, for example a cell shows different levels, such as the extracellular space, the cell membrane, the cytosol, the organelles, and molecules. This hierarchy should be represented in the cell model. In *simBio*, the eXtensible Markup Language (XML, <http://www.w3.org/XML/>), the universal format for structured documents and data on the web, administered by the World Wide Web consortium (<http://www.w3.org/>), is used to describe the model structure consisting of *Reactors* and *Nodes*. For example, the XML description file, which corresponds to the action potential generation model in Fig. 1, is shown in Fig. 3. The description file is tree structured and starts with a base element defined by a model tag (the first and last lines in Fig. 3), which contains all *Reactors* and *Nodes* defined by a specific reactor tag and node tag. Each element is characterized by attributes such as name, initial value and units. The units can be freely defined by users and should be kept consistent in a model. In the future,

```

<model name="Model">
  <compartment name="External solution" className="structure.Compartment" >
    <node name="Na" value="140.0" units="millimolar" />
    <node name="K" value="5.4" units="millimolar" />
    <cell name="Cell" className="structure.compartment.Cell">
      <node name="membrane capacitance" value="132.0" units="pF"/>
      <node name="Vm" value="-86.04" units="millivolt"/>
      <node name="Na" value="4.897" units="millimolar"/>
      <node name="K" value="141.0" units="millimolar"/>
      <INa name="Na channel" units="pA" className="molecule.transporter.channel.Na">
        <link name="Vm" value="../Vm" units="millivolt"/>
        <link name="Nai" value="../Na" units="millimolar"/>
        <link name="Nao" value="../Na" units="millimolar"/>
        <node name="Na permeability" value="2860" units="pA/mM"/>
        <node name="Open probability" value="0.5" units="dimensionless"/>
      </INa>
      <IK name="K channel" units="pA" className="molecule.transporter.channel.K">
        <link name="Vm" value="../Vm" units="millivolt"/>
        <link name="Ki" value="../K" units="millimolar"/>
        <link name="Ko" value="../K" units="millimolar"/>
        <node name="K permeability" value="151" units="pA/mM"/>
        <node name="Open probability" value="0.5" units="dimensionless"/>
      </IK>
    </cell>
  </compartment>
</model>

```

Fig. 3. The structure of the cell model shown in Fig. 1 described in XML format. The model is defined using the *Reactors* shown in Fig. 2.

however, we may use a predefined set of units compatible with CellML. Each *Reactor* element is assigned to a java class via the class name attribute. XML elements for each *Reactor* are created by *simBio* to compose a model. The java class contains a given set of equations and calculates the dynamic changes of the *Nodes*. The “External solution” element (second and second to last line in Fig. 3) is an object of the “Compartment” class, which inherits from the “Reactor” class, and contains the “Na” and “K” *Nodes* (third and fourth lines in Fig. 3, respectively) and the “Cell” *Reactor* (fifth line and third to last line in Fig. 3). The “Cell” *Reactor* contains “ V_m ”, “Na” and “K” *Nodes* and “Na channel” and “K channel” *Reactors*. The Na channel and K channel *Reactors* calculate the amount of a specific ion crossing the membrane using the permeability and open probability of each channel. When a *Reactor* has a *Node*, it is defined by the “node” tag (e.g. the third line in Fig. 3, which refers to the “Na” *Node* defined in the “External solution” *Reactor*). When a *Reactor* uses a *Node* that is owned by another *Reactor*, it is defined by the “link” tag, according to its biological bases (e.g. the eleventh line in Fig. 3, which refers to the “ V_m ” *Node* that is originally defined in the “Cell” *Reactor*).

2.4. Simulation system

The *simBio* system is composed of five components (Fig. 4);

1. The *Reactor* database, which contains the java classes of functional modules.
2. The structure database, which contains the structures and parameters of biological models written in XML.
3. The instantiation component, where the desired model is instantiated using the appropriate *Reactors*, from the *Reactor* database using the aforementioned XML file, and *Nodes*.

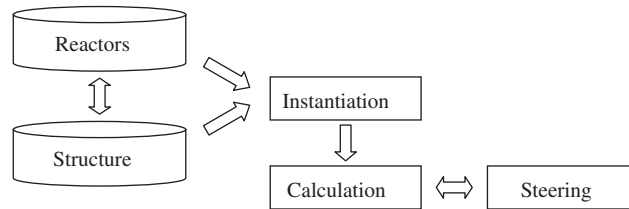


Fig. 4. Diagram for *simBio*. Cylinders indicate database, and arrows the transmission of information.

4. In the calculation component, the dynamics of biological models is computed by *Reactors*, i.e. ordinary differential equations (ODEs) are solved.
5. The steering component provides interfaces to obtain and visualize the results, as well as to manipulate the parameters.

Any computer, which has Java 1.4 or later installed, can execute *simBio*. The *simBio* system reads a model description file (such as the one in Fig. 3), alert users to invalidity of an XML file, creates *Reactor* and *Node* instances, and connects everything together using the link tags. The graph structure of the model instance shown in Fig. 5 is created after the model description file in Fig. 3 is executed. Initial values of *Nodes* such as ion concentrations are simultaneously read from the description file. The *Reactors* are added to a calculation list. *simBio* integrates the ODEs using either an Euler or fourth-order Runge-Kutta method.

