Published in final edited form as:
IET Syst Biol. 2008 September ; 2(5): 352–362. doi:10.1049/iet-syb:20080102.

The Virtual Cell Modeling and Simulation Software Environment

Ion I. Moraru, **James C. Schaff**, **Boris M. Slepchenko**, **Michael Blinov**, **Frank Morgan**, **Anuradha Lakshminarayana**, **Fei Gao**, **Ye Li**, and **Leslie M. Loew**
Center of Cell Analysis and Modeling, University of Connecticut Health Center 263 Farmington Ave, Farmington, CT 06030

Abstract

The Virtual Cell (VCell; http://vcell.org/) is a problem solving environment, built on a central database, for analysis, modeling and simulation of cell biological processes. VCell integrates a growing range of molecular mechanisms, including reaction kinetics, diffusion, flow, membrane transport, lateral membrane diffusion and electrophysiology, and can associate these with geometries derived from experimental microscope images. It has been developed and deployed as a web-based, distributed, client-server system, with more than a thousand world-wide users. VCell provides a separation of layers (core technologies and abstractions) representing biological models, physical mechanisms, geometry, mathematical models and numerical methods. This separation clarifies the impact of modeling decisions, assumptions, and approximations. The result is a physically consistent, mathematically rigorous, spatial modeling and simulation framework. Users create biological models and VCell will automatically (i) generate the appropriate mathematical encoding for running a simulation, and (ii) generate and compile the appropriate computer code. Both deterministic and stochastic algorithms are supported for describing and running non-spatial simulations; a full partial differential equation solver using the finite volume numerical algorithm is available for reaction-diffusion-advection simulations in complex cell geometries including 3D geometries derived from microscope images. Using the VCell database, models and model components can be reused and updated, as well as privately shared among collaborating groups, or published. Exchange of models with other tools is possible via import/export of SBML, CellML, and MatLab formats. Furthermore, curation of models is facilitated by external database binding mechanisms for unique identification of components and by standardized annotations compliant with the MIRIAM standard. VCell is now open source, with its native model encoding language (VCML) being a public specification, which stands as the basis for a new generation of more customized, experiment-centric modeling tools using a new plug-in based platform.

1 Introduction

The difficulties associated with formulation of quantitative mathematical models and computer simulations of biological processes have hampered progress in the fledgling field of computational biology. This is slowing down the advance of systems biology in the quest to realize the ultimate potential promised by the rapidly acquired new, large scale, “omic” datasets [1]. There has been an explosion of development efforts of computer tools to analyze and use this data to build models and run simulations. The website of the Systems Biology Markup Language (SBML; http://sbml.org/; a community-based effort to create a common language to encode kinetic models), currently lists more than 100 software packages that support to some degree the SBML standard. But because biologists rarely have sufficient training in the mathematics and physics required to build quantitative models, modeling has often remained the purview of theoreticians, who have the appropriate training but little experience in the laboratory. This disconnection to the laboratory has limited the impact of mathematical modeling in cell biology.

The Virtual Cell (VCell) project is developing a modular computational framework that permits construction of models, application of numerical solvers to perform simulations, and analysis of simulation results. To address the problems discussed above, VCell has been designed from the beginning to be a computational modeling platform that is easily accessible to cell biologists [2]. This is achieved by abstracting and automating the mathematical and physical operations involved in constructing models and generating simulations from them (see Section 2). VCell is available via the web, allowing users to collaborate, share, and publish their work, and access external resources (see Section 3). Additionally, the client-server implementation allows users to run large and complex simulations without requiring them to have access to their own high performance computer hardware and sophisticated numerical tools and software libraries (see Section 4). The current features enable users to create and run simulations of biochemical networks, membrane transport and electrophysiology. These can be formulated as compartmental ordinary differential equation models and numerically solved with a choice of ODE or stochastic solvers.

One distinctive feature of VCell is that it permits the incorporation of realistic experimental geometries within full 3D spatial models. Thus, the effects of diffusion and flow can be explicitly incorporated into models, and simulations provide solutions to the corresponding partial differential equations. An intuitive graphical interface includes options for database access, geometry definition (including directly from microscope images), specification of compartment topology, species definition and assignment, chemical reaction input, membrane transport mechanisms (including voltage dependence), initial conditions, boundary conditions, simulation solver choices, and computational mesh. As an alternate to this graphical model building interface, VCell also provides a mathematical interface that allows theoreticians to examine and elaborate models through purely mathematical formulations. It allows for the direct entry of mathematical equations that describe a model, through a declarative language (Virtual Cell Mathematics Description Language, VCMDL). The mathematics is then automatically translated into C++ programming code that can than be sent to the numerics solvers. Thus, modelers are relieved of the drudgery of writing ad hoc code for every new modeling task. Furthermore, a VCMDL description of a model can be produced directly and automatically from a model that has been created within the graphical biological interface. This dual interface has the additional benefit of encouraging communication and collaboration between the experimental and modeling communities.

As a result of this approach, VCell has been rapidly adopted as a tool of choice for kinetic models, in particular by experimental biologists and by researchers interested in spatially resolved simulations. To date, more than 1,900 VCell users have created tens of thousands of models and simulations, with over 300 publicly available models in the VCell database, and over 30 publications in high-profile journals (see http://vcell.org/ for a current listing). Below we describe into more detail the overall design, architecture, and capabilities of VCell (Sections 2 through 4), highlight some of the unique features of VCell (Sectiuons 5.1 through 5.6), as well as upcoming developments (Section 6).

2 Three Layers of Abstraction – Separating the Model from Simulation

One of the unique and innovative features in the design of VCell is the introduction of three distinct layers of abstractions in the modeling process. This follows a paradigm similar to hypothesis-driven experimental studies. Figure 1 schematizes the way in which the modeling process is structured within the Virtual Cell. The hierarchical layered structure starts with the system “Physiology” - essentially the mechanistic hypothesis to be investigated. It includes the description of the molecular species and hypothesized interactions localized to cellular structures. Cellular structures need only be specified as the topological arrangement of membranes and membrane bounded compartments. Interactions can be biochemical reactions

defined either within volumetric compartments of the cell or in membranes; molecular fluxes across membranes; and electrical currents. Additionally, the Physiology captures the biochemical and biophysical properties of these interactions as rate laws for reactions and fluxes (that can be determined as an explicit function of the local environment, e.g. concentrations, surface densities, membrane potential, and of one or more kinetic parameters, e.g. k_{on} and k_{off}). Mass action and Michaelis-Menten rate laws are available automatically, and arbitrary user-defined general kinetic expressions can also be entered. Membrane transport kinetics can be specified with expressions for molecular flux or, for ions, the electric current. The transport kinetics can be described in terms of standard electrophysiological formulas (e.g. Goldman-Hodgkin-Katz permeability or Nernst conductance) or as user defined molecular flux or current.

