Pacific Symposium on Biocomputing 4:228-239 (1999)

The Virtual Cell

J. SCHAFF and L. M. LOEW
Center for Biomedical Imaging Technology
Department of Physiology
University of Connecticut Health Center
Farmington, CT 06030, USA

This paper describes a computational framework for cell biological modeling and simulation that is based on the mapping of experimental biochemical and electrophysiological data onto experimental images. The framework is designed to enable the construction of complex general models that encompass the general class of problems coupling reaction and diffusion.

1 Introduction

A general computational framework for modeling cell biological processes, the "Virtual Cell", is being developed at the University of Connecticut Health Center. The Virtual Cell is intended to be a tool for experimentalists. Models are constructed from biochemical and electrophysiological data mapped to appropriate subcellular locations in images obtained from a microscope. Chemical kinetics, membrane fluxes, and diffusion are thus coupled and the resultant equations are solved numerically. The results are again mapped to experimental images so that the cell biologist can fully utilize the familiar arsenal of image processing tools to analyze the simulations.

The philosophy driving the "Virtual Cell" project requires a clear operational definition of the term "model". The idea is best understood as a restatement of the scientific method. A model, in this language, is simply a collection of hypotheses and facts that are brought together in an attempt to understand a phenomenon. The choice of which hypotheses and facts to collect and the manner in which they are assembled, themselves constitute additional hypotheses. A prediction based on the model is, in one sense, most useful if it doesn't match the experimental details of the process - it then unequivocally tells us that the elements of the model are inaccurate or incomplete. Although such negative results are not always publishable, they are a tremendous aid in refining our understanding. If the prediction does match the experiment, it *never* can guarantee the truth of the model, but should suggest other experiments that can test the validity of critical elements; ideally, it should also provide new predictions that can, in turn, be verified experimentally. The Virtual Cell is itself not a model. It is intended as a computational framework and tool for cell biologists to create models and to generate predictions from models via simulations. To assure the reliability of such a tool, all the underlying math, physics, and numerics must be thoroughly validated. To assure the utility and accessibility of such a tool to cell biologists, the results of such simulations must be presented in a format that may be analyzed using procedures comparable to those used to analyze the results of experiments.

In this paper we describe the current status of the design considerations for model management and the user interface. Details of the mathematics and numerical methods employed for simulations are not included but can be found in an earlier publication[1] and on our website: http://www2.uchc.edu/htbit/vcell.

2. Model Management

2.1 Background

Often theoreticians develop the simplest model that reproduces the phenomenon under study[2]. These may be quite elegant, but are often not very extensible to other related phenomena. Other modeling efforts characterize single physiological mechanisms[3,4], but these are often developed ad hoc rather than as part of a reusable and consistent framework.

Our approach to modeling concentrates on the mechanisms as well as the phenomena. The goal of this approach is to provide a direct method of evaluating single models of individual mechanisms in the context of several experiments. This approach enables the encapsulation of sufficient complexity that, after independent validation, allows it to be used as a meaningful predictive tool. In order to include sufficient complexity without overwhelming the user, the models are specified in their most natural form. In the case of chemical reactions, the models are represented by their stoichiometry, a series of reactants, products, modifiers (e.g. enzymes) and their kinetics.

One of the obstacles to modeling is the lack of general purpose simulation and analysis tools. Each potential modeler must have resources in software development and numerical methods at his disposal. Each time the model or the computational domain is altered, the program must be changed. And in practice, the modeling of a new phenomena requires a new simulation program to be written. This is a time consuming and error prone exercise, especially when developed without a proper software methodology.

We are developing a general, well tested framework for modeling and simulation for use by the modeling community. The application of the underlying equations to our framework with nearly arbitrary models and geometry is rigorously investigated. The numerical approach is then properly evaluated and tuned for performance. This methodology results in a proper basis for a general purpose framework. Our approach requires no user programming, rather the user specifies models using biologically relevant abstractions such as reactions, compartments, molecular species, and experimental geometry. This allows a very flexible description of the physiological model and arbitrary geometry. The framework accommodates arbitrary geometry and automatically generates code to implement the specified physiological model.