The core interface described using the unified modeling language (UML) is shown in Fig. 6. The minimum interface is defined in order to instantiate a graph-structured model composed of both biological functions and analyzing modules. The composite pattern and the visitor pattern are used as described in design pattern theory, which is commonly used in object-oriented programming. The analyzing modules inherit from the “Analyzer” class, and are defined in the model description file. In this way, it is possible to change interfaces such as the graphical user interfaces (GUI) or the output format of text files by editing the description file. The *simBio* system is written as a Java package using 400 lines (11.6 KB) for the instantiation and 1030 lines (25.2 KB) for the calculation and steering components.

3. Results

The cardiac ventricular cell model (Matsuoka et al., 2003) is composed of the following *Reactors*: the external solution, the cell, 18 types of ionic currents across the cell membrane, two compartments of the sarcoplasmic reticulum (SR), four types of ionic currents across the SR membrane, two types of Ca^{2+} buffers, an ATP producing system, the contraction model proposed by Negroni and Lascano (NL model, 1996), and the external stimulus current (“Pipette”). The Na^+ , K^+ , Ca^{2+} , ATP concentrations and state variables of the channels are implemented as *Nodes* in that same model. Their interactions are represented in the graph shown in Fig. 7A. In a similar way, the interactions of *Reactors* and *Nodes* representing the LRd model are shown in Fig. 7B. Integration of molecular functions within the cell function is achieved by

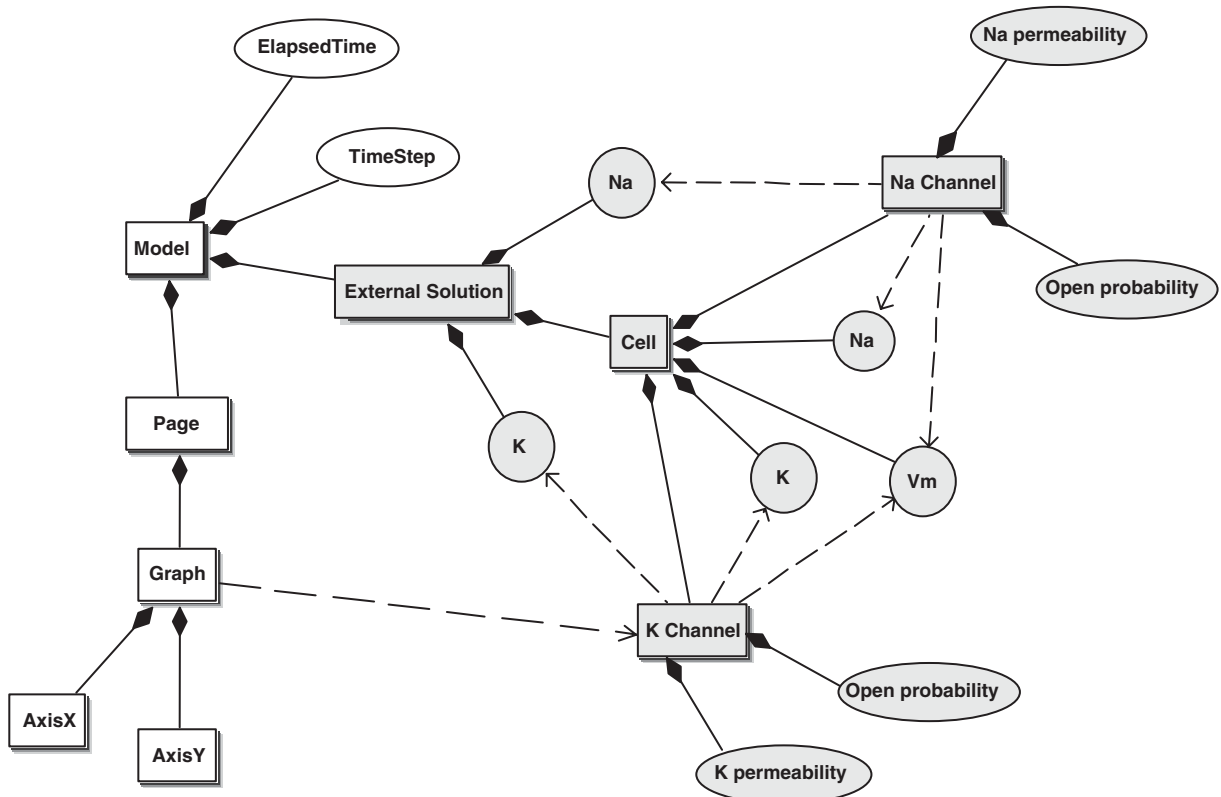


Fig. 5. Graph of *Reactors* and *Nodes* used in the excitation model shown in Figs. 1 and 3. The closed squares are *Reactors*, and circles are *Nodes*. The “Model” open square represents the ODE integrator, while the other open squares are *analyzers*. Filled diamonds indicate composition, while dashed arrows reference.

sharing common *Nodes* among several *Reactors*. Though simpler than the Kyoto model, the LRd model has been created using the same basic structure, which consists of the external solution, the Cell, ionic currents across the cell membrane, two compartments of SR, four types of ionic currents across the SR membrane, and three types of Ca^{2+} buffers. The NL model can be incorporated into the LRd model instead of Troponin by just amending the corresponding LRd XML file. The GUI provides access to all the parameters and to real-time visualization of the simulation, as shown in Fig. 8. Because the set of equations for our ventricular (Matsuoka et al., 2003) and sino-atrial node (Sarai et al., 2003b) cell models are the same, we could incorporate the latter into *simBio* by simply modifying the XML description file. The Noble 98 model (Noble et al., 1998), an epithelial cell model (Strieter et al., 1990) and a pancreatic β cell model (Magnus and Keizer, 1998) have also been incorporated into *simBio* (not shown).