Such a decomposition of modeling concepts makes possible the effective reuse of the model in multiple contexts. The abstract physiological modeling concept allows a single quantitative mechanistic hypothesis to be validated under multiple experimental conditions at different scales and using different physical approximations. While the Physiology describes the overall mechanistic process under study, it can spawn several "Applications" that pose concrete, experimentally testable problems. For an Application, the user specifies geometry, boundary conditions, default initial concentrations and parameter values, and whether any of the reactions are sufficiently fast to permit a pseudo-steady state approximation. Importantly, an application can be associated with geometries in 3 dimensions, 2D, 1D or 0D (where all diffusive processes are approximated as instantaneous). For 0D compartments, the user is asked to specify the volume and surface area of each structure that was defined in the Physiology; this is important for the proper calculation of concentrations associated with molecules that move between the membrane and an adjacent volumetric compartment. Individual reactions can be disabled as an aid in determining the proper initial conditions for a prestimulus stable state. Concentrations of specific molecules can be clamped as in a virtual knockout experiment. An Application of a Physiology is sufficient to completely describe the governing mathematics of the model, and a VCMDL file is generated at this point.

VCell is designed to maintain a separation between this mathematical description and the details of how the simulations are implemented. As shown in Figure 1, several simulations can be spawned from a given Application. That is, the mathematics can be solved with multiple choices of numerical solver, time step, mesh size for spatial simulations, time duration, and overrides of the default initial conditions or parameter values to permit parameter scans. Within the simulation branch of the VCell modeling process, the user can also perform sensitivity analysis for non-spatial deterministic models. Any of such particular simulation specifications are sufficient to describe the software requirements for numerically calculating solutions, and VCell automatically generates and runs the appropriate C++ code to produce simulation results. The fact that the Virtual Cell is designed to handle any reaction system in any geometry precludes the formulation of a general analytical solution for the problem. There are two generic approaches to numerical solutions – stochastic and continuous. The continuous approach provides a deterministic description in terms of average species concentration. This approach is effective and accurate so long as the number of molecules in a system is large, such that thermal stochastic fluctuations around average values can be ignored. The implementations for the deterministic algorithms have been described before [3–6], and more details about the recent implementation of stochastic algorithms can be found in Section 5.4 below. Section 5.5 also provides more details on some of the special features available to facilitate the generation and analysis of simulations.

3 The VCell Database and External Resources

A key feature of VCell that is essential for supporting internal consistency and reusability of model elements is that models, geometries, as well as their individual subcomponents are all stored and maintained in a relational database. Database browsers that emulate an explorer file-type interface, are provided for each of the top-level containers (BioModels, MathModels, and Geometries) that VCell users organize and manage directly. The graphical interface allows interactive browsing of objects in the database and viewing of their corresponding descriptive metadata. A Model Comparator Tool compares any two models based on their XML representation and the resulting differences are displayed. Additionally, the database can be queried for reusing components of models. For example, the BioModel reaction editor has a tool to permit searching for individual reactions from among all accessible user models. These are then automatically inserted and linked to existing components within a reaction network. VCell also offers access to external databases to provide additional modeling resources and data to users. To identify a molecular species unambiguously, VCell provides the ability to bind to a dictionary of species (controlled vocabulary) derived from KEGG (their Compound database for small molecules, metabolites, lipids, etc. and their Enzyme database for enzymes) and SwissProt (for proteins). VCell also allows the direct import of data from external databases to automate the process of defining and binding new species and generating fragments of reaction schemes (Figure 2).

The database-centric design of the persistent storage elements of VCell also allowed for the development of simple mechanisms for collaborative work and publishing of models and simulation results. Permissions to share a model with designated users or with the entire user community (i.e. making a model ‘Public’) can be set from within the database browsers. A shared or public model can be opened by a user just like one of his/her own models, but altering the model or running new simulations requires that the user save a copy of the model under his/her own username.

Mathematical models developed for particular cellular mechanisms and functions often could be ideal starting points for new studies, e.g. for similar problems in different experimental contexts, or for inclusion in larger, more comprehensive models, etc. In practice, however, a major roadblock in the effective leveraging of existing published modeling studies, has been the lack of mechanisms that facilitate the comparison and reuse of computational models. Often models have been simulated using custom-developed software, and even when publicly available software platforms have been used, the exchange of models encoded in different formats has proven to be difficult. We first need a common vocabulary and a common language to describe them. A significant step towards this goal has been the recent development of formal languages to encode models, such as SBML [7] or CellML [8], as well as of appropriate ontologies, such as BioPAX [9], which have been enthusiastically embraced by researchers, modelers, and tool developers. The VCell team has been involved in the development of SBML from its inception and we are continuously working on achieving and maintaining the best possible compatibility for SBML export and import of VCell models. VCell’s import/export facilities currently support SBML, CellML, and Matlab for exchanging of VCell model data with other software tools. However, neither CellML or SBML currently support spatial modeling and they have limited support for the abstractions used in VCell for membrane transport. Additionally, SBML does not support electrophysiological phenomena. Therefore, VCell uses an XML dialect developed specifically to completely capture all VCell model elements (VCML). In addition to allowing ready transfer and documentation of VCell models outside VCell’s central database, VCML is now also used internally for efficient transport of model elements between software modules. The final critical step towards facilitating exchange and reuse of quantitative models is the formulation of standards for model curation and annotation. A community based effort has recently proposed the MIRIAM standard (minimum

information requested in the annotation of biochemical models) [10], and created a growing repository of compliant models (http://biomodels.net/). VCML has been released as a public specification, and has been enhanced to support MIRIAM compliant annotations in RDF format. To facilitate curation efforts, a user interface for viewing, adding, editing, and deleting MIRIAM annotations has been developed in VCell.