Another problem is the lack of a standard format for expressing those models. Even implementing published models can be a non-trivial exercise. Some of the necessary details required for implementation can be missing or buried in the references. Often the models are obscured by geometrical assumptions used to simplify the problem. A standard modeling format is required to facilitate the evaluation and integration of separate models. This standard format should separately specify physiological models and cellular geometry in an implementation independent way.

We suggest that the abstract physiological models used with the Virtual Cell framework can form the basis of such a standard.

2.2 Current Implementation

The current implementation of the cell model description[1] involves the manipulation of abstract modeling objects that reside in the Modeling Framework as Java objects. These modeling objects can be edited, viewed, stored in a remote database, and analyzed using the WWW-based user interface (see User Interface section). These objects are categorized as Models, Geometry, and Simulation Context objects. This adopts the naming convention used in the current Modeling Framework software.

2.2.1 Models

A Model object represents the physiological model of the cell system under study. Each Model is defined as a collection of Species (e.g. calcium, ATP), Reactions (e.g. enzyme kinetics, receptor binding, membrane fluxes), and Features (e.g. ER, cytosol).

The Feature objects define compartments within the cells that are mutually isolated by membranes. Using this definition, the extracellular region is separated from the cytosol (and hence the ER, nucleus, etc.) by the plasma membrane. These compartments contain Species and a collection of Reactions that describe the biochemical behavior of that compartment. Average geometric information, such as surface to volume ratios for the appropriate Feature complete the necessary information for a single point compartmental simulation.

The Species are objects that identify molecular species, and are classified as being either Membrane Species or Volume Species. The Membrane Species are described by a surface density for each feature that contains a membrane. The Volume Species are described by a concentration and a diffusion constant.

The Reactions are objects that represent complete descriptions of reaction kinetics. Reactions are collections of related Membrane Reaction Steps (e.g. membrane receptor binding), Volume Reaction Steps (e.g. calcium buffering), and Membrane Fluxes (e.g. flux through an ion channel). The fluxes and reaction rates are represented by arbitrary algebraic expressions. These expression objects are capable of basic algebraic simplification, partial differentiation, and binding to the appropriate Parameters and Species, and numeric evaluation.

2.2.2 Geometry

The Geometry objects represent the cellular geometry (based on segmented images) and can be mapped directly to the corresponding cellular features. The geometry can currently be specified as 2-D or 3-D segmented images with the appropriate scaling information to properly define a simulation domain.

2.2.3 Simulation Context

The Simulation Context objects represent the context of a particular simulation as a specific mapping of the Model objects to the Geometry objects. This mapping specifies the Species and Reactions present in each Feature within the corresponding region in the Geometry. With the addition of initial conditions and boundary conditions, and the membrane jump conditions for each Species, a particular simulation is completely specified. The jump conditions represent the trans-membrane fluxes as well as binding reactions with membrane bound species. This context specifies the generation of the ordinary and partial differential equations of the system. These equations are represented symbolically within the Modeling Framework using expression objects. The computational mesh (orthogonal grid) is sampled from the Geometry.

2.2.4 Compartmental Simulations

For simulation of compartmental models (single point approximation), the ODEs (Ordinary Differential Equations) representing the reaction kinetics are generated, and passed to an interpreted ODE solver (within the client applet).

This system of equations is solved using a simple explicit integration scheme (forward difference) which is first order accurate in time. A higher order numerical scheme will be integrated in the future.

2.2.4 Spatial Simulations

For the solution of a complete spatial simulation, the PDEs (Partial Differential Equations) that correspond to diffusive species, and ODEs for non-diffusive species are generated. These equations are sent to the remote Simulation Server where the corresponding C++ code is automatically generated, compiled, and linked with the Simulation Library. The resulting executable is then run and the results are collected and stored on the server. The Simulation Data Server then coordinates client access to the server-side simulation data for display and analysis.

