Services-Based Systems Architecture for Modeling the Whole Cell: A Distributed Collaborative Engineering Systems Approach

V. A. Shiva Ayyadurai

Abstract Modeling the whole cell is a goal of modern systems biology. Current approaches are neither scalable nor flexible to model complex cellular functions. They do not support collaborative development, are *monolithic* and, take a primarily manual approach of combining each biological pathway model's software source code to build one large monolithic model that executes on a single computer. What is needed is a distributed collaborative engineering systems approach that offers massive scalability and flexibility, treating each part as a services-based component, potentially delivered by multiple suppliers, that can be dynamically integrated in real-time. A requirements specification for such a services-based architecture is presented. This specification is used to develop *CytoSolve*, a working prototype that implements the services-based architecture enabling dynamic and collaborative integration of an ensemble of biological pathway models, that may be developed and maintained by teams distributed globally. This architecture computes solutions in a parallel manner while offering ease of maintenance of the integrated model. The individual biological pathway models can be represented in SBML, CellML or in any number of formats. The EGFR model of Kholodenko with known solutions is first tested within the CytoSolve framework to prove it viability. Success of the EGFR test is followed with the development of an integrative model of interferon (IFN) response to virus infection using the CytoSolve platform. The resulting integrated model of IFN yields accurate results based on comparison with

V. A. S. Ayyadurai (✉)
Department of Biological Engineering, Massachusetts Institute of Technology (M.I.T.),
77 Massachusetts Avenue, Cambridge, MA 02139, USA
e-mail: vashiva@mit.edu
URL: www.vashiva.com

V. A. S. Ayyadurai
Systems Biology Research Group, International Center for Integrative Systems (I.C.I.S.),
701 Concord Avenue, Cambridge, MA 02138, USA

Commun Med Care Compunetics (2011) 1: 115–168
DOI: 10.1007/8754_2010_1

Published Online: 16 November 2010

previously published in vitro and in vivo studies. A open web-based environment for collaborative testing and continued development is now underway and available on www.cytosolve.com. As more biological pathway models develop in a disparate and decentralized manner, this architecture offers a unique platform for collaborative systems biology, to build large-scale integrative models of cellular function, and eventually one day model the whole cell.

Keywords Complex systems · Systems biology · Distributed computing · Systems architecture · Whole cell modeling · Distributed Collaborative Engineering (DCE) · Molecular pathways · Biological networks · Collaboratory

1 Background

A grand challenge of systems biology is to *model* the whole cell. A model for the purpose of this discussion is defined to be a mathematical representation along with its implementation in software including any input data and documentation. A cell consists of a set of organelles. These organelles interact through the medium of molecular interactions to provide *cellular functions* such as protein synthesis, metabolism, apoptosis, or motility. Systems biology aims to develop a model of the cell by connecting the biochemical kinetics of these interactions at the molecular mechanistic level to derive the quantitative descriptions of higher level cellular functions [1–5].

1.1 The Complexity of Biology

Biology is a field based on experiments, not first principles (ab initio) such as physics or engineering. It is fundamentally an experimental science. Biologists do many experiments to understand genes, proteins, protein–protein interactions. One example of perhaps the largest experiments in biology is the Human Genome Project (HGP) begun in 1990 and completed in 2005. This effort resulted in the discovery of only 20,000–25,000 genes, far less than what was originally theorized [6]. More interesting is the discovery that this number of genes is in the same realm as that of the nematode *Caenorhabditis elegans* which has approximately 19,000 genes [7]. More recently, the genome of the starlet sea anemone—*Nematostella vectensis*, a delicate, few-inch-long animal in the form of a transparent, multi-tentacled tube - was sequenced and found to have 18,000 genes [8].

Human and a nematode (or sea anemone) have a similar number of genes, but a great difference in complexity of function as whole organisms. This contradiction has led scientists to conclude that perhaps the number of genes in the genome is not connected with the complexity of the organism. Much of an organism's complexity can be ascribed to regulation of existing genes by other substances

(such as proteins) rather than to novel genes [8]. The types and kinds of molecular interactions across the nucleus, cytoplasm and organelles, beyond the number of genes in the nucleus, may be the critical element in determining the difference between human and nematode, for example. This reasoning has led to an even greater activity to understand the structure of proteins (e.g. the product of genes) and protein–protein interactions.

1.2 Proteins and Protein–Protein Interactions

As of the writing of this document, approximately 30,000 proteins have been documented across various publications world wide [9]. Thousands of research teams across the world have contributed to these discoveries. Discovering the structure of just one protein is a difficult experimental effort. Such efforts are highly domain specific and one research team, for example, by itself may focus on understanding a small set of genes or the structure of a set of specific proteins or the interactions between certain types of proteins. The protein structures are determined from experiments using x-ray crystallography [10]. While new proteins are discovered each day, the crystal structure of most proteins is not known. In many cases, understanding just *one* protein structure requires the effort of not just this one laboratory's research team but the effort of multiple research teams spread across the world. New databases such as: HPRD, OMIM, PDB, Entrez Gene, HGNC, Swiss-Prot, GenProt are becoming repositories for storing protein structure as well as protein–protein interactions.

Approximately 40,000 protein–protein interactions are documented today and continue to grow. Each day, new protein–protein interactions are found. In addition, each day, updates of knowledge are made to existing protein–protein interactions. Experiments are used to derive such protein–protein interactions since these interactions cannot be derived from first principles. Moreover, sometimes, different research teams may get differing results for the same pair of protein interactions.

1.3 Biological Pathways

Biological pathways are networks of protein–protein interactions. Single protein–protein interactions can be combined to build biological pathways. Biologists, in addition to understanding the nature and function of genes, proteins and protein–protein interactions, also perform experiments to discover biological pathways. Today, approximately 60,000 biological pathway diagrams are recorded across a variety of databases including KEGG, Science STKE, Nature PID, BioPax, BioCarta and others. New biological pathways are being discovered each day.

A biological pathway consists of two elements: (1) molecules (also known as molecular species or species) and (2) interactions among those molecules. A biological pathway is visually represented using a "ball and stick" diagram as shown in Fig. 1. Each "ball" denoted by circles, rectangles or other geometric shapes represent the individual molecules. Each "stick" denoted by arrows or lines represents the molecular interaction. Because experiments not first principles determine the description of a biological pathway (e.g. which molecules will interact with another), biological pathways are constantly changing as new experiments reveal either changes in molecular species or nature of molecular interactions.

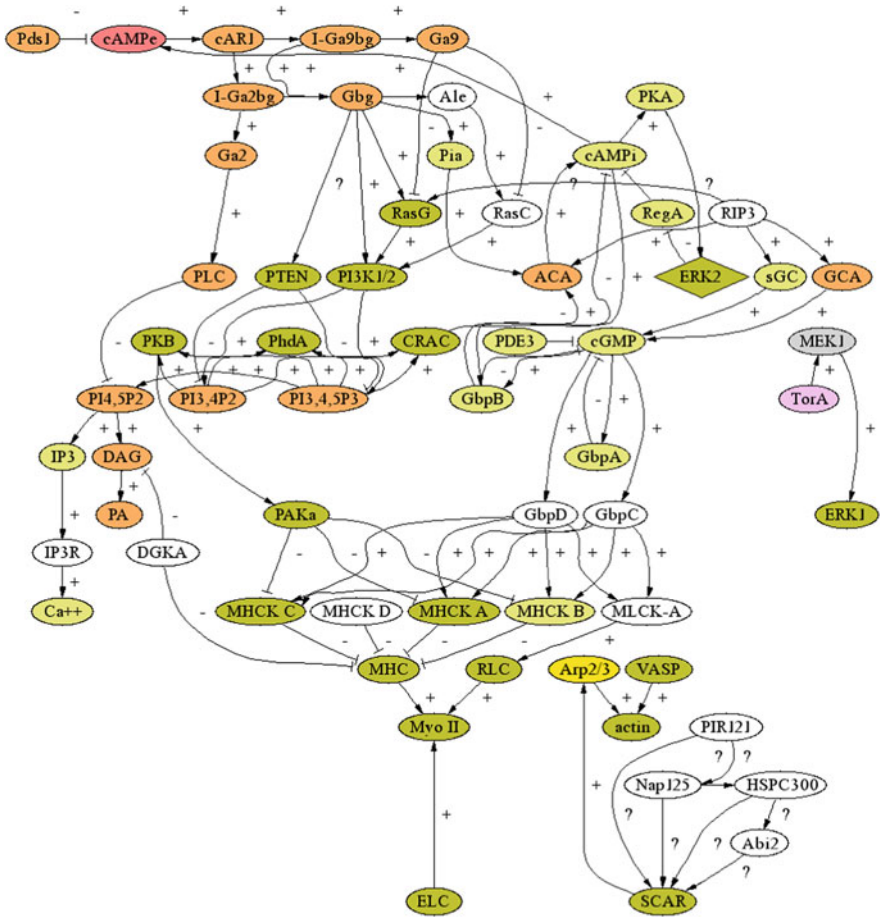

Fig. 1 Example of a biological pathway diagram containing many molecular species denoted by the geometric shapes of circles, ellipses and diamonds along with the multiple molecular interactions denoted by lines and arrows [86]

The biological pathway in Fig. 1 is the result of aggregating knowledge from nearly 35 different published articles. On average each biological pathway consists of approximately 5–15 molecular species and approximately 10–25 molecular interactions. The development of just this one biological pathway requires the integrated effort of multiple laboratories, spread across the world to construct and maintain. Elements of the biological pathway, the number of molecular species and the types of interactions, are subject to change based on new experimental results.

In summary, the development of each and every biological pathway is a highly collaborative effort, requiring the aggregation and integration of numerous experimental results, derived from the fundamental understanding of genes, proteins, and protein–protein interactions. Moreover, such a development effort, as will be discussed in forthcoming sections, is not linear, but cyclic, involving constant updates and refinements to the biological pathway, based on new experiments.

1.4 Systems Biology

Systems biology is a new field; however, building systems-level understanding of biology is not a new phenomenon. Over 6,000 years ago, many traditional systems of medicine including Siddha, Unani, Ayurveda and Traditional Chinese Medicine (TCM) proposed systems approaches to describing the whole human physiome [11, 12]. During modern times, starting in 1930s, with the concept of homeostasis [13] and biological cybernetics [14] attempts were made to understand biology from a systems level using the modern language of physics and control systems theory.

The discovery of the structure of DNA in 1953 [15] combined with recent high-throughput measurement techniques for imaging and quantifying molecular level interactions has enabled a completely new field of biology: systems biology. Systems biology aims to develop system-level understanding by connecting knowledge at the molecular level to higher level biological functions [16]. Such a goal was not possible before. Previous attempts at system-level approaches to biology, whether ancient or modern, were primarily focused on the description and analysis of biological systems, limited to the physiological level. Since these approaches had little to no knowledge of how molecular interactions were linked to biological functions, a *systems biology* of connecting molecular interactions to biological functions was not previously possible. Systems biology, therefore, is a new field of biology as it offers the opportunity, as never before in human history, to link the behaviors of molecules to the characteristics of biological systems. This new field will enable us to eventually describe cells, tissues, organs and human beings within a consistent framework governed by the basic principles of physics [17]. Systems biologists aim to link molecular-level interactions with cellular-level functions through quantitative modeling.

1.5 Biological Pathway Models

Over the past decade, new measurement techniques are enabling biologists to quantify the molecular concentrations and rates of molecular interactions within biological pathways. Such techniques are being used to transform diagrammatic representations of biological pathways to build biological pathway models.

Systems biologists convert "ball and stick" diagrams to *biological pathway models*, mathematical representations expressed in software program code along with the inputs and data needed to run the model. Figure 2 illustrates the high-level process by which a biological pathway is converted to a biological pathway model. There are approximately 300 published biological pathway models today.

The software program source code of these models may be represented in different formats: MATLAB, C++, C, SBML, CellML, etc. Different mathematical representations including ordinary differential equations (ODE's), Boolean Networks, Stochastic approaches, analytical functions, etc. are used to specify these models. The internal parameters for these models, such as kinetic rate constants, for example, are also determined through experimentation. Maintaining just one biological pathway model is a complicated task since the biological pathway models and the internal parameters are constantly updated based on changes to the biological pathway diagrams (e.g. based on new experiments). There are different emerging repositories for hosting biological pathway models including BioModels.Net and CellML.Org [18, 19].

From these repositories, one can download a biological pathway models and execute them on a local computer. Figure 2 illustrates the transformation of the biological pathway diagram, on the left hand side, to a *black box* biological pathway model, on right hand side. This black box has inputs, being the species concentrations of the molecular species at time $t = n$, and outputs being the

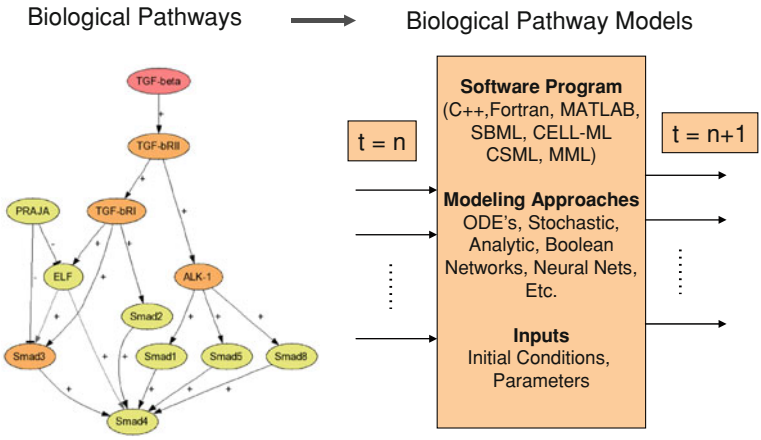

Fig. 2 Systems biologists work to convert biological pathway diagrams to biological pathway models

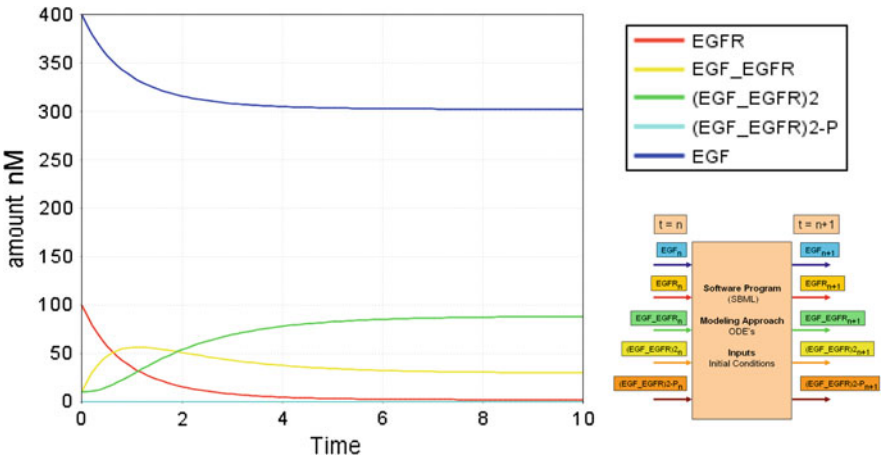

Fig. 3 Results from the execution of the biological pathway model

species concentrations of the molecular species at time $t = n + 1$. The internals of the black box contain the software code and the mathematical representation. In this case, the mathematical representation is an ODE and the software code is in SBML. However, the mathematical representation and the software code could be in any format as described earlier. Execution of this biological pathway model will yield results as shown in Fig. 3.