All functional modules are implemented as individual java classes. The ATP producing system is modeled in a lumped equation, i.e. the rate of the ATP synthesis simply depends on the ADP concentration. The complex class in Fig. 9A, which inherits from the reactor class, was introduced to handle the complicated reaction system. The NL model was implemented using two classes: the

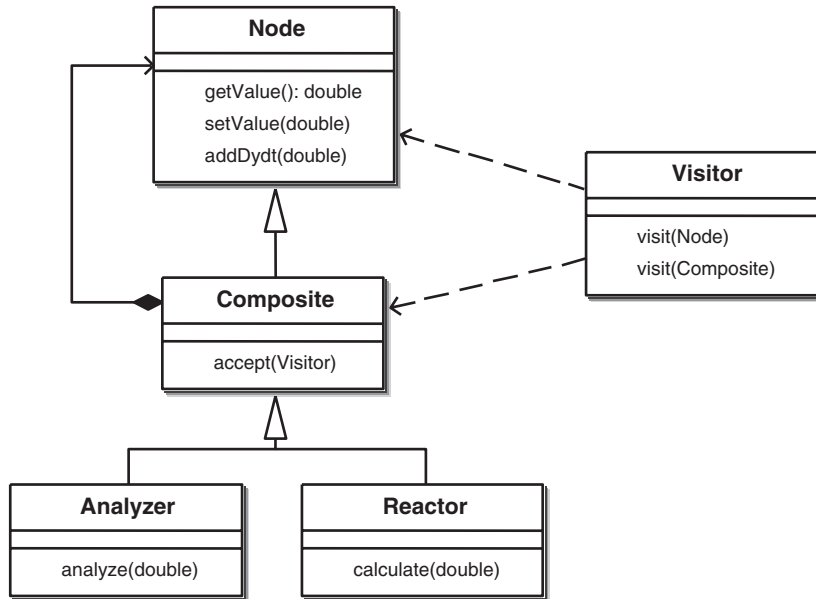


Fig. 6. The core interfaces written in unified modeling language (a standardized modeling language for designing and documenting computer programs). Open boxes indicate a class, which contains a class name (first line) and methods (following lines). Open triangle arrows indicate inheritance. Filled diamonds indicate composition, while dashed arrows reference.

“Troponin” class, which inherits from the “Ca²⁺ buffer” class, and the “Cross Bridge” class, which inherits from the “Enzyme” class. The “Ca²⁺ buffer” and “Enzyme” classes are both subclasses of the “Molecule” class. The hierarchical structure of all *Reactors* shown in Fig. 9 is in agreement with the biological classification. For example, the ATP-sensitive K⁺ channel (KATP) is an extracellular K⁺ dependent, and a pure K⁺ channel, and is classified as a membrane transporter. The KATP current (I_{KATP}) is described by Eqs. (7)–(11), which correspond to Eqs. (1)–(5), respectively. The channel gating depends only on the intracellular ATP concentration (Eq. (7)), and not on V_m as in Eq. (1).

$$P_{\text{open}} = 0.8 / (1.0 + 100.0[\text{ATP}]_i^2), \quad (7)$$

$$V_K = (RT/F) \ln ([K^+]_o / [K^+]_i), \quad (8)$$

$$I_{\text{KATP}} = 5.50588 [K^+]_o^{0.24} (V_m - V_K) P_{\text{open}}, \quad (9)$$

$$dV_m/dt = -I_{\text{KATP}}/C_m, \quad (10)$$

$$d[K^+]_i/dt = -I_{\text{KATP}}/v_i/F. \quad (11)$$

These equations are computed in different Java classes: the “KATP” class calculates P_{open} (Eq. (7)), the “Membrane” class the reversal potential of K⁺ (Eq. (8)), the “Pure K” class I_{KATP} (Eq. (9)), and the “Membrane Transporter” class dV_m/dt and $d[K^+]_i/dt$ (Eqs. (10) and (11)).