4 Software Architecture

The design approach to the Virtual Cell client interface (that implements the user functionality described above) has also inspired and facilitated the creation of an overall modular software structure. This modular structure allows independent development and validation of individual services, which are loosely coupled, which helps to create an overall architecture that is easier to maintain and grow. This also allows a progressive introduction of new capabilities. For example, the solvers are independent of the biological and physical abstractions. One can start with independent development and validation of numerical tools that are designed to handle certain new classes of mathematical problems (e.g. partial and ordinary differential equations of certain forms). When the supported class of VCell mathematics is expanded, this new capability is instantly available to a mathematically oriented user (through the MathModel interface). The richer mathematical form then provides new flexibility for mapping physiological mechanisms and geometry to a concrete mathematical form required for simulation. This can be achieved by separate development of the user interface describing these new abstractions (in the BioModel interface). As a result of this approach, VCell became a continuously evolving, large-scale software project where complexity has been managed by decomposing the software into a collection of loosely coupled services interacting with conceptually simple data abstractions. This strategy is critical for the robustness and flexibility of the distributed architecture of the platform (remote clients/database server/compute nodes/simulation data storage, etc).

The main software components and some of their interrelationships are illustrated in Figure 3. The Math Generation Service was originally deployed as a server-side service, but is now deployed locally on the client. This service generates a declarative Mathematical Description from an Application. This transformation is invoked before saving or running a simulation, and serves to decouple the biological representation from the mathematical representation. Many of the other client-side components were developed to simplify the user interface design, by systematically decoupling the user interface components (editors, etc) from the, simulation database, and data management layers. The latter are being abstracted from the client as “remote” services accessed via protocol wrapper application programming interfaces.

All of these services may be deployed as actual remote services, or in a standalone application context. There is no a priori reason for any of the modules to reside on separate computers, and the entire framework, including the client, could actually be deployed on a single workstation. However, both of the currently deployed software versions (a “release” version and a “beta” version) are configured in a distributed, client-server architecture. The two main reasons to adopt a client-server design were (i) the desire to have a multi-user environment that can facilitate collaboration and model reuse, and (ii) resource requirement issues that are particularly important in the case of VCell’s primary target audience, cell biologists, who typically have limited access to sophisticated and high performance computer hardware and software. Other considerations that influenced the choice of deployment strategy were: a broad compatibility of the user interface components with different PC platforms; user’s need for continuous access and the ability to run uninterrupted simulations for days; the need to continuously update the software while maintaining backwards compatibility; the need to add capabilities by linking to other software and services, etc.

The server-side components were originally deployed as Java RMI services (Remote Method Invocation) with the appropriate RMI protocol wrappers. New protocol wrappers were then developed to adapt these services to an architecture based on a message-oriented middleware solution. The technical details of software design and implementation are beyond the scope of this article. We will only highlight here the major components involved and their overall functionality. Several different classes of services exist, and they are deployed as multiple redundant instances on various nodes of the compute clusters that serve VCell. Services do not communicate directly with each other, but subscribe to, and post messages to, message queues, and execution status is being recorded in a relational database. This ensures robustness against failures of individual service instances (with automatic resubmission of incomplete jobs), and guarantees accurate status reporting to remote clients. The *Database Service* (which internally consists of multiple database layers) provides a component interface consisting of simple object persistence operations such as load, save, update, delete, annotate, and query. The *Simulation Data Service* provides simulation results retrieval in reponse to client requests. It implements caching and data reduction to minimize network traffic. A closely-related *Data Export Service* packages user-selectable simulation data (one or more variables, one or more time points, selected spatial regions) into several different formats (raw data, images, movies, etc.), for download and use outside of VCell. The *Simulation Dispatch Service* is the first step for scheduling simulation jobs, enforces user quotas, and monitors running simulations. Several different types of *Simulation Worker Services* handle the dispatched simulation jobs. ODE-only jobs (from compartmental simulations) can be run directly by the Java worker service itself (e.g when using Runge-Kutta-Fehlberg solver), or the worker service generates an input file and invokes a standalone C++ precompiled executable (e.g. when using the IDA solver). For PDE jobs (from spatially-resolved simulations), the worker service is responsible for code generation and compiling platform-specific code that is being submitted together with relevant input files to a batch scheduler We currently support several batch scheduler middleware software, both local and grid-based (e.g. PBS Professional, Condor-G). To further enhance the robustness and ease of management of the VCell server side, we have recently changed the service implementations to allow them to be themselves deployed as “jobs” submitted under the control of batch scheduler middleware. As a result, we could eliminate the service manager and configure all three VCell sites (alpha, beta, release) to leverage combined cluster pools, for both compute and storage resources, creating a global fully-automated load-balancing and self-healing architecture.

This implementation of VCell’s distributed framework has been critical for scalability, interoperability, flexibility, continuous enhancement, and multi-user support over more than 10 years. A natural consequence of the continued growth in the number and sophistication of users and in the software’s features and capabilities is the continuously increasing resource requirements for the VCell server-side. At the time of this writing, a total of > 2 TFlops of processing power and > 20 TB of data storage are available for VCell users. Our practical experience has shown that, given the average computational cost of simulations and the size of the active user base, this allows for non-contentious operation of the job scheduler using a per-job maximum storage quota of 20 GB and a per-user maximum of 40 concurrent jobs (80 in the case of parameter scans), and currently with no per-user total archival limit. However, it is important to emphasize that the modular design of the VCell framework makes it easy to construct many alternate implementations, ranging from fully packaged, self-contained local installations, to grid-based solutions, which will be featured in some of the upcoming VCell applications that are under development (see also section 6 below).

5 Special Features

In addition to the overall design of the VCell framework described above, we will present in the subsections below a few of the more specialized features that make VCell a unique modeling environment.

5.1 MathModels

As mentioned in the introduction, this review mainly addresses the model building and simulation capabilities of VCell that are accessible through the biological user interface (the BioModel workspace). However, VCell also enables the mathematically-savvy user to build models that cannot easily described within the confines of the supported abstractions of the BioModel editor, by use of the MathModel workspace. This alternate client interface starts directly with the VCMDL declarative mathematical language, where (quasi)arbitrary systems of algebraic and differential equations can be mapped to various geometries. Similar to the BioModel interface Application, the MathModel can then spawn off one or more simulations and can use any of the simulation specifications and solver features available within VCell. MathModels are also stored within the VCell model database and can be shared among users and/or published.

One common scenario to use the MathModel workspace is as follows: a user describes the model using the Physiology module in a BioModel and creates an Application for it, which has an automatically generated mathematical description that can be exported to create a MathModel. The user then can add to the mathematical system additional user-defined equations, which could encode some phenomenological models of biological processes that cannot be described explicitly as molecular interactions. For example, one can write some rate rule connecting an arbitrary extracellular stimulus to the time course of gene expression of a particular gene in the nucleus, where the intermediate steps that must involve signals in the cytosolic compartment are either unknown, or purposely neglected for simplicity. In a Biomodel, there would be no simple way to bypass the cytosolic compartment.