The results in Fig. 3 are the time varying changes in species concentrations. Along the x-axis is time and along the y-axis is the species concentration in nM (nano-molar). Such a biological pathway model serves to provide a quantitative and predictive capability to describe the interactions of five molecular species. Each biological pathway model is treated as a black box, having a defined set of input species and the same defined set of outputs, the values of which are the species concentrations. The construction and validation of such biological pathway models remains a tedious and time-consuming process mainly due to the experimental effort required to determine the many internal parameter values.

Most biological pathway models are developed, used for a single application by a single developer, and then forgotten. One can only imagine how many biological pathway models, for example built in MATLAB, and never published are located in some file folder, in some unknown computer, developed by some graduate student. Therefore, considering all the hard work that goes into developing a model, it is rather surprising that so little attention is paid to the presentation and conservation of existing models [20].

Systems biologists are attempting to create reusable components of biological pathway models by constructing online repositories to offer an archive and curated repository. In addition to supporting the conversion of the diagrammatic representation of biological pathways to biological pathway models, systems biologists also aim to build *larger* biological pathway models by integrating *smaller* biological pathway models. The purpose of such effort is to gain new insights on

cellular functions not possible from experimental research. Currently, there are approximately 5–10 such integrated biological pathway models. There are three main reasons why there are such a low number of integrated biological pathway models.

First, the individual biological pathway models are in different formats. Second, understanding any one model requires a great deal of domain specific knowledge and expertise. Third, the primary method of integration involves merging the source codes of each biological pathway model into one large source code. Because of these reasons, it is very time-consuming and expensive to integrate biological pathway models. Maintenance of the resulting integrated model is also very difficult since the integrated model can become invalid as it has a "half-life." New proteins, protein–protein interactions and new parameters (e.g. rate constants) in any one of the individual biological pathway models are being discovered and/or updated constantly. Integrating models can also become especially difficult, within the current method, if the source code for any one particular biological pathway model is not publicly available. This would require one to recode that entire model's source code from scratch.

2 Motivation

Figure 4 illustrates the path to modeling the whole cell and summarizes the concept from the previous sections.

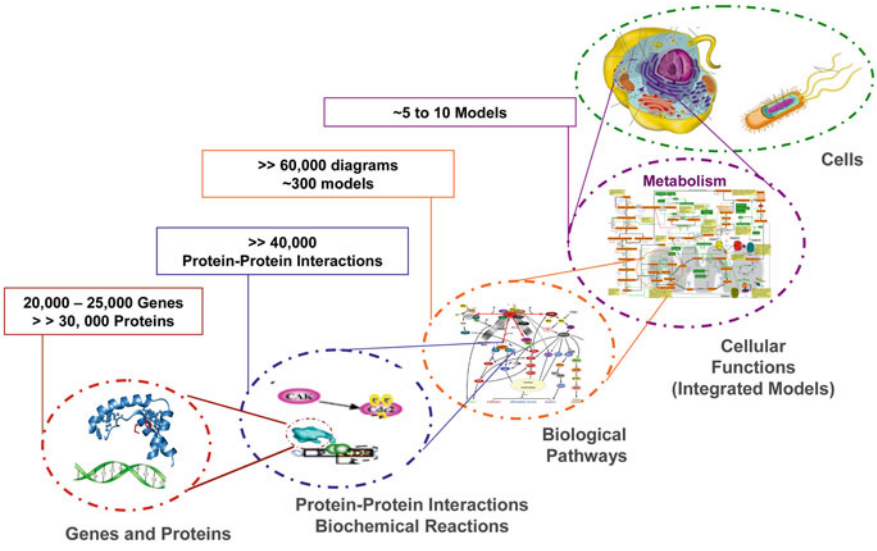

Fig. 4 Summary of the development path towards whole cell modeling

This figure presents four major steps to modeling the cell. First, the understanding of genes and proteins are used to build an understanding of protein–protein interactions. Second, these protein–protein interactions are networked to create biological pathways. Third, the integration of biological pathways serves to describe cellular functions. Fourth, and finally, the integration of cellular functions serves to model the whole cell.

Currently, as shown in Fig. 4, there are only 5–10 such integrated models of cellular function. It is expected that the number of biological pathway models, currently numbering approximately 300, will grow; however, the step in developing larger integrated models is severely limited by the time consuming and expensive effort, for the three reasons highlighted earlier. The discussion below provides greater insights on the efforts needed to build biological pathway models.

2.1 Development of Biological Pathway Models

The process to create and maintain a particular biological pathway model is an iterative process of manipulating, measuring, mining and modeling as shown in Fig. 5. The two major areas are experimentation and modeling. Systematic experiments involve manipulation and measurement. Manipulation involves modifying an existing biological system. Measurement involves collection of data from that manipulated biological system. Quantitative modeling involves both mining and modeling. Mining enables the identification of underlying relationships in large datasets.

These relationships can be used to create predictive mathematical models. Biologists, working in highly domain specific areas, perform systematic experiments, using a range of advanced measurement devices quantify the molecular

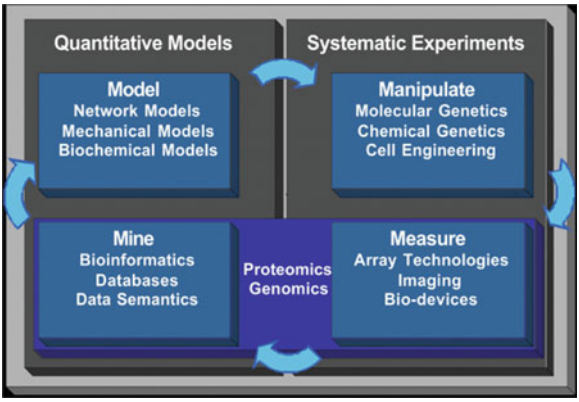

Fig. 5 The four M's of systems biology [87]

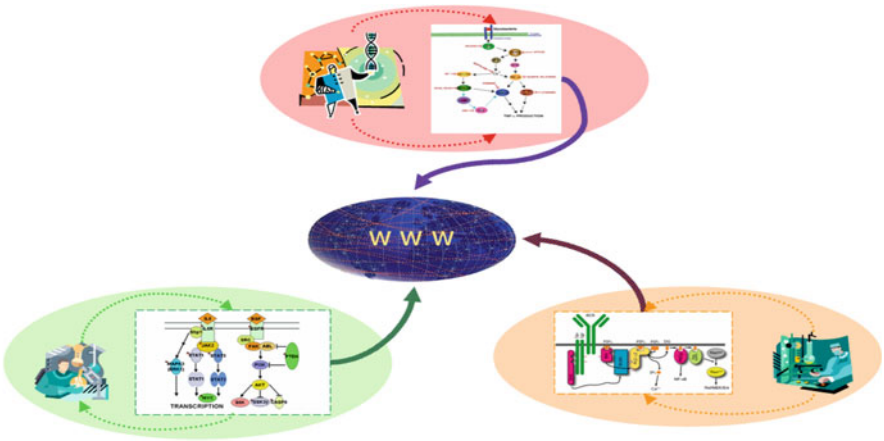

Fig. 6 Scenario of three research teams performing systematic experiments to produce biological pathway models, which are published and Complexity of Integrating Multiple Biological Pathway Models

concentrations and dynamics of molecular interactions. Data mining and modeling efforts are used to refine their conclusions. Biological pathway models are developed and refined through this constant and arduous iterative process. The World Wide Web (WWW) offers a vehicle for scientists to more easily share and publish their biological pathway models. There are thousands of such biological pathway models being published and refined each day by teams of biologists worldwide. Figure 6, for example, illustrates three different research teams, spread across the globe, performing the iterative process of systematic experiments and modeling to produce biological pathways, which are made available and published over the WWW.

Given the decentralized nature of these efforts, the source code of any one biological pathway model may be written and stored in a variety of software programming languages, may be publicly accessible or proprietary. A particular source code is typically built and tested on a particular computer hardware platform, and multiple teams may be involved in maintaining that one source code.

As the number of biological pathway models and our ability to accurately model any one biological pathway model increases, the challenge becomes how to integrate an ensemble of biological pathway models to build more *complex* models of cellular function. As an aside, this term complex needs to be discussed prior to proceeding. Any one biological pathway model within an integrated model may contain hundreds of species and a set of hundreds of resulting mathematical equations describing those interactions; however, this does not mean the model is necessarily complex.

For example, on a personal computer, super computer or even powerful handheld devices, hundreds of simultaneous differential equations can be solved; however, this does not mean that the model we solve is complex just because it has many

equations. A complex system, on the other hand, may be complex even if the number of equations is small and apparently simple if the individual elements of the system have their own unique dynamic behavior. Such a system is said to be complex if it has multiple elements, which reveal different dynamic properties. This may occur, for example, when all system elements are continuous with concentrated parameters, but the model includes very fast and very slow parts [21]. Another example is a system where discrete parts interact with continuous sub-models of different speed and different kind such as an electronic circuit that contains integrated circuits as well as electro-mechanical parts such as relays and motors [22]. In other words, the model complexity has little to do with the model size.

Let us now consider a complex biological system: the interferon (IFN) response to virus infection. This integrated system involves various biological pathways, each of which is a unique domain of knowledge and effort for modeling. Figure 7 illustrates the four key biological pathways involved in this complex integrated system. Different research teams worldwide develop each pathway. At the lower left of this figure is the *virus infection pathway*. This biological pathway model simulates the virus infection of a cell and results initially in the up regulation of IFN-Beta, a critical signaling protein; and later on, results in the up regulation of IFN-Alpha, another signaling protein.

A second biological pathway model is *IFN receptor signaling*, as shown in the upper left. This biological pathway model represents the interactions of IFN proteins, either IFN-Alpha or IFN-Beta, landing on cell receptors to trigger the activation of other proteins within the cell's cytoplasm to up regulate IRF-7, an interferon regulatory factor. A third pathway is the IFN *amplification cycle*, as shown in the upper right. This biological pathway model simulates the production of increased amounts of both IFN-Alpha and IFN-Beta, which results from the by-product of virus infection with IRF-7. A fourth pathway is SOCS1 *regulation*, shown in the lower right. This pathway serves to regulate the production of IFNs by inhibiting the IFN receptor-signaling pathway.

The ensemble of all of the biological pathways depicted in Fig. 7, if integrated, can provide an integrative model of IFN response to virus infection. Each biological pathway is a contribution of different research teams across three continents of North America, Asia, Europe, and four countries: China, Russia, United States and Japan. Any individual biological pathway model within this ensemble does not have thousands of equations but the activity to integrate these four models to create one new model is unequivocally a complex problem for a number of reasons. First, based on the literature, not all of the biological pathway models depicted in Fig. 7 have source codes for their models. Second, the biological pathway models were built using different software programming languages. Third, each team developed their biological pathway models on different hardware platforms. Fourth, each of the biological pathway models exhibits different dynamic properties (e.g. different time scales). To integrate these four models is a complex problem. Such a problem is representative of most cellular functions that involve multiple biological pathways, which need to be connected to build a larger model.

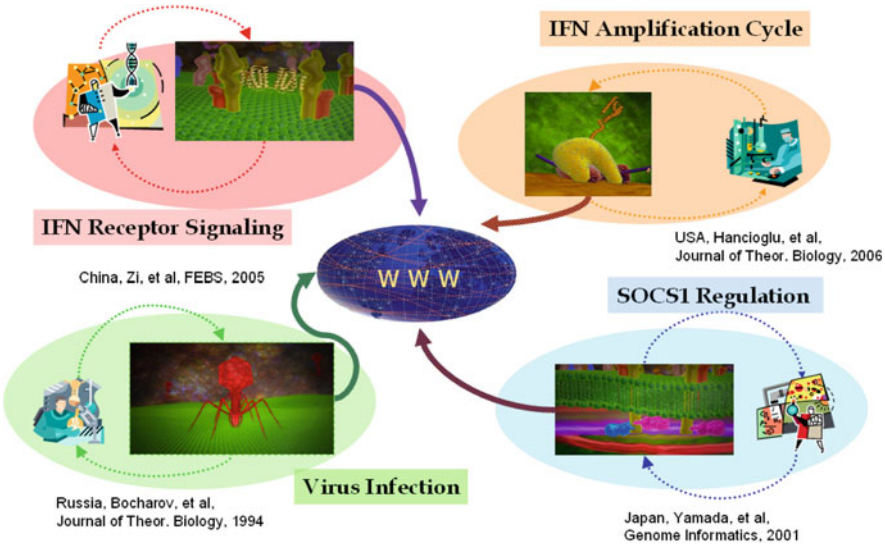

Fig. 7 Scenario of scientists performing systematic experiments to produce biological pathway models, which are published and made available over the WWW

2.2 Complexity of Maintaining an Integrated Model

The above discussion outlined the complexity of creating an integrative model of cellular function from combining various biological pathway models. This complexity is only one part of the problem. Another critical function is: the *maintenance* of the resulting integrated model. In the scenario described in Fig. 7, different teams worldwide develop each biological pathway model. Each team provides a particular domain of knowledge. As discussed in a previous section, the development of any one particular biological pathway model is an iterative process of systematic experimentation combined with quantitative modeling, both supporting each other. The reality is this: *Any one biological pathway model is constantly undergoing refinement.*

This means that the maintenance of a combined set of biological pathway models can be nearly impossible if there is no easy mechanism to receive and incorporate the updates from each biological pathway model; otherwise, the integrated model's accuracy is only good as its latest update. Since each model is developed in different mathematical and software representations, the integration and maintenance is made even more complicated.