The system of PDEs are mapped to a rectangular grid using a finite difference scheme based on a control volume approach[2]. The nonlinear source terms representing reaction kinetics are evaluated explicitly (forward difference) and the resulting linearized PDE is solved implicitly (backward difference). Those membranes separating spatially resolved compartments are treated as discontinuities in the solution of the reaction-diffusion equations. These discontinuities are defined by flux jump conditions that incorporate trans-membrane fluxes, binding to membrane bound species, and conservation of mass. The boundary conditions are defined in terms of a known flux (Neumann condition) or a known concentration (Dirichlet condition).

3 User Interaction

3.1 Background

Some software applications in neuronal modeling, for example Neuron[6] and GENESIS[7], are available with commercial quality user interfaces. These packages allow specification and optimization of models, control of simulation, and interpretation of results. They both include scripting capabilities as well as graphical user interfaces. This level of user interaction is desired for any general purpose modeling and simulation framework. However, these packages are primarily useful for studies of neuronal signal conduction in systems of one or a network of neurons. These packages take advantage of the symmetry and shape of neurons to simplify the underlying equations and user specified models and reduce the solution to a one-dimensional problem.

There are also mature efforts in Metabolic Pathway modeling such as GEPASI[8]. This package allows a simple and intuitive interface for specifying reaction stoichiometry and kinetics. The kinetics are specified by selecting from a predefined (but extensible) list of kinetic models (such as Michaelis/Menten) describing enzyme mediated production of metabolites. These packages are focused on biochemical pathways where the spatial aspects of the system are ignored. However, for these simplified descriptions, they provide Metabolic Control Analysis (sensitivity analysis tools), structural analysis (mass conservation identification), and a local stability analysis.

In order to provide a simple interface to a general purpose modeling and simulation capability, the problem must be broken up into manageable pieces. In the case of the virtual cell, these pieces consist of abstract physiological models, abstract geometric description, and a simulation description that defines the solution method and the conditions of that particular problem. For such an interface to be consistent

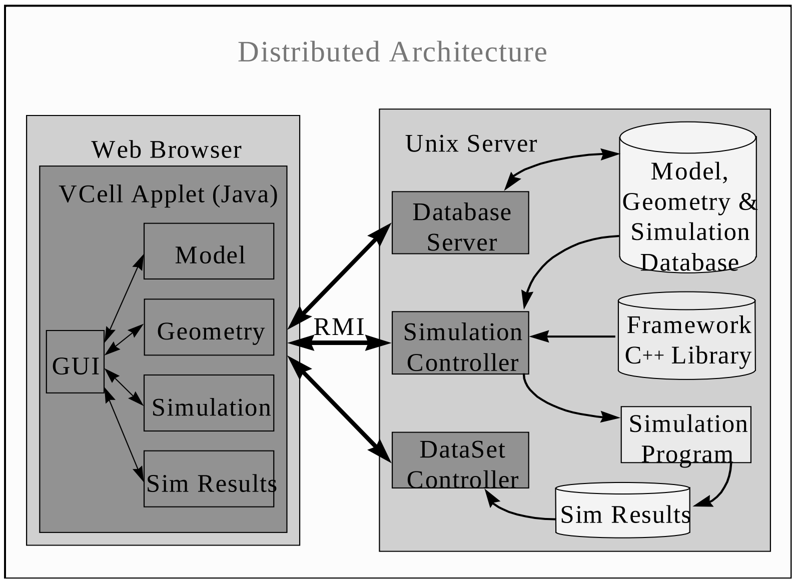


Figure 1: Distributed Architecture for the Virtual Cell Software.

and maintainable, it must map directly to the underlying software architecture.

3.2 Current Implementation

An intuitive user interface is essential to the usability of a complex application. A prototype user interface was developed to provide an early platform for modeling and simulation, and for investigating user interface requirements. The design goal was to capture the minimum functionality required for practical use of the virtual cell.