Another common scenario is the use of the MathModel workspace to enable users to model phenomena for which a more explicit representation exists in VCell, including specialized solver capabilities, but for which the biological description abstracting the process in a graphical interface is not yet available in the BioModel workspace. For example, it is now possible to simulate systems where transport includes not only diffusion but also active flows, such as movement of vesicles driven by molecular motors [11]. For this, additional specifiers for the Cartesian components of a velocity field are introduced: VelocityX, VelocityY, and VelocityZ. Diffusion-advection-reaction equations are solved in VCell using a hybrid method that switches between central difference and upwind discretization schemes for the drift term depending on the local Peclet number [12]. The algorithm was validated extensively against exact solutions and through regression testing. Note that in VCell MathModels, both the diffusion coefficients and the velocity components can be expressed as functions of time and spatial coordinates.

5.2 Membrane Diffusion

Lateral diffusion within the bounds of a biological membrane can influence the spatial localization and propagation of in vivo signaling processes, as well as the cells response to localized experimental interventions such as photobleaching and uncaging. The Virtual Cell supports trans-membrane flux (e.g. channels and pumps), membrane-volume interactions (e.g. translocation or membrane signaling) as well as reactions and diffusion of species within the confines of a membrane. Of these capabilities, the proper handling of diffusion and reactions within an arbitrarily shaped membrane coupled to bulk diffusion and reactions was the most

difficult to properly implement and resulted in the development of a novel implicit surface meshing approach [6]. Using this approach, we preserved our ability to completely automate the generation of computational grids, with special requirements on the embedded surfaces to allow accurate and efficient computation of diffusion on arbitrary membrane surfaces. This capability may be used, for example, to facilitate the interpretation of fluorescence recovery after photobleaching (FRAP) experiments which involve membrane-bound species that may return to the depleted region due to both membrane diffusion as well as exchange with the volume (Figure 4). Molecular species that are associated with a membrane are described by their surface densities and may diffuse with a diffusion coefficient which is specified in the modeling application.

5.3 Visualization Tools

A surface-based, 3D interactive visualization tool has been integrated for the display of 3D geometries and to view simulation results from 3D simulations. This viewer is capable of displaying shaded polygonal surfaces or wireframe meshes with transparency and can be manipulated by the user with a virtual trackball. The user can select surface regions either by selecting regions of the polygonal surfaces or by using analytic conditions). For 3D geometry creation, this visualization gives the necessary visual feedback to adjust the geometric description (equations or images). For visualizing the result of 3D simulations, the simulated concentrations of molecules (or any membrane associated variable or function) can be displayed as a pseudo-colored surface, as illustrated in Figure 4. Statistics of the simulation results can be gathered over selected regions, and the user can request line scans, or time plots of the underlying data. Volumetric data is displayed one slice at time in a two dimensional image viewer that provides the same data analysis options. An additional very flexible and powerful tool available for visualization of both surface and volumetric data is the kymograph. The kymograph view summarizes the spatiotemporal relationships in a simulation along any user-defined line. A pseudocolored image represents the intensity as a distribution over space (x axis) and time (y axis). Two attached graphs display concomitantly the concentration of the elected variable plotted over distance and over time and are interactively updated in response to the current mouse position over the space-time 2D image.

Associated with the visualization, an extensive export utility provides for server-side data reduction (see also Section 3 above). One export capability that deserves to be mentioned for multivariate spatio-temporal data is the creation of downloadable movies, in QuickTime or animated GIF formats, of 2D slices of volumetric data or 3D renderings of surface data. Data for different variables can be exported as separate movies, or optionally stacked up in a single movie frame with synchronous playback.

5.4 Stochastic Applications

VCell currently supports nonspatial (compartmental) stochastic applications for models created with the graphical (“biological”) user interface as well as nonspatial stochastic models in the VCell mathematical workspace that can be created manually following a VCMDL template. In VCell 4.4 (as of this writing, VCell beta), simulations are performed using an exact stochastic simulator based on the Gibson-Bruck Next Reaction method [13], an optimized version of one of the Gillespie algorithms [14] that interprets reactions as Poisson processes. The VCell stochastic tool allows a user to compute individual trajectories as well as perform multiple trials with either default or user-specified seeds. In the case of multiple trials, VCell automatically calculates histograms of the number of copies for the species of interest (see Figure 5). While the algorithm simulates events as they randomly occur, the user can specify regular time intervals at which the results should be displayed.

One problem with deriving stochastic applications from a general VCell BioModel is that the reaction kinetics are introduced in the model as deterministic. The unambiguous interpretation of these in terms of Poisson stochastic processes is possible only for mass-action kinetics, and even in this case, the reversible reactions should be decoupled into two separate processes. To address this issue, an analyzer tool has been developed that automatically maps mass-action reaction rates to probabilistic propensities, and facilitates the mapping of other reactions. In the BioModel interface, VCell offers a number of kinetic types, including a "general kinetics" option that allows for arbitrary rate expressions. The tool identifies the kinetic type of a reaction or membrane flux, and in the case of general kinetics, automatically determines, by parsing the rate expression, whether the mechanism might be a combination of individual Poisson processes (for example, if they are passive fluxes). It then informs the user: (i) which of the mechanisms with kinetics other than mass action can be directly translated into probabilistic propensities (and asks permission to do it automatically), and (2) which of the mechanisms have to be recast manually by the user into a combination of one or more mass-action types.

Exact stochastic simulators may be ineffective for systems that involve significantly different time scales ("stiff" systems). A number of hybrid algorithms [15,16], which allow a user to make an appropriate compromise between accuracy and efficiency, are currently being tested in VCell 4.5.

5.5 Simulation Tools

A major problem with models of intracellular reaction networks is the overall lack of comprehensive quantitative data, be it kinetic constants of molecular interactions, or spatially resolved concentration distributions of the molecular species involved. Most often, modeling projects, even when started from some previously developed models, initially involve considerable effort to fine-tune the model to fit certain experimental results, before being able to use simulations to explore the predicted behavior of the system. For spatial models in particular this can be a daunting task for a couple of reasons. On the one hand, exploring the parameter space of PDE-based mathematical systems is much more compute-intensive than for ODE-only systems. On the other hand, performing appropriate fitting routines automatically is not trivial due to the difficulties in defining the objective function – for example, what is the relative importance of concentration values of a molecule of interest in different areas of the cell when judging the quality of fit?. Many of the specialized features available in VCell's Simulation layer are geared towards helping users overcome these problems.