2.3 Integration of Biological Pathway Models

Thus far, we have used the term, *integrated model* without formally defining it. Moving forward in our discussion, an integrated model refers to a group of

biological pathway models executing together that have the ability to affect each other's computations. Based on the previous discussion, the question arises as to how does one effectively integrate an ensemble of biological pathway models and maintain them to ensure reliability.

One approach is to avoid the problem entirely and take a completely different approach: develop from scratch an entirely new biological pathway model that encompasses the multiple phenomena across each individual biological pathway model. In essence, create one large biological pathway model. The time and expense, however, involved in developing such a model is prohibitive. The design and implementation of a model requires a combination of software engineering skill and domain expertise. In addition, it involves an extensive amount of verification and validation, requiring the iterative process of systematic experimentation and quantitative modeling as previously discussed.

Another approach is to acquire and reuse the source codes from each biological pathway model, and merge them through some mechanism to create one large biological pathway model. While this process may be appearing easier than the previous one, it may not be so. The reuse of software is a key principle of software engineering and is usually achieved by developing a set of simple components, or modules that can be combined in different ways to create more complex components. Ideally scientists would connect their biological pathway models by integrating the existing source codes, treating them as modular pieces that can be easily and quickly plugged together. This, however, can be a very difficult task when the legacy source codes themselves may be poorly understood, and more than likely, were not originally designed to be integrated, and may be written in different software programming languages for different hardware computing platforms.

Despite the potential benefit of building new models from existing ones, integrating pathway models is not a common practice in systems biology community because of the difficulties inherent in working with source codes. Reusing source code in general is difficult for many reasons. Not only are programs difficult to comprehend (a necessary part of any software reuse) but the task of identifying useful source code fragments and integrating these source code artifacts that were not designed for reuse is challenging [22–24]. This is especially true for source codes written in unstructured languages and languages that make extensive use of global data e.g. Fortran [22]. Reusing source code is particularly difficult because biological pathway models are a unique class of computer programs whose design and use is intertwined with a great deal of domain-level theory outside the model code itself [25].

In short, the amount of time and expense required to understand the legacy source codes of each individual biological pathway model, prior to merging them may be more costly than starting from scratch. Moreover, once the integrated model is created, the next problem becomes the complexity of maintaining the resulting integrated model, since changes will no doubt take place in the originating biological pathway model's source codes, as they are refined and enhanced, through systematic experiments and modeling.

Perhaps a better way is to integrate biological pathway models in a decentralized manner such that the integrated model functions as one whole, while any of its component biological pathway models can continue to be owned and maintained by its original authors. If this approach is taken, effort will be required to build a new messaging architecture that enables disparately produced biological pathway models to interface, obviating the need to explicitly integrate the source codes. Such an infrastructure would allow scientists to quickly prototype those integrated models. This thesis is motivated to create such a computational infrastructure to integrate biological pathway models.

2.4 Research Question

The above sections should have clarified to the reader that biology is fundamentally an experimental science. The development and understanding of genes, proteins, and protein–protein interactions, or just any one element along the path to modeling the whole cell is difficult, time-consuming and complex, requiring the collaborative effort of multiple teams of scientists worldwide. The question then becomes: *How can we build larger models from smaller models in a scalable framework to support whole cell modeling given the reality of biology—an experimental science?*

3 Prior Work

Integrating biological pathway models is becoming an important area of research for advancing the field of systems biology. In the next section, we review the key factors that are driving this need and some recent efforts to create integrative models of biological systems. In the third section, we survey general computational architectures for integrating models. The fourth section specifically surveys the current approaches for integrating biological pathway models in the field of systems biology. We conclude this chapter by summarizing the current approaches in tabular form. We also provide a discussion on the critical weakness limiting the integration of biological pathway models.

3.1 Movement to Integrate Biological Pathway Models

There is a worldwide movement in the computational systems biology community to find powerful ways to integrate the growing number of biological pathway models. This movement is being driven by a transition from diagrammatic

representation of pathways to quantitative and predictive mathematical models, which span time-scales, knowledge domains and spatial-scales [26–28]. This transition is being accelerated by high-throughput experimentation which isolates reactions and their corresponding rate constants [1]. Vast amounts of information is now available at the level of genes, proteins, cells, tissues and organs, which requires the development of mathematical models that can define the relationship between structure and function at all levels of biological organization [29].

Systems biology aims to provide a comprehensive quantitative analysis of the manner in which all the components of a biological system interact functionally over time [30]. Such an objective is pursued by an interdisciplinary team of investigators [31]. A significant computational challenge is how we can integrate such sub-cellular models running on different types of algorithms to construct higher order models [32].

Biological pathways, including metabolic pathways, protein interaction networks, signal transduction pathways, and gene regulatory networks, are currently represented in over 220 diverse databases. These data are crucial for the study of specific biological processes, including human diseases. Standard exchange formats for pathway information, such as BioPAX, CellML, SBML and PSI-MI, enable convenient collection of this data for biological research, but mechanisms for common storage and communication are required [33]. However, one the greatest challenges in establishing this systems approach are not biological but computational and organizational [34]. The critical need across all domains of molecular and cell biology is to effectively integrate large and disparate data sets [35].

Vigorous interest in understanding the dynamic aspects of cellular networks is also another driver in the development of integrative techniques for biological pathway models [36, 37]. Such explorations could provide insight into the mechanisms of healthy and diseased cells, as well as a better understanding of how system-level or whole-cell properties emerge from intracellular interactions of molecular components [38]. Moreover, understanding dynamics at the global network level seems to be now a reachable goal, which has motivated the growth of systems biology. Also, it is commonly admitted that the study of the network dynamics is able to enlighten the function of genes and groups of genes [39].

A central question now confronting virtually all fields of biology is whether scientists can deduce from this torrent of molecular data how systems and whole organisms work. All this information needs to be sifted, organized, compiled, and—most importantly— connected in a way that enables researchers to make predictions based on general principles [40]. Mapping protein interactions and transactions (such as phosphorylation, ubiquitination, and degradation) within a cell or organism is essential to developing a molecular understanding of physiology. Over the past decade, protein interaction mapping has evolved from low throughput manual screens to systematic interrogations of entire proteomes [41]. Reconstitution of biochemical and biophysical processes from ‘minimal systems’ of proteins has built confidence those top-down and bottom-up approaches to biology meet somewhere in the middle.

Systems biology has sought to integrate these results and data to reverse-engineer an understanding of biological network function and dynamics. The infrastructure for storing and disseminating information on biological systems, and for modeling them, has grown concurrently. In turn, this allows the rapid access and cross-comparison of information that is critical to establishing data quality and creating interoperability standards that will enable biologists to leverage their efforts and build scalable systems [42].

3.2 Existing Methods

Two critical elements need to be carefully assessed when selecting a modeling approach for any dynamic system: (1) the *level of abstraction* and (2) the *methodology of implementation*. In determining which level of abstraction and which methodology of implementation to use, the notions of *tractability*, *scalability* and *accuracy* are some of the important selection criteria, among others. Tractability is measured by the time and expense needed to design, implement, and test and assess the viability of the modeling approach. Scalability is determined by the ease with which the modeling approach can integrate new components at a particular level of abstraction. Accuracy is determined by the ability of the modeling approach to yield results, which match that observed in nature. Different users of the modeling approach will have these and other qualitative criteria in determining which modeling approach are the most optimal for their particular needs.

Currently there are two existing methods towards building whole cell models. The first method proposes to use first principles (ab initio) and large-scale computing, as has been applied in other fields such as climatology, particle dynamics, etc. to build a whole cell model. The second method involves downloading and accessing existing models and manually integrating their source codes together by hand to create one monolithic software program: *the monolithic approach*. A variation on this approach is to use semi-automation tools that help one to automatically read and integrate source codes together to create one monolithic software program.

3.3 First Principles—Ab Initio

There are various choices for which level of abstraction to use in modeling the cell. In Fig. 8 four potential abstractions are illustrated: quantum, atomic, biological pathways, and organelles. In the first principles approach, one could start at the atomic level of abstraction and solve the time-dependent Schroedinger equation (quantum mechanics, QM) to quantify the dynamics of the whole cell [43]. Such an approach would lead to a detailed understanding of the role that atomic level interactions play in determining the fundamental biochemistry of the whole

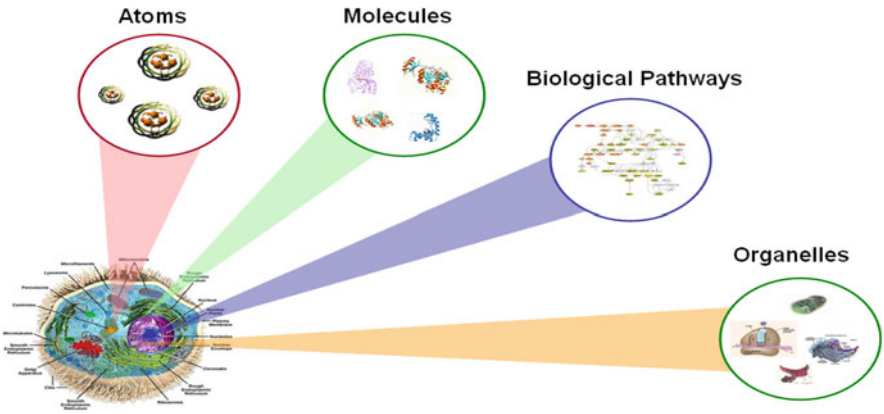

Fig. 8 Various levels of abstraction in modeling the whole cell

cell. The difficulty in using QM, for example, is that the vast range of length and time scales, from a nitrous oxide molecule to an organelle, makes the QM solution both impractical and useless [43].

It is impractical since there are too many degrees of freedom describing the motions of the electrons and atoms, whereas in the functioning of a cell it may only be the rate of transfer across some membrane. The complexity of QM limits its applications to systems with only 10–200 atoms (depending on the accuracy), leading to distance scales of less than 20 Angstroms and time scales of femtoseconds. Even the simplest protein in the cell contains over 1000 atoms. While the atomic level abstraction offers high level of accuracy, the level of abstraction is not scalable as the addition of each new atom increases the computational needs exponentially. This potential solution is also impossible, today, as the computing power needed to model the cell using this level of abstraction does not exist.

The first principle method therefore attempts to leap frog some of the steps in modeling the cell by using the laws of physics to model the cell versus experiments, which are the basis of biology. Another choice of abstraction in using the first principles method is at the molecular level, where Newton's equations are used rather than Schroedinger's to model molecular dynamics (or MD). Where in QM the solution is determined by averaging over the scale of electrons to describe the forces on atoms, in molecular dynamics (MD), one calculates an average over the dynamics of atoms to describe the motion of large molecules. While MD provides the ability to predict the dynamic interaction of molecules ab initio in an accurate manner, this level of abstraction is neither tractable nor scalable for modeling the whole cell since biological molecules such as proteins have far more atoms, degrees of freedom and numbers of states not encountered in other engineering fields where the species and interactions are well-defined [44]. We consider the simple problem of modeling the interaction of two proteins to demonstrate the intractability of using MD for whole cell modeling.

Consider the interaction of two proteins A and B to form the complex AB. We assume that each protein has 100 amino acid residues and each residue has three states (e.g. alpha, beta and other). In MD to model this simple interaction, two key calculations need to take place: 1. Thermodynamic, and 2. Kinetic in order to find the most likely transition state of protein A and protein B combining to produce complex AB [45]. For the thermodynamic calculations, MD requires the calculation of thermodynamic properties such as entropy which requires the need to evaluate all the possible states and associated probabilities of protein A, protein B and the complex AB. Protein A will have 3^{100} possible states (100 residues, each of which can be in three possible states), protein B will have 3^{100} possible states, and the complex AB can have up to 3^{200} (since AB is a combination of A and B) possible states. Just performing this calculation to determine the states and associated probabilities using modern computers is impossible and therefore intractable.

The kinetic calculation requires the identification of an appropriate reaction coordinate by computing the relative energies (or probabilities) for all the conformations along this reaction coordinate. In this case, it will require determining all the possible conformations of A and B that are at the energy of the activated complex, denoted by A' and B', (a higher energy than the energy of A and B); then determining all the possible conformations of A and B within complex AB stage, denoted by A'' and B'', (at an energy lower than that of A and B); and then finally determining all the conformations of AB complex. The kinetic calculation then attempts to link the most probable conformations starting with A and B, then A' and B', then A'' and B'', and finally the AB complex to calculate the reaction coordinate. These sets of multiple calculations using atom-by-atom MD to determine the reaction coordinate to solve even the simple molecular interaction of two proteins is intractable using modern day computers [45]. Moreover, this level of abstraction is not scalable as the number of interactions, number of proteins, and number of atoms per protein increases. In summary, while MD has powerful applications for determining protein conformations, it is not viable for whole cell modeling where hundreds of thousands of proteins are involved in millions of molecular interactions. Therefore, neither QM nor MD, using the laws of physics, offers tractable approaches to modeling the whole cell.

3.4 Biological Pathways as Modules

Another approach towards modeling the cell is to consider biological pathways as being the elemental modules from which complex cellular functions and the whole cell can be modeled. In this section, we present various viewpoints in the existing literature that supports such an approach. Biological systems are thought to have large number of parts almost all of which are related in complex ways [46]. Functionality emerges as the result of interactions between many proteins relating to each other in multiple cascades and in interaction with the cellular environment.