The current Virtual Cell application (figure 1) utilizes a distributed architecture. The system is decomposed into a modeling application and system services. The modeling application is a WWW accessible Java applet and provides a graphical user interface. The system services are the Database Service, the Simulation Control Service (which encapsulates the Simulation Library), and the Simulation Data Service. The architecture is designed such that the location of the user interface and the corresponding backend services are transparent to the majority of the application. The typical configuration is a Java applet running in a WWW browser, with the Database, Simulation Control, and Simulation Data services executing on a remote machine (WWW server). Alternatively, the software may be executed as a standalone application on a local machine with the requirement that the Java Runtime Environment and a C++ compiler are installed.

3.2.1 Physiological Modeling

The Feature Editor (figure 2) allows the specification of cellular features that represent the concept of compartments within the cell model under study. The basic hierarchical structure of these features determines the case when one or more features are enclosed by another feature. The specification of average spatial properties of an enclosed feature (surface to volume ratio and volume fraction of enclosing feature) provides a complete description of a compartmental model. This is useful in calculating quick approximate solutions.

The Species Editor allows the user to specify the volume and membrane species of a model. The default initial concentration and diffusion rates are specified for volume species within each feature. The initial surface densities of the membrane species are specified for each feature. The Reaction Editor permits the user to define reaction models. A reaction model is composed of a series of simple reaction steps including any combination of volume reaction, membrane reaction, and flux reaction.

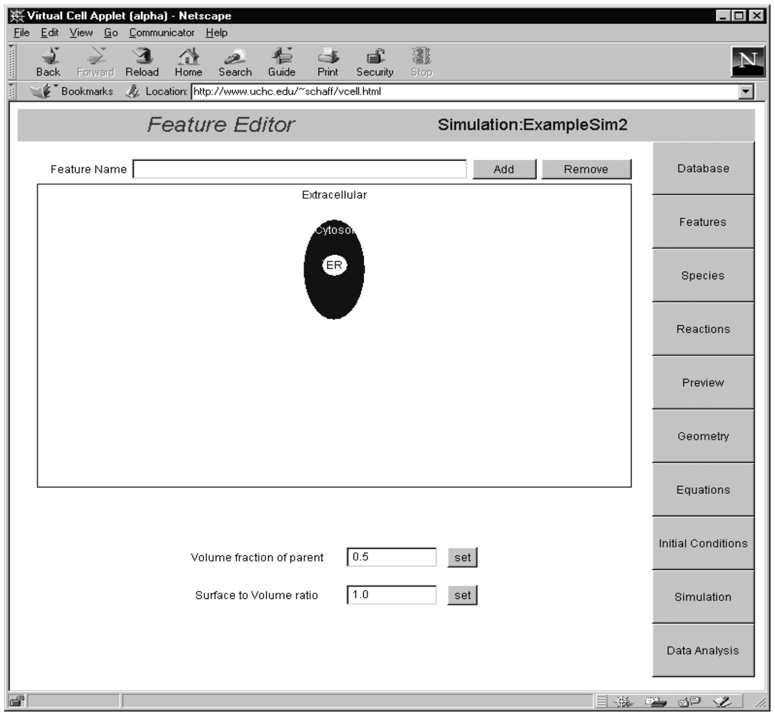

Virtual Cell Applet (alpha) - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location: http://www.uchc.edu/~schaff/vcell.html
Feature Editor
Simulation:ExampleSim2
Feature Name
Add
Remove
Extracellular
Cytosol
ER
Volume fraction of parent
0.5
set
Surface to Volume ratio
1.0
set
Database
Features
Species
Reactions
Preview
Geometry
Equations
Initial Conditions
Simulation
Data Analysis

Figure 2 - Feature Editor

3.2.2 Geometric Modeling

The Geometry Builder is a standalone application that permits the construction of two or three-dimensional cellular geometric models based on a series of image files. Due to the security restrictions of Java applets reading local file systems, the Geometry Builder is capable of running as a signed applet (trusted by the browser) or as a standalone application which does not have the same restrictions.

3.2.3 Model Analysis

The Compartmental Simulation (Preview) component executes a compartmental (single point) simulation based on the defined physiological model and the geometric assumptions entered in the Feature Editor (surface to volume ratios and volume fractions). This results in a set of nonlinear ordinary differential equations that typically are solved in seconds. This allows an interactive, though manual, modification of parameters and a quick determination of the effect over time. Once the simulation is complete, each species can be viewed easily.