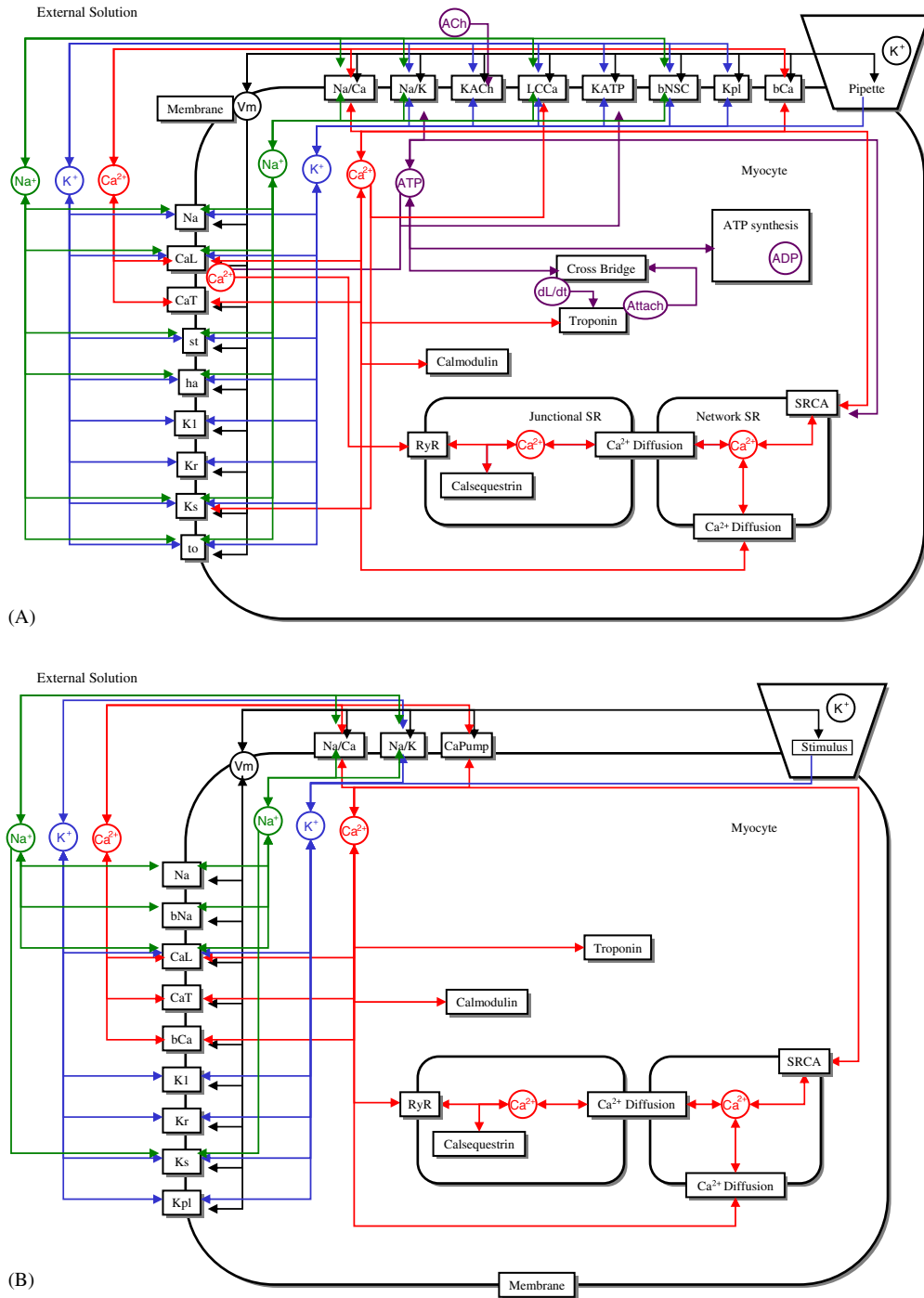


Fig. 7. Graph of the *Reactors* and *Nodes* used in a couple of cardiac cell models. A: the excitation-contraction coupling model proposed by Matsuoaka et al. (2003). B: the excitation model proposed by Faber and Rudy (2000).