By computing these interactions, it can be used to determine the logic of healthy and diseased states [47]. One way to model the whole cell is through bottom up reconstruction. Such bottom up reconstruction, for example, of the human metabolic network was done primarily through a manual process of integrating databases and pathway models [48].

It is possible, for example, to regard signaling networks as systems that decode complex inputs in time, space and chemistry into combinatorial output patterns of signaling activity [49]. By treating biological pathways as modules our minds can still deal with the complexity. In this way, accurate experimentation and detailed modeling of network behavior in terms of molecular properties can reinforce each other [50]. The goal then becomes that of linking kinetic models on small parts to build larger models to form detailed kinetic models of larger chunks of metabolism, and ultimately of the entire living cell [20]. The value for integrating pathways is that it was found that the integrated network shows emergent properties that the individual pathways do not possess, like extended signal duration, activation of feedback loops, thresholds for biological effects, or a multitude of signal outputs [51]. In this sense, a cell can be seen as an adaptive autonomous agent or as a society of such agents, where each can exhibit a particular behavior depending on its cognitive capabilities.

Unique mathematical frameworks will be needed to obtain an integrated perspective on these complex systems, which operate over wide length and time scales. These may involve a two-level hierarchical approach wherein the overall signaling network is modeled in terms of effective "circuit" or "algorithm" modules, and then each module is correspondingly modeled with more detailed incorporation of its actual underlying biochemical/biophysical molecular interactions [52]. The mammalian cell may be considered as a central signaling network connected to various cellular machines that are responsible for phenotypic functions. Cellular machines such as transcriptional, translational, motility, and secretory machinery can be represented as sets of interacting components that form functional local networks [53].

As biology begins to move into the ''postgenomic'' era, a key emerging question is how to approach the understanding of how complex biological pathways function as dynamical systems. Prominent examples include multi-molecular protein ''machines,'' intracellular signal transduction cascades, and cell–cell communication mechanisms. As the proportion of identified components involved in any of these pathways continues to increase, in certain instances already asymptotically, the daunting challenge of developing useful models—mathematical as well as conceptual—for how they work is drawing interest [54].

Multi-scale modeling is essential to integrating knowledge of human physiology starting from genomics, molecular biology, and the environment through the levels of cells, tissues, and organs all the way to integrated systems behavior. The lowest levels concern biophysical and biochemical events. The higher levels of organization in tissues, organs, and organism are complex, representing the dynamically varying behavior of billions of cells interacting together [55]. Biological pathways can be seen to share structural principles

with engineered networks, along with three of the most important shared principles, modularity, robustness to component tolerances, and use of recurring circuit elements. [56].

An important attribute of the complexity pyramid is the gradual transition from the particular (at the bottom level) to the universal (at the apex) [57, 58]. Others have recognized that one can build cellular-like structures from a bottom up approach [59]. Integrated models would represent the most compact, unambiguous and unified form of biological hypotheses, and as such they could be used to quantitatively explore interrelationships at both the molecular and cellular levels. [60]. At this time, for instance, the computational function of many of the signaling networks is poorly understood. However, it is clear that it is possible to construct a huge variety of control and computational circuits, both analog and digital from combinations of the cascade cycle [61].

3.5 Integrative Modeling Efforts

As discussed in the previous section, systems biology is determined to find new ways to integrate biological pathway models to build larger systems. Neuroscience, for example, seeks such integration of computational models for better understanding of different signaling pathways in neurons [62].

In the area of metabolism, researchers have created comprehensive mathematical descriptions of the cellular response of yeast to hyperosmotic shock. Their model integrates a biochemical reaction network comprising receptor stimulation, mitogen-activated protein kinase cascade dynamics, activation of gene expression and adaptation of cellular metabolism with a thermodynamic description of volume regulation and osmotic pressure [63].

The IUPS Physiome Project is an international collaborative open source project intended to provide a public domain framework for computational physiology, including the development of modeling standards, computational tools and web-accessible databases of models of structure and function at all spatial scales and across all organ systems [64].

For the first time, kinetic information from the literature was collected and used to construct integrative dynamical mathematical models of sphingolipid metabolism [65]. In another example, a model of 545 components (nodes) and 1259 interactions representing signaling pathways and cellular machines in the hippocampal CA1 neuron were combined. Using graph theory methods, this effort analyzed ligand-induced signal flow through the system. Specification of input and output nodes allowed them to identify functional modules. Networking resulted in the emergence of regulatory motifs, such as positive and negative feedback and feed-forward loops, that process information [53].

Oda et al. [66] have developed a complete map of the macrophage pathway. In this example, 234 published manuscripts were reviewed and 506 reactions were integrated within the single centralized software framework of Cell

Designer [67]. No models were integrated in this case; rather the work produced a large and complex monolithic diagram interconnecting the various biological pathways.

These examples demonstrate current efforts to integrate models to gain greater insight into a particular area of biology. The results seem promising, and such efforts are only growing. There is an also an equally growing need for foundational tools and architectures that support the continued development of such integrated models in a far more scalable manner. This has led to the development of many new tools such as Cell Designer, which aim to offer an easy-to-use interface for linking biological pathway models. In the next sections, we first survey general techniques that are used for integrating such models in order to gain a perspective for reviewing the more specific techniques in computational systems biology.

3.6 Generalized Architectures for Integrating Models

There are computationally two broad approaches to integrating multiple models: monolithic and messaging. The *monolithic approach* involves the creation of one monolithic source code resulting from the merger of the source code of the individual models. The *messaging approach* involves the need for no such merger, but creates a mechanism by which the necessary input and output data streams common across all models can be shared and transferred either statically or dynamically. Some have referred to this messaging approach as a *communicationsapproach* [22].

3.6.1 Monolithic Approach

There are three broad types of monolithic approaches for integrating models: *manual*, *semi-automated* and *module-based*.

Manual Monolithic Approach

The *manual monolithic* approach is process where the model integrator manually creates a single program or "file" by "cutting and pasting" the source codes or "wiring together" the pathway diagrams of individual models. Figure 9, illustrates the cutting and pasting of source codes from two biological pathway models Model A and Model B to produce Model C. A model integrator may alternatively wire together the pathway diagrams of two pathway models to produce a single pathway diagram using a visual design tool. Many find the manual monolithic approach easy to use. It offers full control to the model integrator of the source codes or pathway diagrams. (Fig. 10)

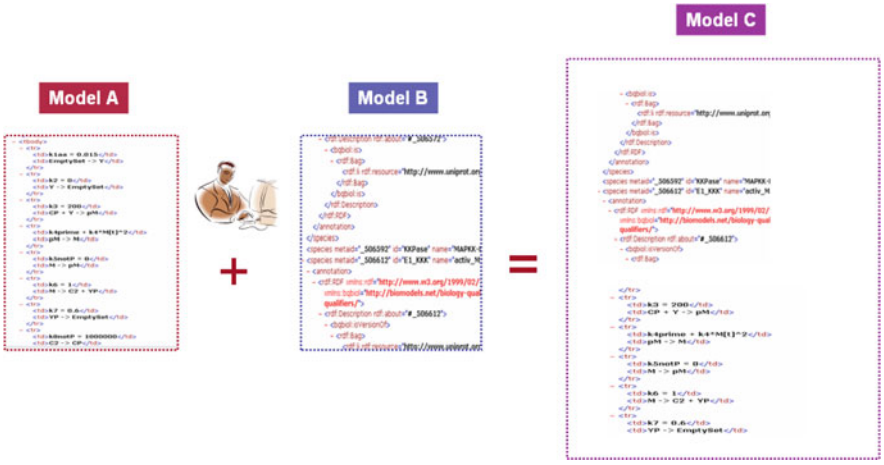

Fig. 9 Monolithic approach of cutting and pasting source codes of two models: Model A and Model B to produce a new source code of Model C

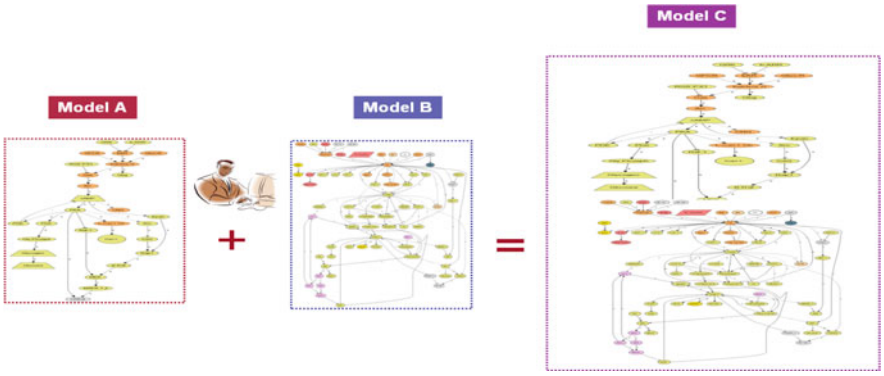

Fig. 10 Monolithic approach of wiring the pathway diagrams of two models: Model A and Model B to produce a new pathway diagram of Model C

The model integrator has control of all the coding details (control structure, memory allocation, data types, input/output file formats, etc.) [22]. Although this approach works, it has significant drawbacks. The individual performing the integration needs a complete and detailed understanding of the constituent models. In many cases, the source code is often difficult to obtain since legacy model codes are frequently complex, uncommented, and poorly documented [22]. The single integrated model's source code is also difficult to work with from a software engineering point of view (testing, debugging, verifying, updating, etc.) since it is much larger than its constituent model source codes, and improvements made to the original model source codes must be repeatedly made to each integrated model's source code as well.

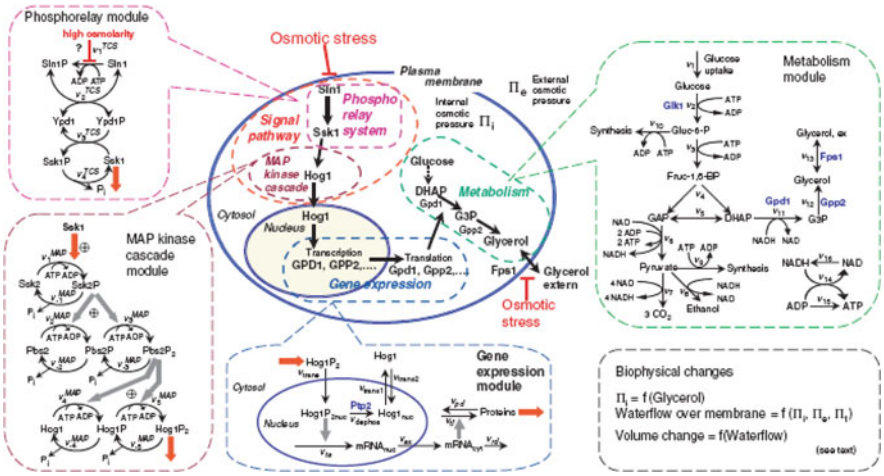

Fig. 11 An example of integrating models using the monolithic approach to integrate a model for the cellular function of osmoregulation [63]

Examples of recent published papers in using this monolithic approach are shown in Figs. 11 and 12. In Fig. 11 four models are integrated to produce on monolithic model for modeling the cellular function of osmoregulation [63]. In Fig. 12, three existing kinetic models are linked with one focusing on yeast glycolysis, a second extending this glycolytic pathway to the glycerol branch, and a third model introducing the glyoxalase pathway [20].

Semi-Automated Monolithic Approach

The *semi-automated* approach is a slight variation of the manual monolithic approach. In the *semi-automated* monolithic approach, software tools are used to accelerate the development of a single program from the individual model's source codes. In these approaches, a software tool, as illustrated in Fig. 13, combines together source codes or diagrams using some additional information to produce a single source code or diagram. Rarely do these tools produce the right source code, the first time. There is always manual intervention to review the initially produced output and then perform manual manipulations to produce the final output.

This semi-automated monolithic approach has the advantage of speeding up the initial merger process of source codes; however, the result varies based on the kind of semi-automation tool being used. In some cases, more effort is spent trying to get the tool itself working properly. Although this approach works, it too has the same significant drawbacks as the manual monolithic approach. One example of such a tool, SBMLMerge, is shown in Fig. 14 [68]. This tool merges two biological pathway models to produce one biological pathway model.