One of the common workflows employed by VCell users is to first develop and analyze a model in a compartmental Application (i.e. a zero-dimensional model where a well-mixed compartment approximation is being used – essentially assuming infinitely fast diffusion rates). This produces and ODE-only mathematical description, for which the Simulation interface provides a "Parameter Estimation" tool. Import of multiple sets of experimental data (e.g. time-course data) in CSV format is supported, which can be optionally subsampled. All parameters in the model are available for selection in an automatic multiple-fitting routine, with automatically provided initial guesses and bounded ranges, but which can be manually specified and further constrained. Two algorithms are currently available to perform the optimization task, the Powell method and CFSQP (C code for Feasible Sequential Quadratic Programming). Such parameter estimation tasks can be independently saved and reused, and the results can be used via "smart copy-and-paste" to initialized new Applications using the optimized parameter values.

The automatic translation of a compartmental Application into a spatial Application simply by changing the Geometry to a spatially-resolved one allows users to test the results of the compartmental simulations in a spatial context. Moreover, the "smart copy-and-paste" allows

users to easily apply entire sets of parameter values (either initial or boundary conditions or reaction parameters) to pre-existing Applications and simulations, even from a completely different BioModel. But often more tweaking of parameters is necessary for spatial simulations, and, furthermore, some parameters cannot be optimized in compartmental models – e.g. diffusion rates, since they are not present in compartmental approximations. Therefore, an additional Simulation feature has been implemented (for both spatial and compartmental Applications) – parameter scans ("sweeps"). This feature allows user to select one or more parameters that can be chosen to vary, independently or in lockstep. The parameters to be scanned for can be defined directly as lists of values, or with an arbitrary number of equal intervals (linear or log) across a user-defined range (upper and lower bounds). All values of all parameters with the "scan" flag set are combined to create a matrix of simulations that are performed in parallel and that generate a number of result sets that are logically linked and can be displayed in a single result set viewer. A quite powerful feature is the possibility to specify a particular parameter to be varied in a manner dependent on another parameter, through a free-form mathematical expression (e.g. the diffusion coefficient for molecule X may be scanned across 5 different values, while the diffusion coefficient for molecule Y is defined as always being twice that of X).

Finally, a simulation tool recently introduced that further helps model analysis and comparison with experimental data is the possibility to specify and express spatially-resolved distributions of values, labeled "Field data" in VCell. This allows, for example, the initializing of a spatial simulation with explicit non-uniform concentration values for one or more variables, which could be provided directly from quantitative microscopy data. Not only experimental data can be recorded as Field data, but also results of simulations performed within VCell. This way, the end results of a particular simulation can be used to initialize new Applications or simulations, allowing users to effectively "chain" simulations and model complex experimental protocols.

5.6 BioNetGen@VCell

Models of biochemical kinetics accounting for dozens of different molecular species are a norm; models accounting for hundreds of species and reactions are no longer rare [17]. This is a typical situation when protein complexes and phosphoforms are included into a model. For example, a receptor with 10 tyrosine phosphorylation sites can exist in 2^{10}=1024 different phosphoforms. Many or all of these phosphoforms have to be accounted for individually in a quantitative model to simulate and predict the time course for receptor-mediated signaling [18]. Manually specifying a list of species and reactions becomes error-prone and slow. Furthermore, when potential protein complexes are being included, the size of reaction network can easily increase by a few orders of magnitude. A solution for this combinatorial complexity challenge is provided by a rule-based approach, in which a model is specified in the form of bio-molecular interaction rules [19]. A modeler specifies molecules, their interacting and modification domains, such as tyrosines and SH2 domains, and rules of activities and interactions among domains and molecules. Rules are used to automatically generate a model in the form of a reaction network comprised of all chemical species corresponding to specified molecules, and all transitions among these species, thus freeing the user from the intense bookkeeping that would be required to enumerate such a network by hand. This approach has been developed into general-purpose software, BioNetGen [20]. For the Virtual Cell framework, we have implemented it as a BioNetGen@VCell service invoked by Virtual Cell. It allows a user to write or upload a BioNetGen input file and perform reaction network generation and time courses simulation. A VCell BioModel can be then generated and processed by the Virtual Cell. However, the current implementation does not eliminate all the problems discussed above. While automatic generation of all required species and reactions defined by some sets of rules is guaranteed to be accurate, it can nevertheless lead to reaction

networks that are too large to be simulated. However, given the relatively small number of molecules inside cells, only a subset of all the possible species and complexes do occur at any given time. While the standalone BioNetGen can perform on-the-fly model generation during the course of simulation, this capability is incompatible with the current onetime model import mechanism in the BioNetGen@VCell service. A next generation solution will include such dynamic model reduction for stochastic Applications in VCell.

6 Future Directions

The current VCell platform has so far relied exclusively on a proprietary client-server architecture that provided users with a web-accessible environment. It facilitates development of complex modeling studies, including collaborative work, without difficult software installation procedures, high-end hardware requirements, or intimate knowledge of mathematics, physics, and computer science. This has attracted a considerable number of experimental biologists to computational modeling. We will continue to build on this strength by using the VCell platform to develop problem solving environments tailored for quantitative cell biology studies. On the one hand, new core capabilities of VCell are being added, such as support for modeling mechanical forces, accounting for diffusion in crowded spaces, enhanced numerical tools (hybrid solvers and parallel solvers) etc. On the other hand, in a departure from the "one size fits all" strategy, customized environments are being created, that can be either web-based or standalone, focusing on the particular needs of modeling specific experimental contexts (such as "Virtual Microscopy"-type tools), with explicit support for protocols and integration of experimental and simulation data. Last, but not least, is a continued effort to enhance the model exchange capabilities as well as allowing direct integration with external resources (databases, ontologies, and software components). The ultimate goal is for VCell to enable researchers to use powerful and flexible workflows that can integrate the wealth of rapidly accumulating quantitative systems biology data to generate predictive simulations of complex cellular behavior in a collaborative, shared environment.

Acknowledgments

This work is supported by NIH Roadmap Award for a National Technology Center for Networks and Pathways (#U54RR022232) and grant # P41RR013186 from the NIH National Center for Research Resources.