The Equation Viewer displays the equations generated as a result of mapping the physiological model to either a cellular geometry model (spatial simulation) or a single point approximation (compartmental model). The parameter values may be substituted (and the expression simplified), or left in their symbolic representation.

3.2.4 Spatial Simulation

The Geometry/Mesh Editor allows some participation in the choice of spatial resolution, thus considering the computational costs and the goodness of geometric representation. This interface directs the binding of regions of the segmented geometry to the corresponding features within the physiological model. An orthogonal mesh is specified and displayed interactively.

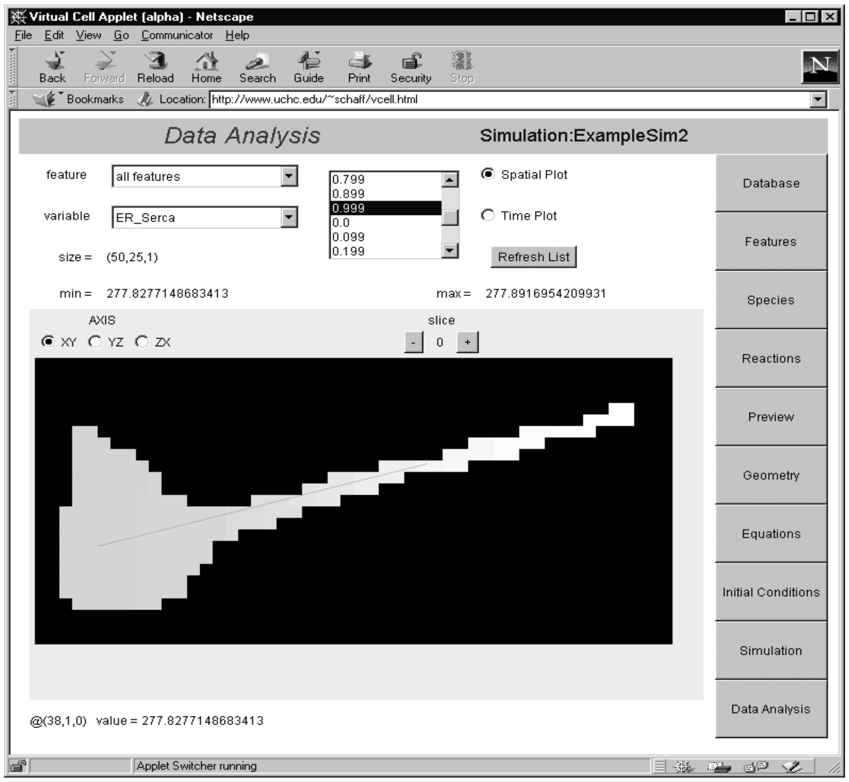
The Initial/Boundary Condition Editor allows the specification of initial conditions and boundary conditions for each of the species for each feature. The boundary conditions for each simulation border is specified independently for each feature, this gives maximum flexibility. For example, the concentrations may be specified at simulation boundaries in the extracellular space to indicate a sink. And a zero molecular flux may be specified at a simulation boundary belonging to cytosol to indicate symmetry of function with the missing portion of cytosol (the implied mirror image).

The Simulation Controller permits the specification of the time step and end times of the currently defined spatial simulation. The Simulation Controller services are then invoked, including automatic code generation and the initiation of a remote simulation job. It is worth noting that the time step will generally not be the users responsibility, and will be constrained by the problem and the spatial discretization.

The Simulation DataSet Viewer (figure 3) displays the results of the current spatial simulation. The species concentrations are displayed superimposed on the mesh. The analysis capability includes graphing the spatial distribution of a species as a line scan and graphing a time series at a single point.