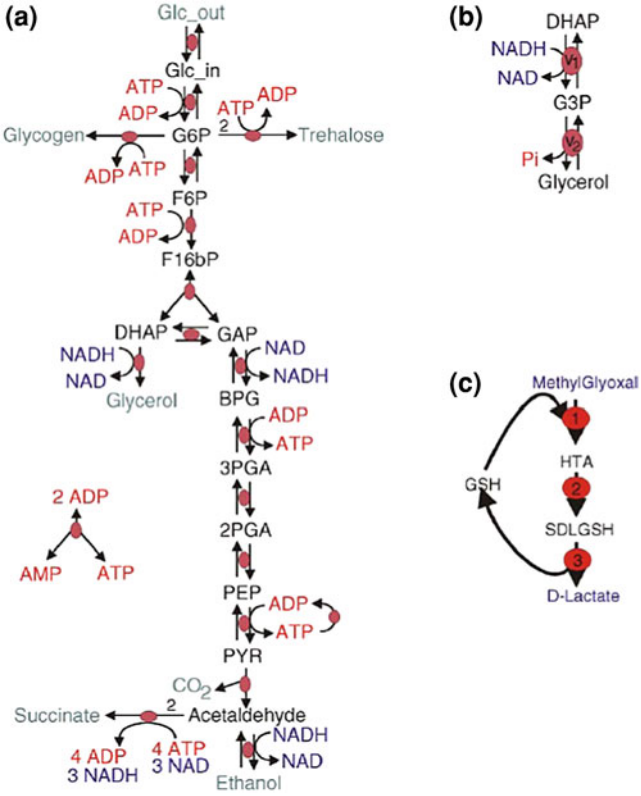

Fig. 12 An example integrating three pathways in a monolithic approach [20]

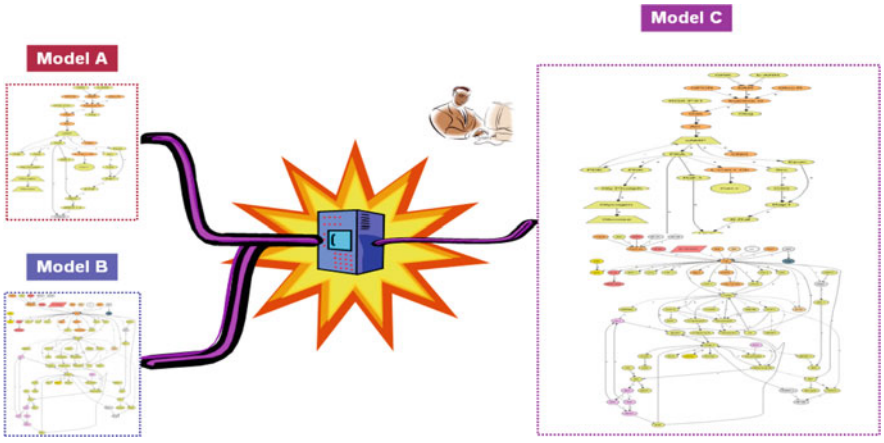

Fig. 13 Semi-automated monolithic approach of wiring the pathway diagrams of two models: Model A and Model B to produce a new pathway diagram of Model C

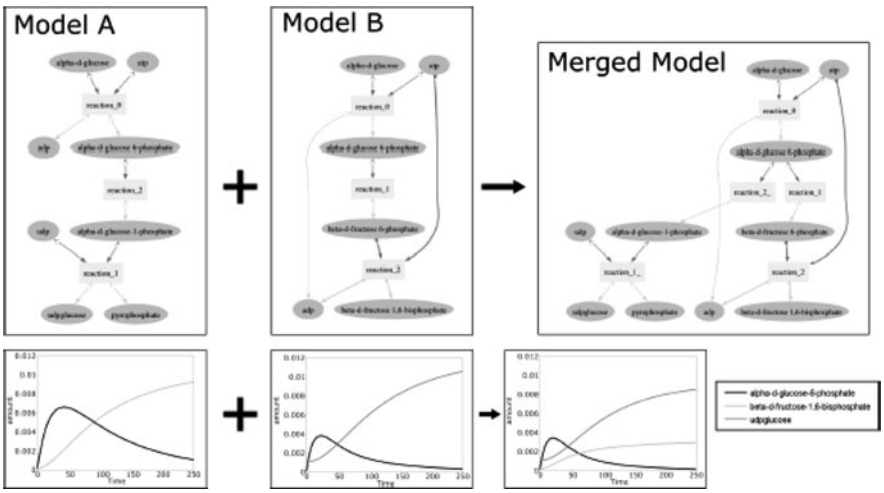

Fig. 14 SBMLMerge is an example of a tool that takes two biological pathway models and produces a merged model in a monolithic format [68]

For this tool to operate both biological pathway models need to be in SBML format and the tool produces one monolithic source code of the merged model in SBML format.

While these semi-automated tools may speed up the process, one disadvantage is that the model integrator does not have as much control of the resulting integrated source code, as they had in the manual approach. Some semi-automated tools insert new proprietary code and data structures along with a new format, which may take time to understand and debug. The model integrator, in that event, needs to not only have a complete and detailed understanding of the constituent models and their source code, as in the monolithic approach, but also now needs to understand how the tool for semi-automation itself works.

Module Based Monolithic Approach

The *module-based monolithic* approach addresses some of the limitations and drawbacks of the manual and semi-automated monolithic approaches. This approach offers the ability to employ reusable techniques for integrating models. The module-based monolithic approach results in the creation of single set of source code, but differs in that rather than decompose each model's source codes into blocks of source code designed for integration into another specific model's source code, the scientist decomposes each model's source code into software *modules*.

Modules are subroutines that are reusable and can be written with little knowledge of the other modules, and they can be replaced independently without

significant changes to the rest of the program [22]. We use the term *reusable* to mean that a module can be used in a variety of different situations without any changes made to it. Each module possesses a standard interface for invoking and passing parameters. Connecting their interfaces then recomposes the modules. The interfaces for each module are generally simple and consist of a set of input data that must be supplied before the module can be executed, and a set of output data that is available upon completion [22]. The computation that a module performs is encapsulated and hidden within the module. These modules can be classified and organized into searchable libraries and the strict interfaces allow for automatic compatibility checking.

3.6.2 Messaging Approach

Due to the limitations of the monolithic approaches, scientists turned to approaches that obviate the need to produce either manually, semi-automated manually or even programmatically, one source code from the combined models and approaches that require a substantial programming. This led to the messaging approach. There are two types of messaging approaches: static and dynamic.

Static Messaging Approach

In the *static messaging* approach, the models remain independent programs and do not affect each other as they are executing. Any one model accepts as input a dataset, which may reside in any variety of formats, and executes to completion to generate an output dataset. That output dataset is then given to another model (perhaps after some transformation) which that model uses it as input and also executes through to completion. This process can then be continued with other models, and they can be executed concurrently if there are no dependencies between their datasets.

Architectures for supporting static messaging architectures offer tools that provide automated ways for the user to select models and datasets, and then specify the distribution of datasets [22]. For example, in one such architecture, called Le Select, a database-oriented approach is used in which both models and datasets are stored in geographically distributed databases and the user specifies the execution and data distribution through textual queries in a standard database query language [22]. Other architectures provide a visual interface to specify the order of execution of the models [69, 70]. These architectures typically have different user interfaces and different input/output data formats making them difficult to use, especially for non-specialists, requiring a significant investment of time to learn.

Some architectures provide a standard user interface to each model [71]. This requires that the original user interface source code be removed from each model and replaced with the common user interface source code. To address the issue

of non-uniform data input and output formats, some architectures require the user to perform this data transformation manually between model runs while others require the user to change the model codes so that they use a standard data format [69, 70]. This requires that all the input and output source code be removed and replaced with source code to access the common database and use its data types.

Dynamic Messaging Approach

Using the dynamic messaging approach, the underlying model's source codes remain independently executing programs that interact only by exchanging data via message passing during execution. Architectures that use this approach can be classified by whether or not they include an independent application (a controller) that mediates the execution and messaging between the models.

The primary role of the controller is to transform exchanged data, which typically involves data type conversions, but they also sometimes control the startup of the models or track the global state of the integrated model as well. Architectures that do not include a controller are essentially libraries of data transfer and transformation routines customized for the data types and messaging styles needed by models. Architectures that do include a controller have messaging libraries that support direct model-to-model messaging as well as model-to controller messaging. In either case, these libraries often require the scientist to convert the model data into a standard data type, which is then communicated. All of these architectures require the scientist to perform an initial exercise of creating some mechanism of interfacing with each model through library calls in order to send or receive data.

The user then writes configuration files that specify which models are to execute, and the data that are to be sent and received. The messaging approach avoids the substantial model code rewriting required by the monolithic approaches, but the user still needs a complete and detailed understanding of the model codes in order to properly set up the correct interfacing for them [22].

In systems biology, for example, if one biological pathway model is written in MATLAB, and other model is in SBML, with each sharing four common variables that need to be communicated to solve the integrated problem, then the interface code is developed for the MATLAB model and for the SBML model. Once this interface code is written, both can transfer data. This interfacing process is often specific to a specific type of integration, so the interfacing process has to be repeated for each different type of interfacing.

4 Approaches to Integrating Biological Pathway Models

In this section, we will review current approaches to integrating biological pathway models within the field of systems biology. Two developments are predominant

in addressing this integration problem: (1) the development of new software systems that allow the integration of multiple biological pathway models within a single centralized software framework, and (2) the development of common *standards* to define and code biological pathway models.

4.1 Software Systems for Integrating Biological Pathways

As discussed in the previous section on *Generalized Architectures for Integrating Models*, there are two broad classifications for approaches to performing such model integration: monolithic and messaging.

In systems biology, we propose another dimension of classification: *informational* and *computational*. Informational approaches are those that provide a way for integrating multiple sources of biological pathway information but do no computing with that integrated information. Computational approaches are those that are a superset of informational architectures by *also* provide the ability to perform integrated computations across the biological pathways. In the diagram below, we review the predominant software systems for integrating biological pathways and characterize them in this two-dimensional context of monolithic versus messaging and informational versus computational.

In the previous sections, we have surveyed a number of different architectures that can be used for integrating biological pathway models. In Table 1, we summarize these extant approaches based on our two-dimensional classification

Table 1 Summary of architectures for integrating biological pathways

Architecture	Monolithic/messaging	Informational/ Computational
Virtual CELL	Monolithic (manual)	Computational
Kinetikit/Genesis	Monolithic (manual)	Computational
Cell Designer	Monolithic (manual)	Computational
Jarnac/JDesigner	Monolithic (manual)	Computational
JSim	Monolithic (manual)	Computational
E-CELL	Monolithic (manual)	Computational
Gepasi	Monolithic (manual)	Computational
Jarnac	Monolithic (manual)	Computational
StochSim	Monolithic (manual)	Computational
PathSys	Messaging (static)	Informational
SBMLMerge	Monolithic (semi-automated)	Computational
CellAK	Messaging (module-based)	Computational
Cellulat	Messaging (module-based)	Computational
Cellware	Monolithic (module-based)	Computational
SigPath	Messaging (static)	Informational
Gaggle	Messaging (dynamic)	Informational
XPAUT	Monolithic (manual)	Computational
MATLAB	Monolithic (module-based)	Computational

methodology from the previous section. The 18 architectures summarized in Table 1 represent those architectures that support the integration of multiple biological pathway models. As noted earlier, some of these architectures support the integration of biological pathway model *information* but do not support the computing of the integrated models.

Most of the approaches are monolithic. Clearly within the monolithic approach, the module-based mechanism is an ideal way to construct new models if modules are available, but the approach is impractical for integrating existing models. Cellware and MATLAB offer a module-based monolithic approach; however, they are very difficult to perform the actual programming for this kind of application for a non-specialist, have a significant learning curve, and cannot produce code that is scalable. What is interesting to note; however, is that the most widely use architectures in the systems biology community is the monolithic approach.

The messaging approach allows existing models to be integrated with minimal changes to the model's source codes. Since we are interested in model reuse and scalability, we will focus on the messaging approach in this work. Moreover, we will ignore those architecture that are only *informational* for obvious reasons, since the do not support computation.

Based on Table 1, there are only two messaging-style architectures in the current systems biology community: CellAK and Cellulat. These two architectures offer an approach that lets biological pathway models be integrated without the need to explicitly integrate the source codes as in the monolithic approach. Neither of these approaches, however, performs dynamic messaging among the integrated models, which would be even more advantageous.

4.2 Weakness of the Current Approaches

As can be seen from the previous discussion, the predominant method for performing such integration is a *monolithic approach*, which involves creating one large biological pathway model through either a manual, semi-automated or module-based method that executes on a single computer.

There are many weaknesses, which make this approach not scalable. First, scaling to, for example, approximately 1,000 pathways—the level required to describe a single cell—would require a massive effort beyond the research and development expended to obtain the original individual pathways [72]. Second, each pathway represents a knowledge domain, and it would be essentially impossible to have one person sufficiently knowledgeable in all the scientific areas to understand each of these domains well enough to manually construct a single monolithic program.

Third, the monolithic approach does not provide a means for pathways from proprietary models to be used with other models that are open source. The monolithic approach wrongly assumes that the owners of any one model will be freely willing to share their models directly and will not protect proprietary

information. The reality is that some models may be public and others, say ones owned by a pharmaceutical company may be private.

Fourth, most monolithic approaches support only one standard format. This means all other models need to be converted to that standard format. Fifth, and related to the previous weaknesses, the monolithic approach wrongly assumes that all models within an ensemble to be integrated are in the same format. The reality is that, while standards efforts are underway, models are coded in software platforms convenient to the author; and more practically any one platform is not capable of modeling all biological pathways. In the [66] example, all pathways had to be constructed in SBML and the integration had to be performed within the centralized framework of Cell Designer. Thus, this monolithic approach demands that in order to model the whole cell, all pathways would have to be placed into SBML or one common standard and then integrated within the framework of a centralized software system such as Cell Designer. Given the reality of standards adoption as aforementioned and the existence of 136 different software systems such as Cell Designer, a monolithic approach does not provide a scalable method to integrate multiple biological pathways to model the whole cell.

Sixth, the monolithic approach also wrongly assumes that all the models will run on the same computer and/or the same hardware platform. Different models may run on only certain hardware platforms, and more than likely were optimized and tested to run on particular hardware systems. This will become more important as the size and complexity increase, and special coding and computational accelerators are used to control the computational time.

Seventh, the monolithic approach assumes that all models reside in the same geographical location and ignores that biologists, even in a particular domain area of research such as immune response or cell motility, are distributed across laboratories worldwide. While they may build models within the same software platform and on the same hardware environment, the models themselves may be resident at a completely different location.

Eighth, the monolithic approach offers no real viable or practical mechanism to *maintain* the single large body of software code emerging from the merger of the software codes of the individual models. Consider four individual models that have been merged using the monolithic approach, and consider what process the author of the monolithic model will need to employ to track and maintain updates and changes to any of the four individual models. To any experienced software development manager, this is known as a *change management nightmare*. The reality is that any model may change constantly due to any one of several reasons including new advances in measurement, corrections to rate constants or identification of new species in a particular pathway.

Ninth, the monolithic approach, since it is centrally managed and maintained, places a new burden on the authors of the integrated model: that they become experts in the multiple domains represented by the individual pathway domains that were merged. This is nearly impossible.

Tenth and finally, many of the monolithic approaches attempt to enforce “standards” as a way to further reinforce a “monotheism”. This centralized

approach that would not fare well compared with more democratic, community-based approaches that understand and include research-driven development efforts. Creating a rigid standard before a field has matured can result in a failed and unused standard, in the best of circumstances, and, in the worst, can have the effect of stifling innovation [73].