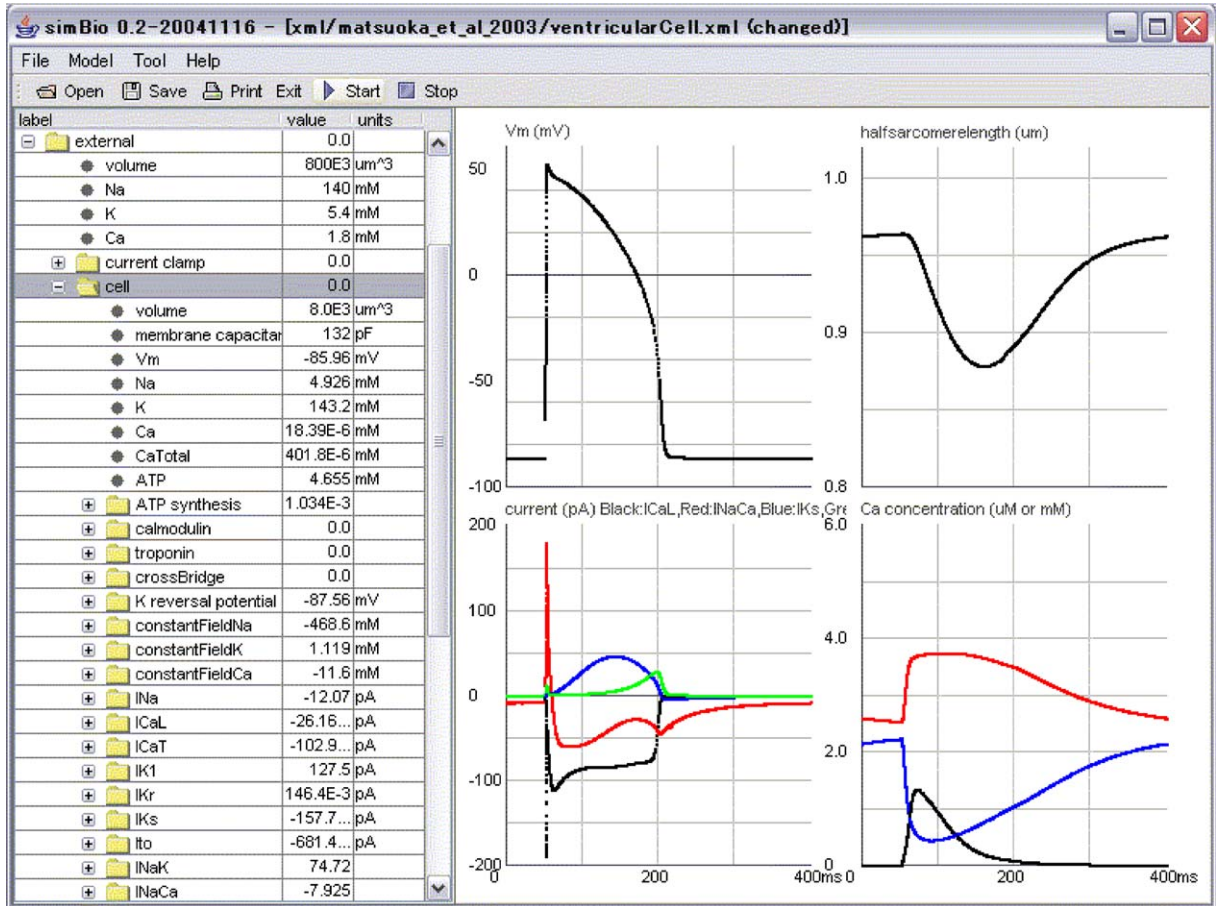


Fig. 8. The GUI provides access to all the parameters and charts of the ventricular cell model simulation. The upper left graphical panel shows time-dependent changes of V_m (mV), and the lower left graph, four types of ionic currents (pA): the L-type Ca^{2+} current in black, the $\text{Na}^+/\text{Ca}^{2+}$ exchanger current in red, the slow component of the delayed rectifier K^+ current in blue and its rapid component in green. The upper right graph depicts the half-sarcomere length, and the lower right graph the cytosolic Ca^{2+} concentration in black (μM), the Ca^{2+} uptake site of the SR in red (mM), and the Ca^{2+} release site in blue (mM).

Eqs. (10) and (11) only describe I_{KATP} 's contribution to dV_m/dt and $d[\text{K}^+]_i/dt$. Total changes to V_m and ion concentrations are the sum of all the membrane currents, including the I_{KATP} contribution. In the present model, $[\text{K}^+]_o$ was assumed to be constant, because of v_i being very small compared to v_o (Eq. (6)).

Matsuoka et al. (2004b) further expanded the ventricular cell model by incorporating pivotal reactions of the ATP metabolism employing the mitochondrial oxidative phosphorylation model of Korzeniewski and Zoladz (2001). The excitation–contraction–metabolism coupling model was created through the replacement of the lumped ATP synthesis model by detailed classes that are based on the molecular reactions shown in Fig. 9B. The models of the mitochondrial Ca^{2+}