References

1. Moraru II, Loew LM. Intracellular signaling: spatial and temporal control. Physiology 2005;20:169–179. [PubMed: 15888574]
2. Schaff J, Fink CC, Slepchenko B, Carson JH, Loew LM. A general computational framework for modeling cellular structure and function. Biophys J 1997;73:1135–1146. [PubMed: 9284281]
3. Slepchenko BM, Schaff JC, Macara I, Loew LM. Quantitative cell biology with the Virtual Cell. Trends Cell Biol 2003;13:570–6. [PubMed: 14573350]
4. Schaff JC, Slepchenko BM, Choi Y, Wagner JM, Resasco D, Loew LM. Analysis of non-linear dynamics on arbitrary geometries with the Virtual Cell. Chaos 2001;11:115–131. [PubMed: 12779447]
5. Moraru, II.; Schaff, JC.; Loew, LM. Think simulation – think experiment: the Virtual Cell paradigm. Proc. 2006 Winter Sim. Conf., IEEE, Inc; Piscataway, NJ. p. 1713-1719.
6. Novak IL, Gao F, Choi YS, Resasco D, Schaff JC, Slepchenko BM. Diffusion on a curved surface coupled to diffusion in the volume: Application to cell biology. Journal of Computational Physics 2007;226:1271–1290. [PubMed: 18836520]
7. Hucka M, et al. The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. Bioinformatics 2003;19:524–31. [PubMed: 12611808]
8. Lloyd CM, Halstead MD, Nielsen PF. CellML: its future, present and past. Prog Biophys Mol Biol 2004;85:433–450. [PubMed: 15142756]

9. Luciano JS. PAX of mind for pathway researchers. Drug Discov Today 2005;10:937–942. [PubMed: 15993813]
10. Le Novère N, et al. Minimum information requested in the annotation of biochemical models (MIRIAM). Nat Biotechnol 2005;23:1509–15. [PubMed: 16333295]
11. Slepchenko BM, Semenova I, Zaliapin I, Rodionov V. Switching of membrane organelles between cytoskeletal transport systems is determined by the regulation of the microtubule-based transport. J Cell Biol 2007;179:635–641. [PubMed: 17998399]
12. Ferziger, JH.; Peric, M. Computational methods for fluid dynamics. Springer; NY: 2002.
13. Gibson M, Bruck J. Efficient exact stochastic simulation of chemical systems with many species and channels. J Phys Chem A 2000;104:1876–1889.
14. Gillespie DT. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. J Comp Phys 1976;22:403–434.
15. Salis H, Kaznessis Y. Accurate hybrid stochastic simulation of a system of coupled chemical or biochemical reactions. J Chem Phys 2005;122:054103.
16. Salis H, Sotiropoulos V, Kaznessis Y. Multiscale Hy3S: Hybrid stochastic simulation for supercomputers. BMC Bioinformatics 2006;7:93. [PubMed: 16504125]
17. Schoeberl B, Eichler-Jonsson C, Gilles ED, Müller G. Computational modeling of the dynamics of the MAP kinase cascade activated by surface and internalized EGF receptors. Nat Biotechnol 2002;20:370–5. [PubMed: 11923843]
18. Blinov ML, Faeder JR, Goldstein B, Hlavacek WS. A network model of early events in epidermal growth factor receptor signaling that accounts for combinatorial complexity. Biosystems 2006;83:136–51. [PubMed: 16233948]
19. Hlavacek WS, Faeder JR, Blinov ML, Posner RG, Hucka M, Fontana W. Rules for modeling signal-transduction systems. Sci STKE 2006;344:re6. [PubMed: 16849649]
20. Blinov ML, Faeder JR, Goldstein B, Hlavacek WS. BioNetGen: software for rule-based modeling of signal transduction based on the interactions of molecular domains. Bioinformatics 2004;20:3289–91. [PubMed: 15217809]

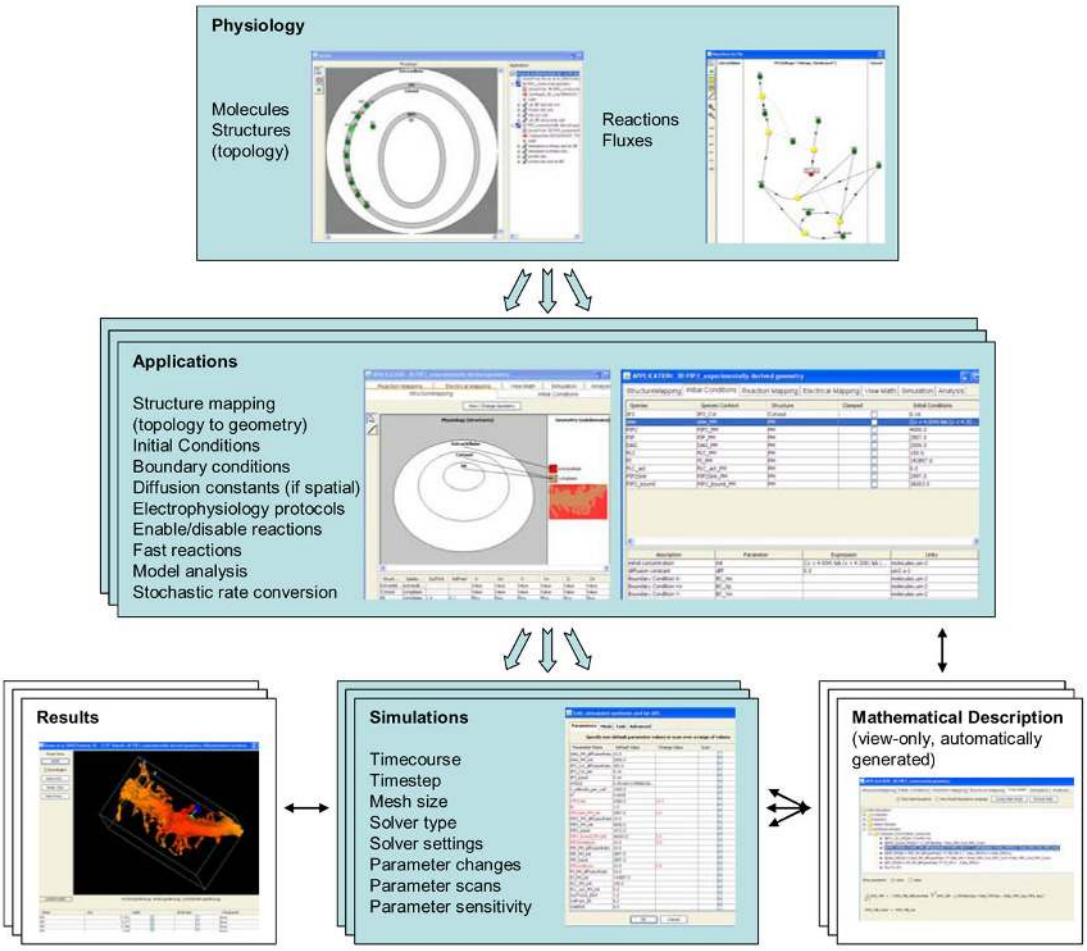

Figure 1.