4.3 What is Necessary?

What is needed is a method that directly addresses this integration and scalability problem by providing a parallel and distributed architecture allowing any individual pathway model to exist in any format and across different computers. Such a method would obviate the need to manually load, understand and interconnect each individual pathway, as is required in monolithic systems. It is clear from the above literature review, the common approach today is a monolithic approach in which computational biologists seeking to create an integrate model "cut and paste" source codes to create on monolithic model.

More specifically, of the two recent papers that provide examples of integrating biological pathway models, both involve the merging of SBML models using a manual monolithic approach [20, 63]. Thus, from a short-term perspective, even an architecture, which allows the integration of distributed SBML models, would be of huge benefit. SBMLMerge offers a tool that is a semi-automated monolithic approach, while valuable for those seeking to "cut and paste" models faster, it is not a scalable solution, since the resulting code base still cannot be maintained as the models change overtime, requiring re-integration each time every time the elemental models change.

A more advantageous solution would be an architecture that could, with minimal effort and no programming, *connect* or *couple* the codes of multiple SMBL models, linking their computations. Such a tool would be of immense benefit. If this same architecture could allow for the integration of models also not written in SBML, but in any other format, it will accelerate the development of complex models, versus waiting for all extant models to be converted and curated in SBML. This too will be particularly useful, given the reality that most of the encoded models available in scientific publications or on the Internet are not in a standard format. Of those that are encoded in a standard format, it turns out that most actually fail compliance tests developed for these standards [74]. In fact markup languages for model encoding such as SBML may not be always the only way for modeling biological pathway models, there are others that provide a range of descriptive and analytical powers. As the field matures, there will be a wider uptake of these alternative approaches for several reasons, including the need to take into account the great complexity of cellular organization [75].

From a software engineering perspective, each biological pathway model represents an individual software program, each with different inputs and outputs, written in different programming languages, by different developers, potentially

distributed worldwide. Modeling the whole cell, therefore, can be likened to a large-scale systems integration problem.

The research goal of this effort, therefore, is to develop a new approach to integrate biological pathway models that overcomes the intractability and lack of scalability of existing approaches. In his seminal work, [76] has demonstrated that the amount of effort to develop such a large-scale system increases exponentially with the amount of additional personnel communications required to coordinate development personnel. The thesis will explore a method that resolves this bottleneck in personnel communications by allowing individual teams to own and manage their own pathway models without being involved in a massive project management effort to coordinate the integration.

5 The Platform

This section presents the platform and scalable architecture for integrating biological pathway models. This architecture is called *CytoSolve*.

5.1 Key Requirements

Ten key requirements are identified below to address the weaknesses of previous approaches.

5.1.1 Scalability

If the goal is to model complex cellular systems and eventually the whole cell, the architecture must be able to integrate new pathway models with the same ease as it is to integrate the first one. Scalability is measured by the ease in which additional models can be integrated. Recall that complexity of integration, from our earlier discussion, has little to do with the number of equations in any one model. Two models with numerous equations can be relatively easy to integrate if they are written in the same program, same time scales, in the same domain and developed on the same hardware platform.

5.1.2 Opacity of Multiple Knowledge Domains

The architecture must express the property of *opacity*. Opacity means that when one individual is integrating a particular pathway model into a pre-existing ensemble of integrated models, one should not have to know the details and inner workings of all other pathway models. Each pathway may represent a unique knowledge domain,

and it would be essentially impossible to have one person sufficiently knowledgeable in all the scientific areas to understand each of the domains.

5.1.3 Support for both Public and Proprietary Models

The architecture must support both Public models (models where the source code is readable and available to all) and Proprietary models (models where the source code is inaccessible). The monolithic approach does not provide a means for pathways from proprietary models to be used with other models that are open source. Many pharmaceutical companies, for example, will not want to share the inner source code of their particular proprietary models; however, they are interested in coupling their models with other models to gain better understanding of a larger cellular process. Alternatively, researchers in the academic environment may wish to integrate their Public models with existing Proprietary models to learn some new aspect of science, but cannot currently due to confidentiality issues. By enabling a way to ensure protection of the source code of those Proprietary models, new industry and academic collaborations will be possible with far greater ease. Those with Proprietary models currently chose either not to follow standards to protect their models or if they do follow standards are unwilling to share their model code, which was the reason for the standard itself.

5.1.4 Support for Multiple "Standards"

The architecture must support any pathway model code in any format or "standard." While the architecture should support the integration of pathway models constructed in a standard such as SBML, for example, it should be able to communicate with models in any format.

5.1.5 Heterogeneity of Integration

At any time models may be in different formats. The architecture should support the ability to integrate, in real-time, models that are in different formats. Thus, if there are three models, even if one model is in SBML, another in MATLAB and a third in FORTRAN, the architecture should be able to integrate them with minimal to no effort.

5.1.6 Cross Platform Support

The architecture should allow models developed on different hardware and computing environments to be integrated with ease. Different models may run on only certain hardware platforms and more than likely were optimized and tested to

run on a particular hardware system. It will be far easier to keep a model resident on a hardware platform for which it was designed and tested for versus having to recode or reconfigure it in any manner to a new hardware platform, which could prove to be very expensive and time-consuming.

5.1.7 Independence of Location

The architecture should support integration of models across geographical boundaries. While each model may be on different computers, they may also be physically at different locations anywhere in the world. The model should support protocols for communicating with models anywhere without regard to geographical location.

5.1.8 Ease of Maintenance

Any model integrated within the architecture should be able to dynamically change with little to no source re-coding efforts to incorporate changes for that model into the larger integrated model. In the monolithic approach, any change to an individual model typically requires significant recoding and retesting of the integrated model. In this new architecture, we want to avoid such a process. This requirement is extremely important to create a scalable architecture. The addition of new models should not require changes to any of the other existing models.

5.1.9 Decentralized Management and Distributed Control

Decentralized management and distributed control means that each model is maintained at the "local" level, not at a central level. The current monolithic approach requires centralized model curation, such as many of the existing model repositories. We want to support local management of models. Moreover, any creator of a model should be able to integrate their model from their local location to an ensemble of distributed models. This means that if one owner of a model has Model A and wishes to quickly test or integrate their model with a set of three other models: Model B, C and D. They should not have to download each of the other models to their local computer. With ease, the architecture should enable the owner of Model A to integrate with the other three models with little to no effort.

5.1.10 Hierarchical Support

Biological systems are systems of systems. This means that the architecture should support the ability for systems to be composed of other systems, while ensuring that each system satisfies the requirements herein.

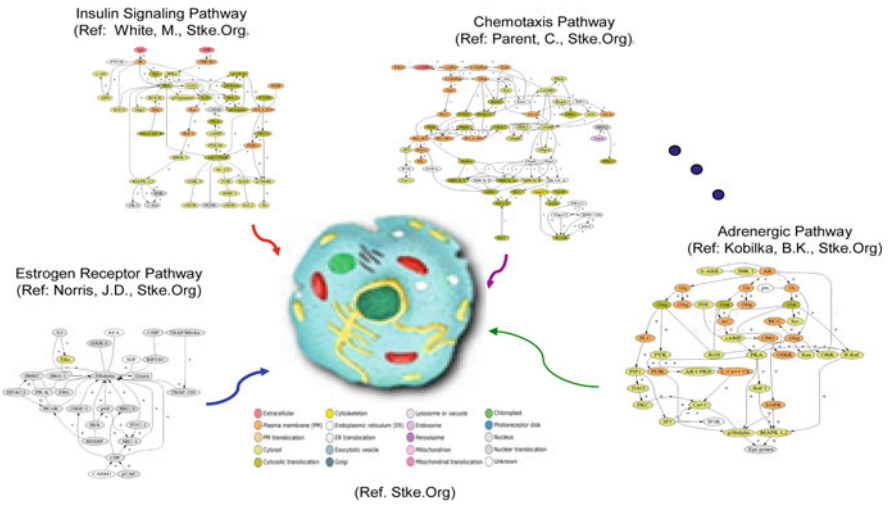

Fig. 15 A cell as an interconnection of biological pathways

5.2 Functional Specification

The above ten requirements will be used as the basis for defining the architectural design. We begin our functional specification process by abstracting the cell to be interconnection of biological pathways, as shown in Fig. 15. In this approach, we make the following key assumptions:

1. The cell or compartment is well-mixed. This means that a sufficiently long-time between reaction collisions takes place to ensure that each pair of molecules is equally likely to be the next to collide. This also means that the concentration of each species is high and transport essentially instantaneous.
2. The progress of the system only depends on previous state (e.g. Markov process).
3. Between cells and compartments, transport is slower and associated with an observable rate.
4. We treat each pathway model as a *black box*. This means the following:

   - Inputs and outputs are species concentrations
   - Changes in localization are represented by *compartments*
   - Species are defined by their compartments
   - Species can move through compartments
   - Species can inhabit one or more compartments

5. The cell can be modeled as an integration of biological pathway models. Recall, biological pathways are moving from diagrammatic representations, as shown in Fig. 2 above, to biological pathway models, and each biological pathway

model has internal parameters along with inputs and outputs which are the molecular species concentrations for the nth and $(n + 1)$th time step, respectively.

6. The cell can be modeled as an integration of biological pathway models. Recall, biological pathways are moving from diagrammatic representations, as shown in Fig. 2 above, to biological pathway models, and each biological pathway model has internal parameters along with inputs and outputs which are the molecular species concentrations for the nth and $(n + 1)$th time step, respectively.

Thus, modeling the cell therefore can be seen as an interconnection of biological of pathway models as shown in Fig. 15.

5.3 Platform Design

Based on the earlier discussions, it is clear that the monolithic approach is widely used primarily because there are no other real alternatives in systems biology today. The main problems with this approach have already been discussed. The messaging approach, which has not been fully developed for systems biology, except in the two cases of Cellware and Cellulata, is in the right direction. However, both of these methods use a static messaging approach. There are no known solutions, to our knowledge, which use a dynamic messaging approach that supports scalability as well as ease of maintenance. Our first decision is to pursue a dynamic messaging approach as the basis of the architecture. This dynamic messaging approach provides nearly all of the advantages to address the weaknesses of the monolithic approach and the static messaging approach, for the reasons previously stated.

5.3.1 CytoSolve: A Dynamic Messaging Approach

Based on the above discussion, we now introduce CytoSolve, a scalable architecture for integrating an ensemble of distributed biological pathway models. In Fig. 16 the layout of the architecture for supporting a dynamic messaging approach is illustrated. The layout of this architecture will meet all of the requirements of the specifications outlined in the previous sections.

The elements of this architecture are:

- Biological Pathway Models—These are the boxes with the input and output arrows along the outer edges of the diagram. Each biological pathway model is a computer software program.
- Internet—This is represented by the internet "clouds". This serves to show that the biological pathway models can reside anywhere geographically on the planet

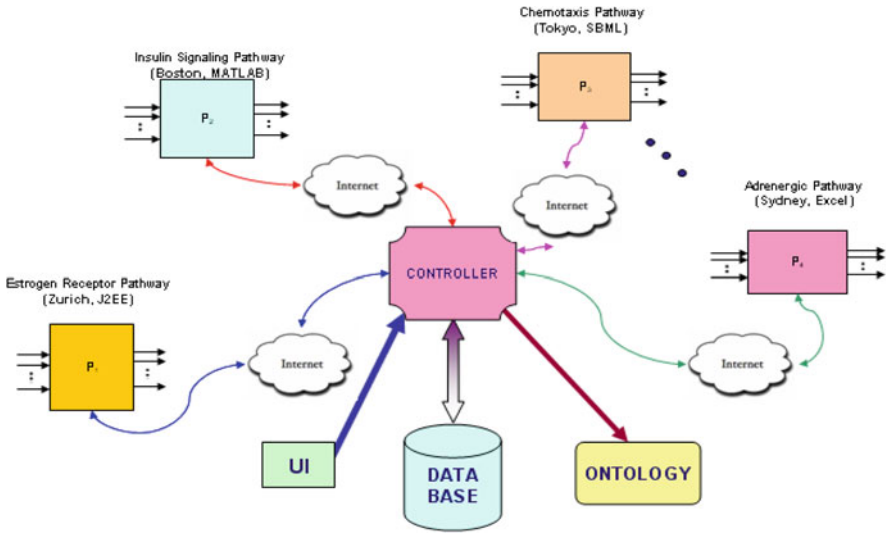

Fig. 16 CytoSolve—a dynamic messaging approach

and can use the internet for communication through the controller. This does not mean that we *have* to use the Internet. All models can be centralized on one computer and CytoSolve will communicate locally to each model.

- Controller—This module serves to coordinate the computational activities across the various models.
- Controller-to-Pathway Interface—These are the arrows in the diagram from the Controller to the Pathway, and represent the mechanism by which the controller communicates with each individual pathway model.
- User Interface—The user interface allows the user to specify which models will be included within the architecture.
- Ontology—This is the data which specifies characteristics of each model's input–output behavior to allow the Controller to effectively communicate across all models.

The key features of this architecture include an infrastructure that provides simple communications interface to each model, is distributed, Web-enabled, and automatically aggregates the models to build the integrated model. The architecture supports both distributed and parallel processing, while using a hybrid of shared memory and message passing. The shared memory is for tracking the species concentrations across all models in time. Message passing is used for remote communications between the controller and individual models. The architecture is built in an open environment supporting: (1) publicly available tools, and (2) emerging standards. The discussion of the details of the implementation, which is evolving, and currently in a web-based environment at www.cystosolve.com, is beyond he scope of this discussion.

5.3.2 CytoSolve: Solution Methodology

CytoSolve dynamically integrates the computations of each model to derive the species concentration of the integrated model. Details of the solution methodology are detailed in another publication [77].

In summary, the method works with the Controller performing initialization of the system by allocating memory storage for the computed species concentrations of each model and the integrated model. A component called the *Monitormonitors* the progress of each model's computation, during its initialization, accesses the initial conditions. Control is then passed to the a component called the *Communication Manager* which awakens all the models to start up and become ready to process a time step of calculation, and then invokes the Monitor.