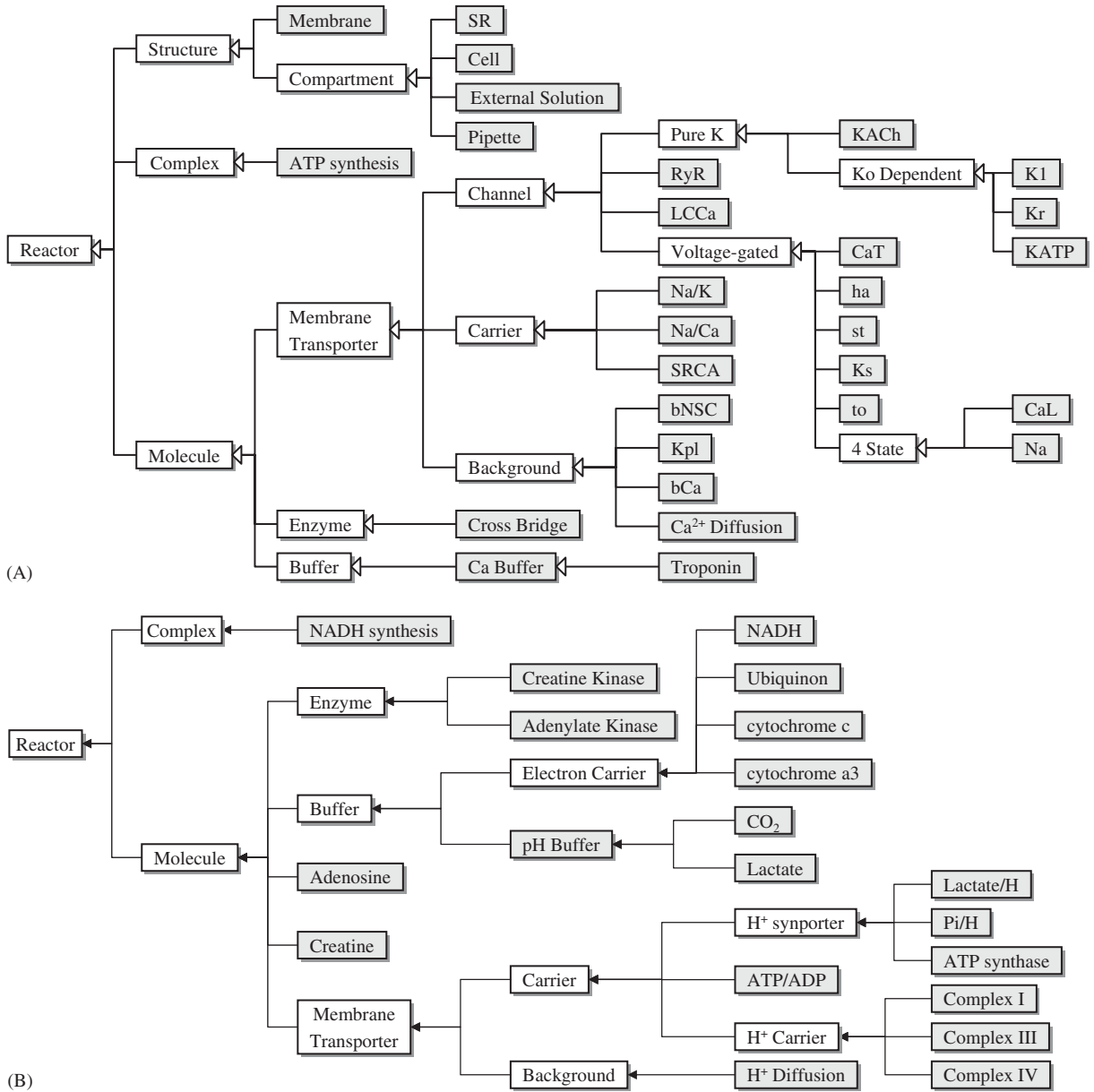


Fig. 9. Classification diagram of the *Reactors* used in the Kyoto models. Arrows indicate inheritance, and open squares abstract classes (i.e. an object for those classes cannot be created, unlike for the grey classes). A: the *Reactors* used in the excitation–contraction coupling model (Matsuoka et al., 2003) shown in Figs. 7A and 8. B: the *Reactors* included in the ATP metabolism model (Matsuoka et al., 2004b).

uniporter and of the $\text{Na}^+/\text{Ca}^{2+}$ exchanger of the pancreatic β cell have been incorporated into our ventricular cell model to simulate mitochondrial Ca^{2+} concentration (Matsuoka et al. 2004a).

4. Discussion

4.1. Motivation

An important part of modeling is spent researching quantitative data from the literature (Takahashi et al., 2002). In addition, it is often difficult for biologists to formulate empirical equations and convert them into a computer-readable language. Furthermore, an existing model must be continuously revised to simulate new experimental findings. Therefore, an easy-to-use model-construction tool is needed, which biologists can use for a fast but comprehensive model development. Several complementary tools are being developed for the modeling of biological systems, as shown in Table 1. However, we need a tool for the simple conversion of a biological system into executable code, for easy model expansion, and for a re-use of models as a whole or in part as sub-models in different contexts. In this study, we have presented a new object-oriented tool called *simBio*, which was developed *de novo* using Java and XML.

The C++ implementation of our ventricular cell model (available from <http://www.card.med.kyoto-u.ac.jp/> and compiled with Microsoft Visual C++ 6.0) proved to be the quickest. *simBio* took 1.6 times longer to compute the exact same model, but we believe this is compensated for by the fact that the code can be executed on a variety of systems, from a PC to a workstation, or even a supercomputer. In every tool cited in Table 1, to implement new equations or equations found in the literature the modeler must use either a metalanguage (e.g. CellML) or a programming language (e.g. Fortran). In this respect, *simBio* with its use of Java is no different.

4.2. Model expansion

In the field of systems biology, attempts have been made to describe biological function as a system based on information of all the molecules involved, which was acquired from genomic and proteomic analysis. However, quantitative and functional aspects of individual proteins still remain to be elucidated, and there is an important variety of molecular species. It is, therefore, difficult to develop a comprehensive model based only on individual molecules, and simply not feasible with the current computing power. It is essential to develop a model that is as detailed as necessary to clarify the quantitative mechanisms underlying a given experimental observation. For those functional modules, in which the proteins involved and underlying interactions have not yet been clarified, we define temporarily an input-output relationship referring to experimental data of various levels of detail. Whenever details of the molecular mechanisms become available, the simple input-output relationship can be revised. For example, the ATP producing system in our original cell model (Matsuoka et al., 2003) is given by a single equation, as a function of ADP concentration. This simple model is useful to examine the effects of ATP production inhibition on cellular function. However, for analyzing the effects of oxygen and nutrient depletion, a more detailed description including the molecular steps of oxidative phosphorylation is necessary (Matsuoka et al., 2004b).