Decomposing the modeling process in VCell: the relationship between the principal components of the BioModel workspace are shown. A single Physiology can be used in different contexts, as described by one or more Applications. Every Application is translated into a unique Math Description, which in turn is the basis for running one or more different Simulations for each of the Applications.

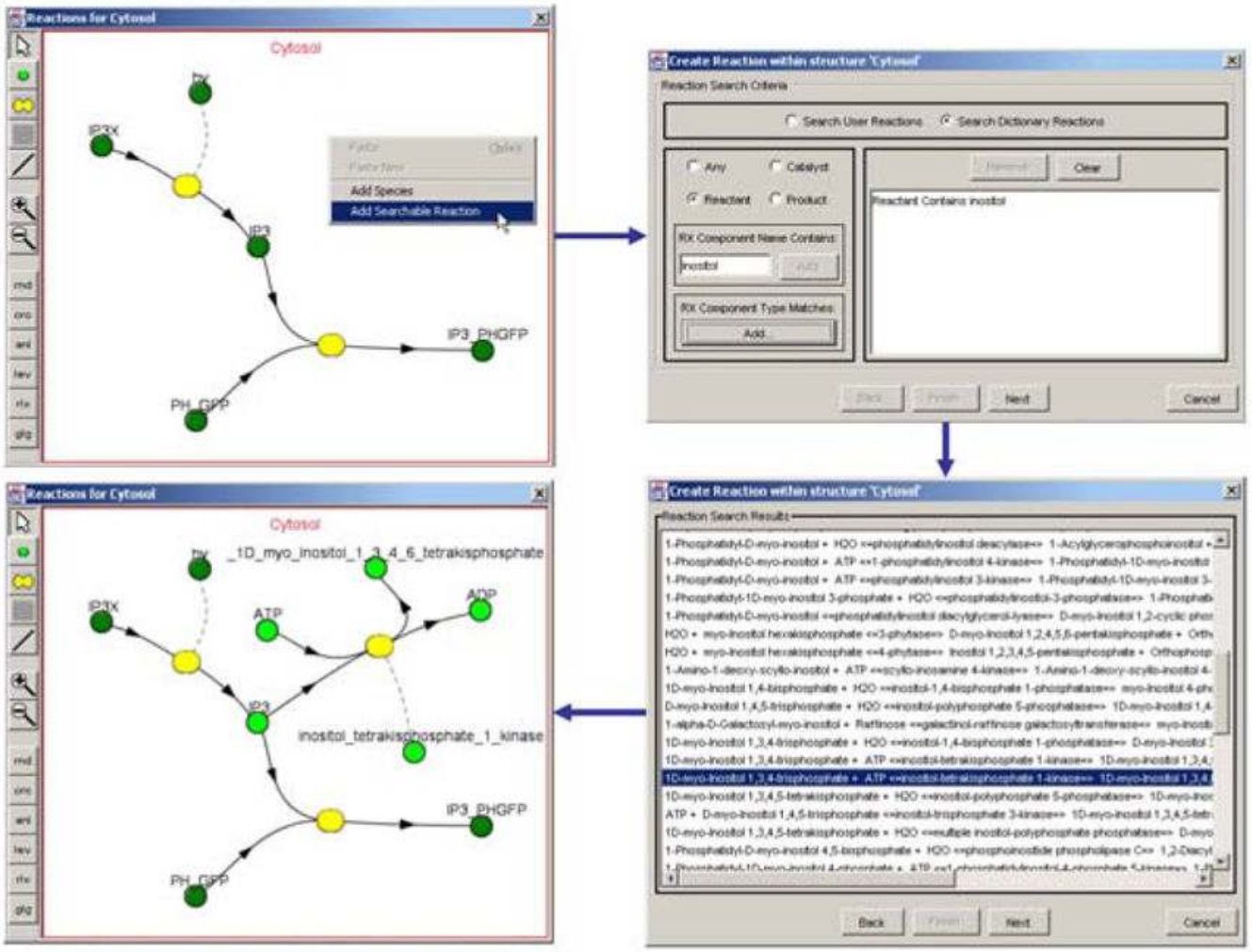

Figure 2.
Automatic import of reactions into VCell from the KEGG database.