The Monitor proceeds to invoke all models in parallel to execute a time step of calculation using the species concentration values of the integrated model at time step n, as the input to all models. Each model executes and computes one time step of calculation on its own Remote Server. Monitor tracks the progress of each model's completion. Once a model completes its computation, the output is stored, and the model then goes to *sleep* to optimize use of the Remote Server's CPU usage. By sleep, we mean that the model goes dormant until invoked again by the Communications Manager to process another time step of calculation. Once all models have completed their processing for a time step, the Monitor passes control back to the Communications Manager, and the Monitor itself goes to sleep.

Once all models have completed a time step of calculation, the Communication Manager invokes the *Mass Balance* component to dynamically couple the computations at time step n of each model to evaluate the integrated model.

6 EGFR Model Validation

CytoSolve is validated by comparing the solution it produces with the one generated by Cell Designer, a popular tool for building molecular pathway models in a monolithic manner. As a control, the Epidermal Growth Factor Receptor (EGFR) model published by Kholodenko et al. is selected for this comparison. Snoep et al. have authored the model into the SBML language. The *entire* EGFR model is loaded into Cell Designer and executed on a single computer. The same entire EGFR model, to test CytoSolve, is *split* into four models and distributed on four different computers, as shown in Fig. 17. CytoSolve and Cell Designer are run for a total of 10 s in simulation time.

6.1 Results

CytoSolve and Cell Designer produce near exact results as shown for two example species in Fig. 18. The results demonstrate the viability of CytoSolve's

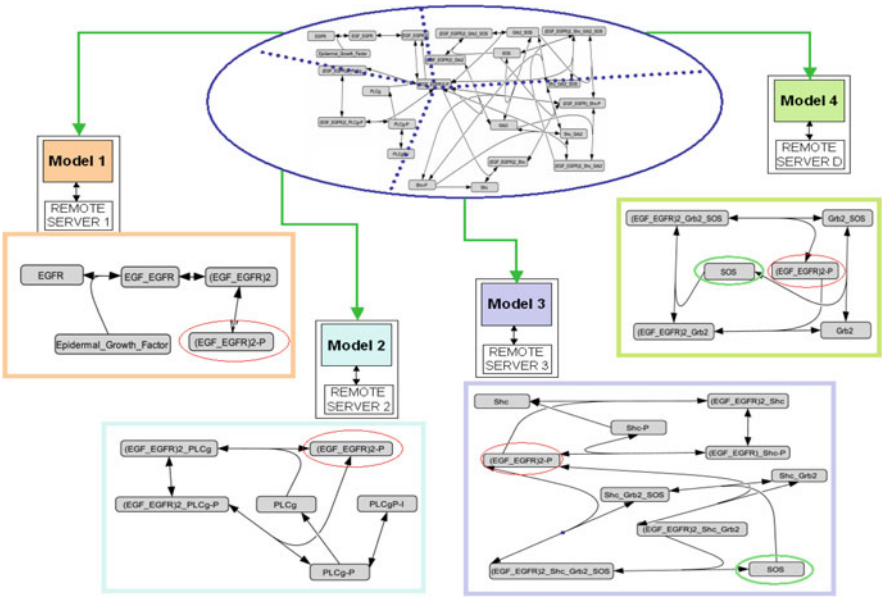

Fig. 17 Complete EGFR model of Kholodenko split on four Remote Servers for CytoSolve solving

unique distributed approach not only to solve problems that monolithic approaches are capable of solving but also to provide greater flexibility and scalability in integrating multiple biological pathway models, which monolithic approaches are incapable of doing. In CytoSolve, any *one* pathway can exist in any format on any computer, and there is no need to manually load, understand and interconnect each individual pathway, as is required in monolithic systems.

CytoSolve generated exact results to Cell Designer; more importantly, the integration of the four models in CytoSolve did not require any manual "wiring" as is needed by Cell Designer. CytoSolve's compute time was greater than Cell Designer; however, most of this compute time was due to Transmission Time. Since CytoSolve works in a distributed parallel fashion, its compute time is a direct function of the compute time of the largest pathway plus the associated Transmission Time and overhead for Controller Time to integrate. For Cell Designer, the compute time will be the compute time of the whole integrated pathway.

Finally, and perhaps equally important, is that managing a monolithic model, composed of other models, is a change management nightmare. Consider a small example of a monolithic model "cut and pasted" or concatenated from the four models of EGFR, aforementioned, and each model being published and created by different authors. Now, suppose once the monolithic model has been constructed,

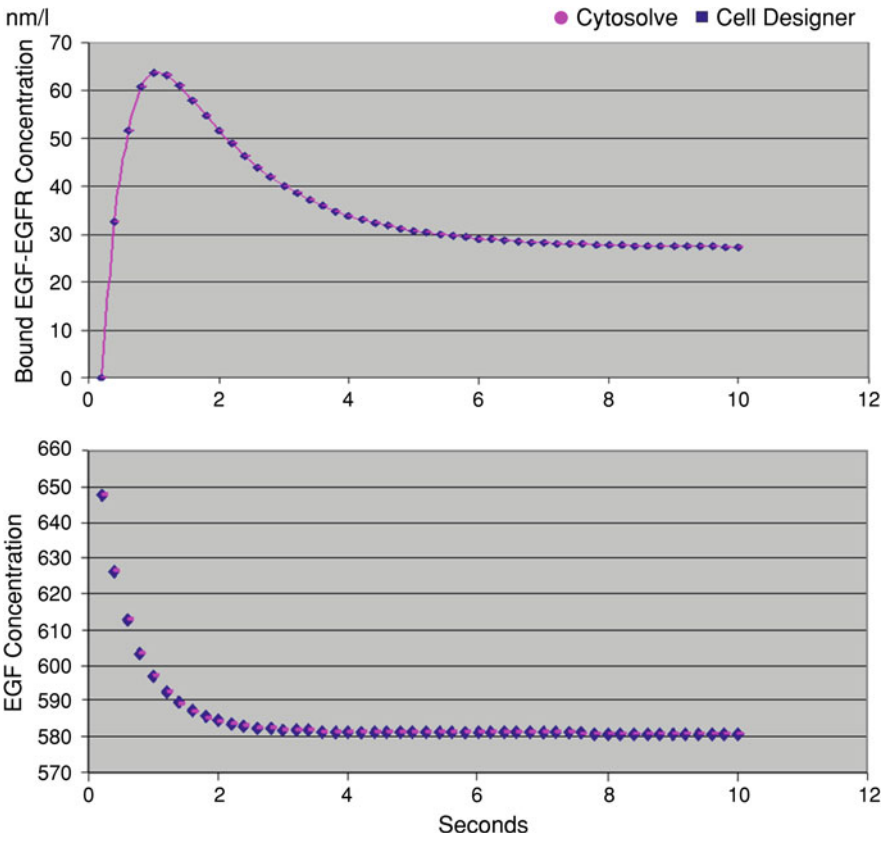

Fig. 18 Comparison of results from CytoSolve and cell designer for two species. **a** compares values of the EGF-EGFR species. **b** compares values of EGF concentration

that many months later, the authors of each of these models changes rate constants, pathway connections, etc., at that point the author of the monolithic model would have to rebuild the entire monolithic model, by instantiating changes from each author's model, which may be tenable for four models (possibly based on the complexity and domain specificity of each model). Modeling the whole cell while managing such changes across a suite of hundreds of such models will be untenable.

7 Solving an Unknown Problem Using CytoSolve

The immune system has many different types of cells acting together to protect the body against viruses, bacteria, and other "foreign invaders." Part of this protection

includes the production of interferon (IFN), a protein that plays a special role in triggering the body's response. The following describes what interferon is and why it is so important to the immune system.

7.1 IFN Response to Viral Infection

The immune system consists of a complex network of cells, tissues, and organs all working in tandem to ward off infection and keep us healthy. This includes interferon, one of the proteins called cytokines, which are diverse and potent chemical messengers that can trigger the immune system to attack invading pathogens. Interferon signals neighboring cells into action and also interferes with how foreign cells grow and multiply.

In humans, IFNs also play a roles in cell growth, differentiation and immunomodulation. IFNs are divided into two groups depending on their molecular basis; type I IFNs (IFN-alpha and IFN-beta) are produced by a variety of cells following virus infection, and type II IFN (IFN-gamma) is produced by activated T cells and natural killer (NK) cells [78]. There are three classes of Interferon, alpha, beta and gamma. Interferon alpha and beta are produced by many cell types, including the infection-fighting T-cells and B-cells in the blood, and are an important component of the anti-viral response.

7.2 Elements of the IFN Response

The IFN response mechanism of the cell to virus infection is a core cellular function. There are four key biological pathways, which are involved to elicit IFN response to virus infection as shown earlier in Fig. 7:

- Up regulation of IFN-Beta
- IFN receptor signaling to produce IRF-7
- Virus amplification cycle to produce more IFN-Beta and IFN-Alpha
- Regulation and balancing by SOCS-1

7.2.1 Virus Infection Model

The virus infection pathway creates IFN-Beta as an initial response to virus infection. Scientists in Moscow, Russia in 1994 modeled this pathway [79]. The original code was written in MATLAB. In this pathway, the viruses injects single-stranded RNA into the host cell, which leads to the formation of double-stranded RNA. Double-stranded RNA triggers the activation of virus-activated kinase (VAK), which phosphorylates IRF-3. Phosphorylated IRF-3 is a transcription

factor for the IFN-Beta gene. The expression of this gene results in the initial production of IFN-Beta.

7.2.2 IFN Receptor Signaling Model

The IFN receptor signaling produces IRF-7 as a preparation mechanism for the infected cell and neighboring cells. Scientists in China in 2005 defined this pathway model in *FEBS* [80]. Only a pathway diagram along with parameters exists for this pathway model, but no software code. IFN-Beta (or IFN-Alpha) lands on the IFNAR receptor to initiate the up regulation of IRF-7 which is a critical protein for signaling the cell itself as well as neighboring cell of the virus infection. The binding of IFN with the receptor leads to the STAT protein being phosphorylated. The phosphorylated STAT forms a homo-dimer and becomes the transcription factor for IRF-7 gene after binding to IRF-9, which leads to expression of IRF-7. This signaling mechanism *prepares* the cell for further defenses by producing IRF-7.

7.2.3 IFN Amplification Cycle Model

The IFN amplification cycle is a critical step in the response to protect the cell from virus infection. A team of scientists from the America created a dynamic model of this pathway. The article was published in the *Journal of Theoretical Biology* [81]. They programmed the pathway in XPAUT which can be saved in SBML. The virus interaction with IRF-7 not only serves to up-regulate IFN-beta but also serves to up-regulate IFN-alpha.

7.2.4 SOCS1 Regulation Model

This biological pathway produces SOCS1 to regulate and balance the production of IFNs. Without this pathway, the additional levels of IFNs, beyond what is necessary to stop the virus infection, can itself have detrimental effects on the cell. Scientists in Japan in 2001 defined this pathway model in *Genome Informatics* [82]. They programmed the pathway in MATLAB. Here, JAK binds to the IFN receptor and forms the JAK-IFNR receptor complex. Once IFN binds to the receptor, the resulting complex associates with each other and forms a homo-dimer. This dimer undergoes phosphorylation, leading to a form as IFNRJ2*, which catalyzes the phosphorylation of STAT1. The phosphorylated STAT1 also forms a homo-dimer and acts as a transcription factor of SOCS1 gene. The resulting protein, SOCS1, inhibits the kinase activity of IFNRJ2 and is the key component of the negative feedback loop.

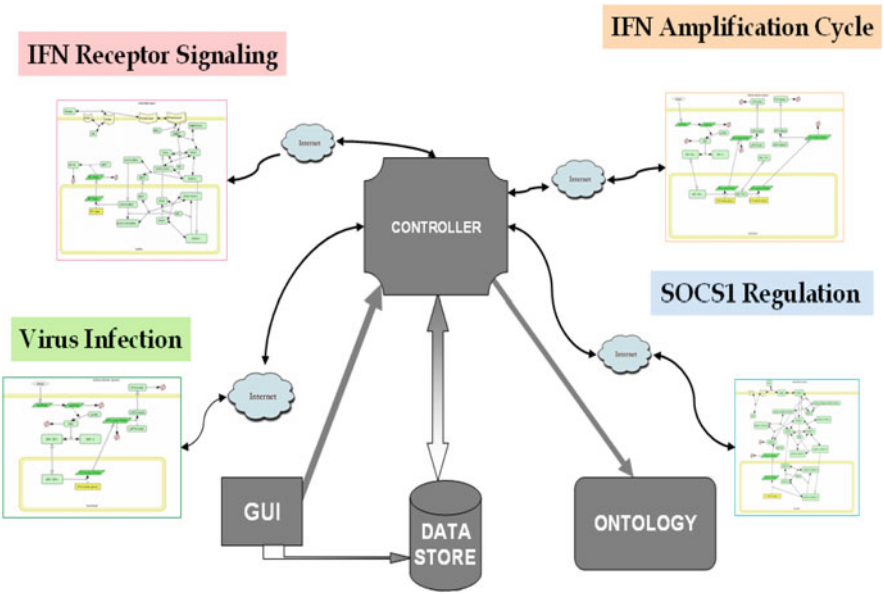

Fig. 19 CytoSolve approach to integrating the four models

In Fig. 19 below, each of the above models, as depicted is loaded into CytoSolve for dynamic integration.

7.3 Results

The summary results from the integrated model of IFN are shown in Fig. 20. Key molecular species presented in this figure are IRF-3, IRF-7, IFN-Beta and IFN-Alpha. This integrated model combines the four pieces of interferon pathway: virus infection leads to up regulation of IFN-Beta; IFN-Beta then results in the creation of IRF-7; the existence of IRF-7 then results in positive feedback to which increases a massive production of IFN-Alpha and IFN-Beta; finally, control of JAK/STAT signal transduction pathway by SOCS1 results in regulating and balancing the production of IFN-Alpha and IFN-Beta.

Close review reveals several important elements of the integrated model. First, during the first ~13 h (~50,000 s), the concentration of IRF-7 through time is a sigmoidal curve which reaches the steady state value of 0.7 nM. Second, during this same first ~13 h period, the concentration of IFN-Beta and IFN-Alpha slowly increases. What is interesting to note is that within the first 3.3 h (~12,000 s), the initial production of IFN-Beta is then followed by the production of IFN-Alpha. IFN-Beta is produced within the first 30–40 min (~2000 s to ~2500 s).