4.3. Model sharing

Cardiac cell models are not only applied to describe or validate a possible physiological mechanism, but also to assess a drug's efficiency and side effects. Biological models can also be

used to select new targets for drug discovery. Moreover, to include patient data into the model will result in a personalized medical consultation (Michelson, 2003). Applications of those biological models will be facilitated through their exchange, via a computer-readable publication. Although most researchers develop a model using variable tools familiar to them, or even by creating their own model specific tools, the process of developing a biological model can be divided into nine common steps, which are:

1. Define the variables,
2. Set the initial values,
3. Define the equations,
4. Integrate the ODEs,
5. Output the time series data,
6. View the result charts,
7. Optimize the parameters to reconstruct a given phenomenon,
8. Analyze the model behaviors,
9. Save and load the settings.

CellML (Table 1, entry #4) and SBML (Table 1, entry #5) provide information necessary for step 1 to 3, which are the minimum requirement for model sharing. To share all processes of modeling and simulation, SBW (Table 1, entry #8), BioSPICE (Table 1, entry #6), or BioUML (Table 1, entry #7) offers a framework for the communication between visual modeling and/or simulation tools. To provide free access to the source code of mathematical models is an alternative to model sharing, and both the LRD and Kyoto models (<http://www.card.med.kyoto-u.ac.jp/>) are available on the Web. In *simBio*, both Java and XML are used, but a set of equations written in Java can be converted to and from the XML format. For example, COR developed by Dr. A. Garny (Table 1, entry #10) can convert a model described in CellML to a Java and XML files that are directly useable by *simBio*. Furthermore, the simple *simBio* interface shown in Fig. 6 allows every step from one to nine to be written in an individual Java class as well as to be executed by other connected tools. Our aim is not to hide the process of computing, but to avoid people from bothering with a programming common process in modeling. Our aim is also to share the whole modeling process and to make models more readable and understandable, so that the models can be widely executed and evaluated.

The *simBio* package is freely available from <http://www.sim-bio.org/>.

Please see also related communications in this volume by Youm et al. (2005), Nickerson and Hunter (2005). For downloadable content please see <http://www.physiome.org.nz/publications/PBMB-2005-89/Sarai/>

Acknowledgements

The authors thank Drs. A. Garny, A. Amano and Mr. T. Shimayoshi for valuable discussion, and Drs. T. Powell and N. Schneider for carefully reading the manuscript. This study was supported by the program for promotion of fundamental studies in health sciences of the organization for pharmaceutical safety and research, a grant-in-aid for scientific research, and the

leading project for biosimulation from the Japanese Ministry of Education, culture, sports, science and technology.

References

- Allen, N.A., Shaffer, C.A., Vass, M.T., Ramakrishnan, N., Watson, L.T., 2003. Improving the development process for eukaryotic cell cycle models with a modeling support environment. *Simulation* 79 (12), 674–688.
- Demir, E., Babur, O., Dogrusoz, U., Gursoy, A., Nisanci, G., Cetin-Atalay, R., Ozturk, M., 2002. PATIKA: an integrated visual environment for collaborative construction and analysis of cellular pathways. *Bioinformatics* 18, 996–1003.
- Faber, G.M., Rudy, Y., 2000. Action potential and contractility changes in $[Na^+]_i$ overloaded cardiac myocytes: a simulation study. *Biophys. J.* 78, 2392–2404.
- Finney, A., Hucka, M., 2003. Systems biology markup language: level 2 and beyond. *Biochem. Soc. Trans.* 31, 1472–1473.
- Funahashi, A., Tanimura, N., Morohashi, M., Kitano, H., 2003. CellDesigner: a process diagram editor for gene-regulatory and biochemical networks. *BIOSILICO* 1, 159–162.
- Garny, A., Kohl, P., Noble, D., 2003. Cellular open resource (COR): a public CellML based environment for modelling biological function. *Int. J. Bifurcat. Chaos* 13, 3579–3590.
- Garvey, T.D., Lincoln, P., Pedersen, C.J., Martin, D., Johnson, M., 2003. BioSPICE: access to the most current computational tools for biologists. *OMICS* 7, 411–420.
- Goryanin, I., Hodgman, T.C., Selkov, E., 1999. Mathematical simulation and analysis of cellular metabolism and regulation. *Bioinformatics* 15, 749–758.
- Kanehisa, M., Bork, P., 2003. Bioinformatics in the post-sequence era. *Nat. Genet. (Suppl)*, 305–310.
- Kitano, H., 2002. Computational systems biology. *Nature* 420, 206–210.
- Kolpakov, F.A., 2002. Biouml—framework for visual modeling and simulation biological systems. *Proceedings of the International Conference on Bioinformatics of Genome Regulation and Structure, 2002*
- Korzeniewski, B., Zoladz, J.A., 2001. A model of oxidative phosphorylation in mammalian skeletal muscle. *Biophys. Chem.* 92, 17–34.
- Kurata, H., Matoba, N., Shimizu, N., 2003. CADLIVE for constructing a large-scale biochemical network based on a simulation-directed notation and its application to yeast cell cycle. *Nucleic Acids Res.* 31–14, 4071–4084.
- Le Novere, N., Shimizu, T.S., 2001. STOCHSIM: modelling of stochastic biomolecular processes. *Bioinformatics* 17, 575–576.
- Lloyd, C.M., Halstead, M.D., Nielsen, P.F., 2004. CellML: its future, present and past. *Prog. Biophys. Mol. Biol.* 85, 433–450.
- Loew, L.M., Schaff, J.C., 2001. The Virtual Cell: a software environment for computational cell biology. *Trends Biotechnol.* 19, 401–406.
- Ludemann, A., Weicht, D., Selbig, J., Kopka, J., 2004. PaVESy: pathway visualization and editing system. *Bioinformatics* 20, 2841–2844.
- Magnus, G., Keizer, J., 1998. Model of beta-cell mitochondrial calcium handling and electrical activity. I. Cytoplasmic variables. *Am J Physiol* 274, C1158–C1173.
- Matsuno, H., Tanaka, Y., Aoshima, H., Doi, A., Matsui, M., Miyano, S., 2003. Biopathways representation and simulation on hybrid functional Petri net. *In Silico Biol.* 3–3, 389–404.
- Matsuoka, S., Sarai, N., Kuratomi, S., Ono, K., Noma, A., 2003. Role of individual ionic current systems in ventricular cells hypothesized by a model study. *Jpn. J. Physiol.* 53, 105–123.
- Matsuoka, S., Jo, H., Sarai, N., Noma, A., 2004a. An in silico study of energy metabolism in cardiac excitation–contraction coupling. *Jpn. J. Physiol.* 54, 517–522.
- Matsuoka, S., Sarai, N., Jo, H., Noma, A., 2004b. Simulation of ATP metabolism in cardiac excitation–contraction coupling. *Prog. Biophys. Mol. Biol.* 85, 279–299.
- Mendes, P., Sha, W., Ye, K., 2003. Artificial gene networks for objective comparison of analysis algorithms. *Bioinformatics* 19 (Suppl. 2), 122–129.