3.2.5 Storage

The Database Access Form presents a rudimentary model and simulation storage capability. The current implementation allows whole physiological models, geometric models, and simulation contexts to be stored and retrieved. The simulation context is stored in a way that includes the physiological and geometric models such that it encapsulates all of the information to reproduce and describe a particular spatial simulation. There is, however, no ability to querying the stored models for specific attributes. The models are currently stored intact using Java's



Location: http://www.uchc.edu/~schaff/vcell.html

Data Analysis   Simulation:ExampleSim2

feature: all features

variable: ER_Serca

0.799
0.899
0.999
0.0
0.099
0.199

[✓ Spatial Plot] Time Plot

size = (50,25,1)

min = 277.8277148683413   max = 277.8916954209931

AXIS: [✓ XY] YZ ZX   slice: 0

@(38,1,0) value = 277.8277148683413

Figure 3 - Simulation Data Analysis

Object Serialization capability.

4. Conclusion

The need for the Virtual Cell arises because the very complexity of cell biological processes severely impedes the application of the scientific method. A pair of separate factors that contribute to this problem are addressed by the Virtual Cell.

First, the large number of interdependent chemical reactions and structural components that combine to affect and regulate a typical cell biological process forces one to seek the help of a computer to build a model. This issue is the subject of an eloquent essay by Dennis Bray[8]. We are now faced with an overwhelming body of data describing the details of individual molecular events occurring inside cells. As Bray puts it, “What are we to do with the enormous cornucopia of genes and molecules we have found in living cells? How can we see the wood for the trees and understand complex cellular processes..?” Brays solution: “Although we poor mortals have difficulty manipulating seven things in our head at the same time, our silicon protégés do not suffer this limitation. ...The data are accumulating and the computers are humming. What we lack are the words, the grammar and the syntax of the new language.”

The second factor recognizes that scientists trained in experimental cell biology are not typically equipped with sufficient mathematical, physical or computational expertise to generate quantitative predictions from models. Conversely, theoretical biologists are often trained in the physical sciences and have difficulty communicating with experimentalists (bifurcation diagrams, for example, will not serve as a basis for a common language). By maintaining the physical laws and numerical methods in separate modular layers, the Virtual Cell is at the same time accessible to the experimental biologist and a powerful tool for the theorist. Also, by maintaining a direct mapping to experimental biochemical, electrophysiological and/or image data, it ensures that simulation results will be communicated in a language that can be understood and applied by all biologists.

Acknowledgments

We are pleased to acknowledge the support of NIH (GM35063) and the Crtitical Technologies Program of the State of Connecticut.

References

1. J. Schaff, C.C. Fink, B. Slepchenko, J.H. Carson, L.M. Loew, “A general computational framework for modeling cellular structure and function” *Biophys. J.*, **73**, 1135 (1997)
2. R. Kupferman, P.P Mitra, P.C. Hohenberg, S.S-H. Wang, “Analytical calculation of calcium wave chaaracteristics”, *Biophys. J.* **72**, 2430 (1997)
3. J. Sneyd, J. Keizer, M.J. Sanderson, “Mechanisms of calcium oscillations and waves: a quantitative analysis” *FASEB J.* **9**, 1463 (1995)

4. Y.-X. Li, J. Rinzel, "Equations for $InsP_3$ receptor mediated calcium oscillations derived from a detailed kinetic model: A Hodgkin-Huxley like formalism", *J. Theor. Biol*. **166**, 463 (1994)
5. S.V. Patankar, "Numerical Heat Transfer and Fluid Flow" (Taylor and Francis, 1980)
6. M. Hines in *Neural Systems: Analysis and Modeling*, "NEURON: A program for simulation of nerve equations" Ed. F. Eeckman (Kluwer Academic Publishers, 1993)
7. J.M. Bower and D. Beeman "The book of GENESIS: Exploring realistic neural models with the general neural simulation system." (Springer-Verlag, New York, 1994)
8. P. Mendes "GEPASI: a software package for modeling the dynamics, steady states and control of biochemical and other systems" *Comput. Applic. Biosci*. **9**, 563 (1993).
9. D. Bray "Reductionism for biochemists – how to survive the protein jungle" *TIBS*, **22**, 325 (1997)