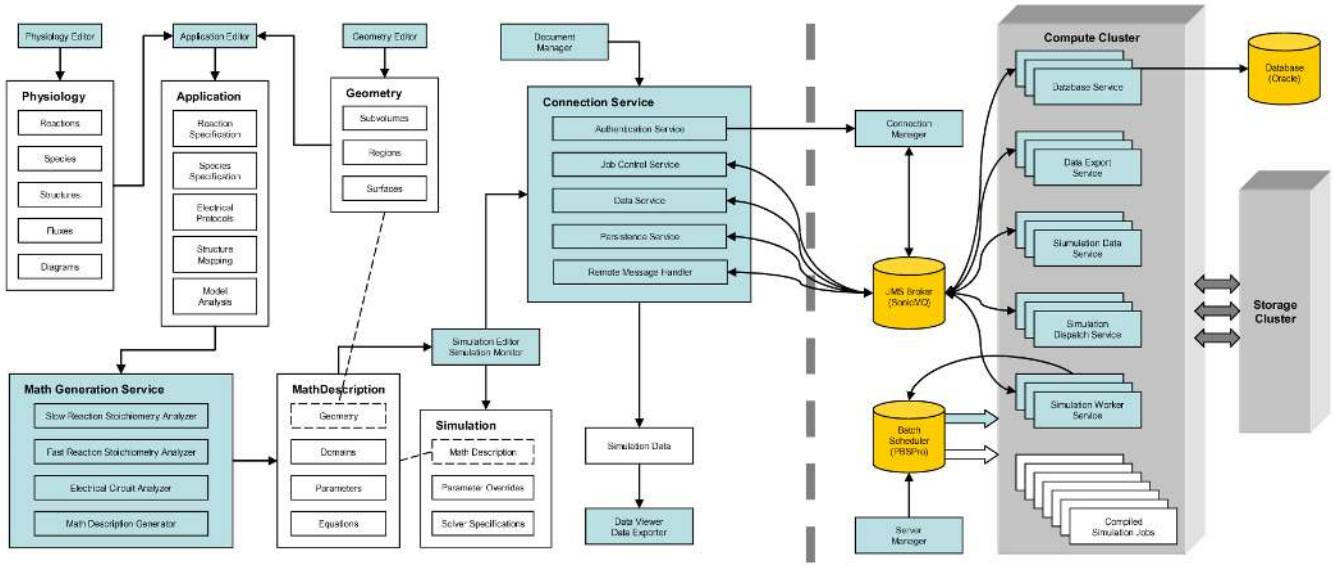

Figure 3.

The current VCell platform. The overall architecture of the web-based distributed system is shown, including the software components developed in house (white and blue) and third-party middleware (yellow). All the components to the left of the separator line are deployed as a single Java application on client (user's) computers.

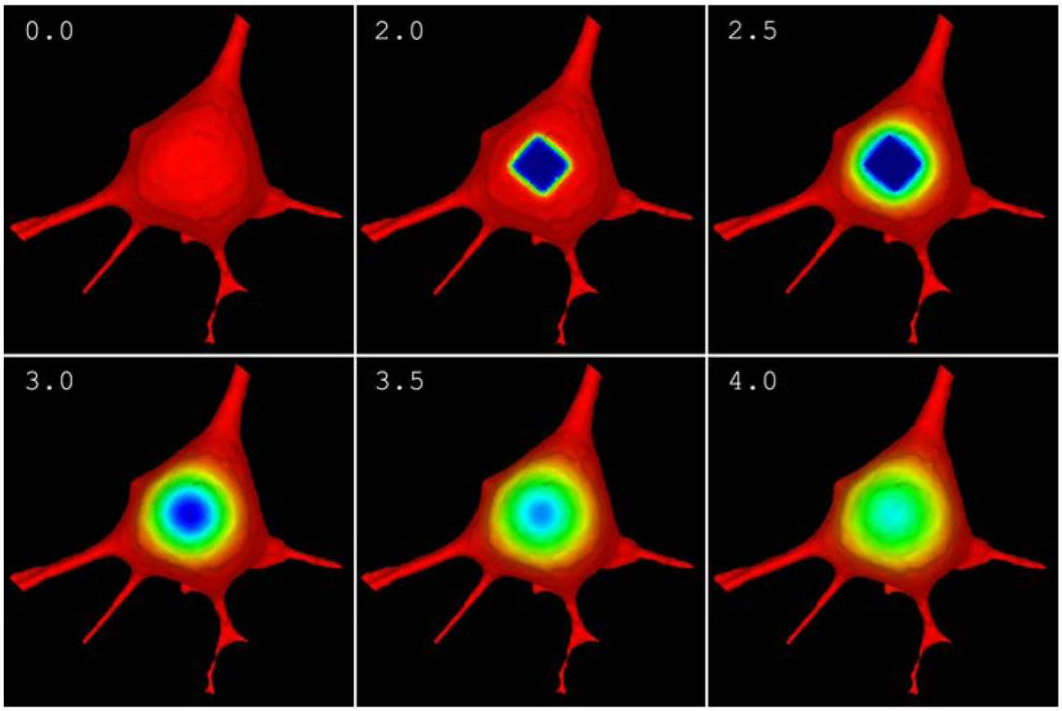

Figure 4.

A time sequence illustrating membrane diffusion of a molecular species (D=10μm2/s) on a 3-D cellular membrane after "bleaching" in a prototypical FRAP experiment. Each frame is labeled with time in seconds from beginning of experiment. Time 0.0 shows an evenly distributed concentration of a diffusible molecule (in red, high concentration) followed by a "bleach" event at time 2.0 creating a localized concentration deficit (in blue, low concentration) and subsequent diffusion at times 2.5–4.0 from un-bleached membrane to bleached area causing a concentration gradient (green, yellow) to form. In addition to illustrating the results of the VCell lateral diffusion algorithm, this also exemplifies the data display and export features.

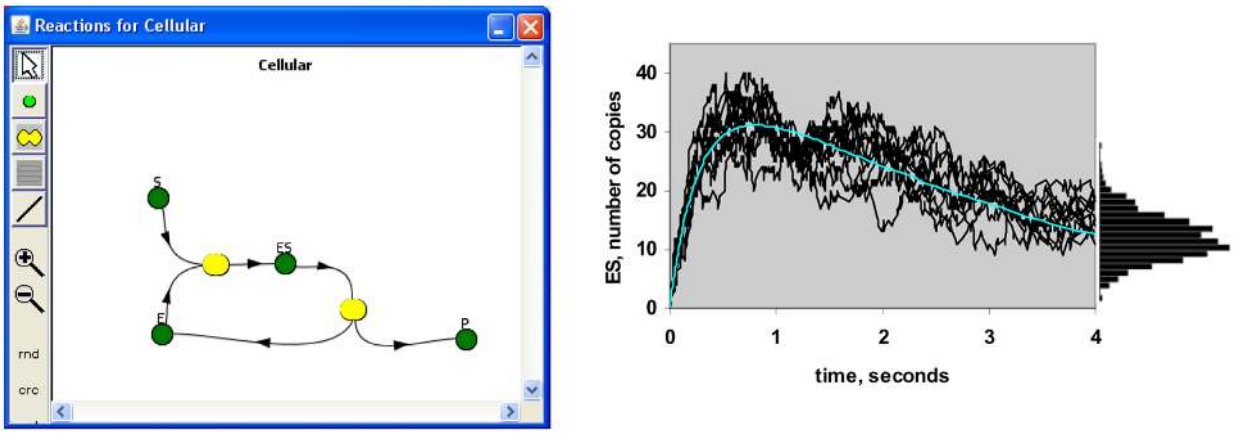

Figure 5.
Stochastic simulations with the Virtual Cell: a VCell model of an enzymatic reaction (left); a deterministic solution for the enzyme-substrate complex ES is superimposed over ten stochastic trajectories (right); histogram of the number of copies at t = 4.0 obtained from 1000 trials.