- Michelson, S., 2003. Assessing the impact of predictive biosimulation on drug discovery and development. *J Bioinform Comput Biol* 1, 169–177.
- Negróni, J.A., Lascano, E.C., 1996. A cardiac muscle model relating sarcomere dynamics to calcium kinetics. *J. Mol. Cell. Cardiol.* 28, 915–929.
- Nickerson, D.P., Hunter, P.J., 2005. The noble cardiac ventricular electrophysiology models in CellML. *Prog. Biophys. Mol. Biol.* 89.
- Noble, D., 2002. Modeling the heart—from genes to cells to the whole organ. *Science* 295, 1678–1682.
- Noble, D., Varghese, A., Kohl, P., Noble, P., 1998. Improved guinea-pig ventricular cell model incorporating a diadic space, IKr and IKs, and length- and tension-dependent processes. *Can. J. Cardiol.* 14, 123–134.
- Puglisi, J.L., Bers, D.M., 2001. LabHEART: an interactive computer model of rabbit ventricular myocyte ion channels and Ca transport. *Am. J. Physiol. Cell Physiol.* 281, C2049–C2060.
- Sarai, N., Amano, A., Matsuoka, S., Matsuda, T., Noma, A., 2003a. Development of the Cardiac Cell Model by Applying Object-Oriented Methods. *IEEE EMBC*, 2702–2705.
- Sarai, N., Matsuoka, S., Kuratomi, S., Ono, K., Noma, A., 2003b. Role of individual ionic current systems in the SA node hypothesized by a model study. *Jpn. J. Physiol.* 53, 125–134.
- Sauro, H.M., Hucka, M., Finney, A., Wellock, C., Bolouri, H., Doyle, J., Kitano, H., 2003. Next generation simulation tools: the systems biology workbench and BioSPICE integration. *OMICS* 7, 355–372.
- Shannon, P., Markiel, A., Ozier, O., Baliga, N.S., Wang, J.T., Ramage, D., Amin, N., Schwikowski, B., Ideker, T., 2003. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Res.* 13, 2498–2504.
- Shapiro, B.E., Levchenko, A., Meyerowitz, E.M., Wold, B.J., Mjolsness, E.D., 2003. Cellerator: extending a computer algebra system to include biochemical arrows for signal transduction simulations. *Bioinformatics* 19, 677–678.
- Strieter, J., Stephenson, J.L., Palmer, L.G., Weinstein, A.M., 1990. Volume-activated chloride permeability can mediate cell volume regulation in a mathematical model of a tight epithelium. *J. Gen. Physiol.* 96, 319–344.
- Takahashi, K., Yugi, K., Hashimoto, K., Yamada, Y., Pickett, C., Tomita, M., 2002. Computational challenges in cell simulation. *IEEE Intelligent Syst.* 17, 64–71.
- Tomita, M., Hashimoto, K., Takahashi, K., Shimizu, T.S., Matsuzaki, Y., Miyoshi, F., Saito, K., Tanida, S., Yugi, K., Venter, J.C., Hutchison III, C.A., 1999. E-CELL: software environment for whole-cell simulation. *Bioinformatics* 15, 72–84.
- Youm, J.B., Han, J., Kim, N., et al., 2005. Role of stretch-activated channels on the stretch-induced changes of rat atrial myocytes. *Prog. Biophys. Mol. Biol.* 89.