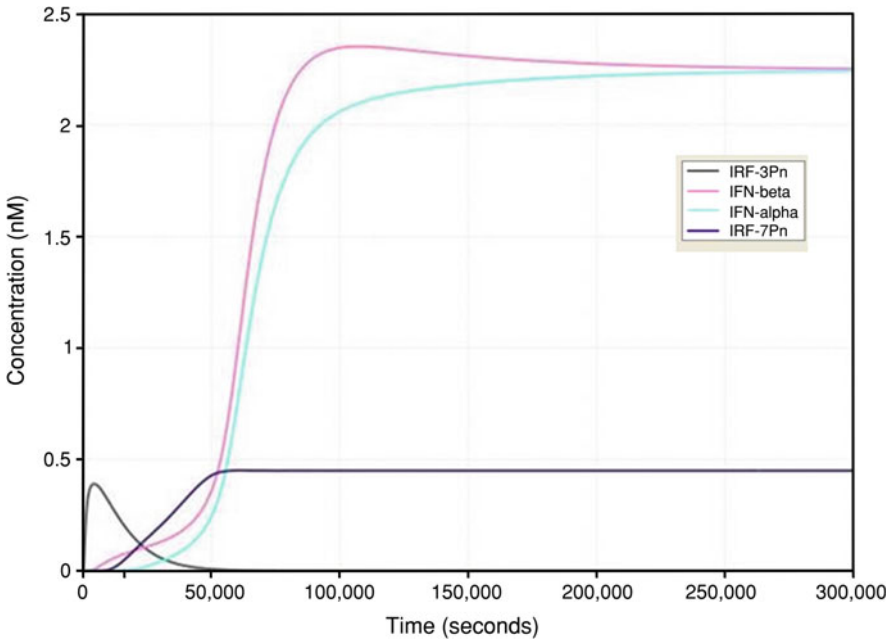

Fig. 20 Integrated model solution

The initial production of IFN-Beta, after the 40 min period and before the 3.3 h period is defined by a marked increase in IFN-Beta production.

Third, after the first ~13 h (~50,000 s) to ~25 h (~90,000 s), IFN-Beta and IFN-Alpha exponentially increases. Fourth, after ~25 h (~90,000 s), IFN-Beta and IFN-Alpha concentrations reach their maximum and gradually approach steady state due to the balance between positive feedback system and negative feedback control from SOCS1 activation.

The results of the CytoSolve solutions were tested with known experimental data. This revealed that IFN Beta production in the first 30–40 min, as predicted by the model, matches experimental data. In addition, IFN-Alpha begins production after a ~3 h delay time, and this verifies with the experimental approximated expected time required for the positive feedback cycle to start. Finally, both IFNs reach their peak in the ~20 h range as predicted by various experimental research [83–85]. Greater details of this integrative model of IFN are described in a forthcoming paper.

8 Discussion and Conclusions

This paper has introduced CytoSolve, a new computational environment for integrating biomolecular pathway models and a collaborative platform for computational molecular systems biology. The initial results from the EGFR example

has demonstrated that CytoSolve can serve as an alternative to the monolithic approaches, such as Cell Designer. The solution of the IFN integrated model demonstrates CytoSolve's ability to solve new problems by merging multiple molecular pathway models.

The purpose of CytoSolve is to offer a platform for building large-scale models by integrating smaller models. Clearly, if we want to model the whole cell from hundreds of sub-models, each of which is owned by various authors (each making changes to their models), the monolithic approach is not scalable. CytoSolve's dynamic messaging approach offers a scalable alternative since the environment is opaque (treats each model as a black box) support for both public and proprietary models is extensible to support heterogeneous source code formats, and finally supports localized integration, a user can initiate integration from their own local environment.

CytoSolve is now available at http://www.cytosolve.com as on-line web computational resource. The portal offers researchers an environment to collaborate and integrate quantitative molecular pathways without needing to be a specialist on the computational architecture and neither an expert of all the multiple sub-models to be merged. Future work we will include a more sophisticated Ontology to manage nomenclature and species identification across all individual biological pathway models to be integrated by means of the web application, and automated searching for related biological pathways.

8.1 Future Work

This research may also serve as a foundation for several new areas of investigation including:

8.1.1 Spatial Scale Variation

At the present time, CytoSolve supports only computational models that represent one single pool of material or several distinct pools connected with specific transport relations. We have not considered changes in concentrations on a continuous spatial scale. We believe that the architecture, based on its modular approach and support for multiple compartments, can support varying spatial scales. However, more testing will have to be performed to understand the computation times required to fully support such spatial variations. The description language FieldML is available to support this process.

8.1.2 Adaptive Time Stepping of the Controller

All models are currently computed using a single adaptive time step, which is taken to be the fastest time step among the ensemble of models. This is not

optimal, as some component models may be varying more slow than others. Additional effort is required to implement intelligent adaptive time stepping at the Controller level to observe the time scales of different models and invoke them only when necessary. Such an effort will result in improved computation time performance.

8.1.3 Implementation and Integration with Emerging Ontologies

CytoSolve has support for integrating other ontologies such as MIRIAM; however, future research needs to be done to fully integrate MIRIAM and other such ontologies. This effort will enable CytoSolve to support many more model formats with greater ease, leveraging standards that the systems biology community globally accepts.

9 Epilogue

In the Summer of 1978, I was 14 years old and had just completed my sophomore year at Livingston High School in New Jersey. That summer I had been fortunate to have been accepted into the Courant Institute of Mathematical Sciences gifted students program in computer science at New York University. During that Summer, I and 40 other of my fellow students learned eight different computer programming languages including FORTRAN, COBOL and PL/1 to name a few. The course that summer at NYU, and my deep interest in mathematics, in retrospect were important elements that supported my future work.

Swamy Laxminarayana, or Swamy as I called him, however, was my gateway into a completely new world from which I can trace nearly all of my accomplishments. Swamy exposed me to the world of pattern analysis and recognition at an early age. He was a friend above all; although I was nearly 30 years younger than him, he treated me as a colleague with great respect, genuine warmth, and love. My mom, Meena Ayyadurai, worked at the University of Medicine and Dentistry of New Jersey (UMDNJ) as a Systems Analyst while Swamy worked upstairs with Dr. Leslie P. Michelson in the Laboratory for Computer Science. My mom had first introduced me to Dr. Michelson with whom I had begun to build an electronic mail system. In fact, that electronic mail system was the worlds first E-Mail System, for which I received recognition by the Westinghouse Science Award in 1981.

I recall while I was in the midst of building that E-Mail System in 1980, Dr. Michelson began interviewing candidates to find a research staff member to do new research in biomedical engineering. I remember Swamy coming to see Dr. Michelson for the interview. He was dressed in a brown suit with a tie and beige shirt. His hair was worn back with light streaks of grey, he had a peppered mustache, and greeted me as we passed with a huge smile. He appeared very eager, with a mission and purpose.

Swamy, like my mom and I, was one of the few Indians who worked at UMDMJ. Since the chance meeting in Dr. Michelson's laboratory, I saw him again at various times passing in the hallway. Again the greetings were silent with simple exchanges of smiles. One day when I went home for dinner, I saw Swami seated at our dinner table! My mom, always generous and kind, had invited Swamy over to our house for dinner as she had bumped into him in the hallway. After dinner, Swamy and I sat down and spoke for the first time. He seemed to be on a mission, in a gentle way, to convince me to pursue science, and use my skills in science, mathematics and computer science to help the world.

In particular that conversation opened me to the world of pattern analysis and pattern recognition. I remember that conversation well. Swamy began by telling me about some research he and his sister had done in India. He used this research example to give me an example of how pattern analysis could be used. The example was rather unconventional compared to his conventional research assignments. He and his sister were very curious about seeing if there was a correlation or pattern to peoples palm prints and the onset of disease. In short, Swamy was interested in exploring if there was any truth in palmistry. He explained in detail how they had collected nearly 1000 palm prints from various people. They then created a database with each individual's name and their palm print. The palm print was a hand drawn sketch. They then had asked each person for their health history including any major diseases they had had or were suffering from. This health history was attached to the palm print.

Each palm print and the associated health history were reviewed manually to see if there were correlations between palm print features to health history. According to Swamy they had found some clear patterns. Certain palm print features had a high correlation to certain forms of disease. He provided me various examples of what he and his sister had discovered. Here was mathematics and modern science being applied to understand an ancient practice. Palmistry itself was a method of visual pattern recognition. Swamy's example intrigued me.

Swamy then provided me another example of how pattern recognition could be used. One of Swamy's passions was to model the human heart's electrophysiological behavior. He shared with me diagrams written on notepads at our dinning room table of the then current theories of how heart signals propagate. It seemed quite complex. He advised me to explore this field since there were many unknowns which would require someone good at both mathematics and computer science to model such phenomenon. Remember, this was at a time when the use of computers was just in their nascent state in the biological sciences.

That dinner conversation really got me thinking. While I had learned a lot of mathematics, was a star student in my high school, and was challenging myself in building the E-Mail System, something in my heart awoke. Swamy's conversation inspired me to explore how the skills I was developing could be applied to pattern analysis. Moreover, applying computing to biology seemed fascinating. Over the next several months, Swamy and I kept in touch and would have lunch together in the UMDMJ cafeteria. He was always encouraging of my work and kept asking me to work with him. In the Summer of 1981, I was close to completing a version of

the first E-Mail System and would have more free time. I had just been accepted to M.I.T., and I promised Swamy to do a project with him, even if it was long distance from M.I.T.

During that first semester at M.I.T., the Institute had recreantly implemented UROP, Undergraduate Research Opportunity Program. One of the research opportunities was to work in pattern analysis for Tadoma. Tadoma is a method by which deaf-blind people communicate. Few understood how deaf-blind people were able to "listen" to someone else through the tactile method of putting their hand on someone's face. I mentioned this project to Swamy in a phone call and he encouraged me to pursue it since it would provide me some hands-on learning on pattern analysis project.

I kept Swamy posted on my activities at M.I.T. In one of our conversations, Swamy told me about another interesting project that involved heart electrophysiology relative to young infants. Apparently there was a phenomenon where young infants were dying in there sleep. This was called Sudden Infant Death Syndrome (SIDS). In SIDS, babies died in their sleep from what was known as an apnea. Today many hospitals and sleep labs test people for sleep apnea. At that time in the early 1980s, research in SIDS was just a new field. Swamy, through his collaborations, had acquired access to the best infant sleep data through Montefiore Hospital in New York City. This time series sleep data provided sleep states of thousands of babies as well as points in time at which they had the occurrence of an apnea. Swamy's thesis was that babies sleep states, or patterns of sleep states, may be indicative of the onset of an apnea.

Swamy gently prodded me to get involved in this project to build algorithms to test his hypothesis. Over the next several months, he provided me various books on Haar and Walsh Transforms as well as books on signals processing. Signals processing was a passion of Swamy's. He wanted me to learn as much as possible in this field for he knew that it would be a strong foundation for future work in pattern analysis. I as then a 17 year old had no idea of its long term importance. The Tadoma project I was working on at that time provided me experience in data acquisition but not in actually developing algorithms.

Swamy, to support my learning, advised me to write up sample programs using Haar and Walsh Transforms. I took his advice and learned a great deal about these two signals processing methodologies. Next year, Swamy provided me raw data of SIDS from Montefiore Hospital that he had acquired in 1977. He also provided me various papers he had written dealing with SIDS in his previous work. The earlier programming project I had done on Haar and Walsh Transforms was valuable to the SIDS project. In this project, my task was to review the data and see if I could find patterns of sleep states leading to the onset of an apnea.

The SIDS project was like solving a puzzle. Swamy taught me that babies have six states of sleep, whereas adults only have five. The data resembled up and down steps, each step being a sleep state. There were certain points which were marked when an apnea took place. The goal was to look backward from the point of the apnea to see if certain patterns sleep state correlated to when an apnea occurred.

I used the earlier Haar and Walsh Transforms to build pattern analysis methods to predict the onset of an apnea from the sleep waiting times.

While traveling back and forth between M.I.T. and UMDNJ, Swamy never micro managed me but was always there with patience to answer any of my questions. I worked on this for over a year. Some valuable results came from the analysis. The next year, the Fall of 1983, we had heard about the international conference to be held in Espoo, Finland. This conference on medical and biological engineering was the worlds largest such event. In the world of academia were as someone once said, "people fight over nothing", Swamy was unbelievably generous, again in retrospect. He wanted to include my results in a paper on SIDS, and where many in academia would have never thought about giving an undergraduate authorship on a paper, Swamy made me a co-author. In addition he invited me to come to Espoo to co-present the paper with him.

Going to Finland would be my first trip out of the United States to a foreign country besides India. Our paper was accepted for the Summer 1984 Conference. I was on my way to Finland. Attending that medical conference was an amazing experience. Twenty-seven years later today, I still remember that trip vividly. I was the youngest registering at the conference, but Swamy introduced me as his colleague from M.I.T. to all of the other scientists. For a 19-year old to go on a flight, attend banquets, hear scientific talks, travel the Finnish countryside, and hear a new language, was out of this world. Swamy had made that experience possible for me. He exposed me to a larger world, beyond science.

It was clear to me that scientists went to these conferences not just for the science but to experience other lands, local cuisines, and meet other people. Had Swamy not taken me on this trip, I would never have been exposed to this aspect of science. After coming back from that Conference, my enthusiasm for science and pursuing research in pattern analysis and pattern recognition skyrocketed.

During 1985–1994, to the beginning of my Ph.D. at M.I.T., I participated in numerous pattern recognition research projects. Swamy and I would talk from time to time on the phone, and I would keep him abreast on my work at MIT. He was always so very supportive, encouraging and uplifting. He was always positive and always ready to help. Swamy shared with me only a bit of his personal history. All I knew is that he had been in Holland for sometime, had gotten married to a Dutch woman, and had children. His moral support of my research activities helped me in my research pursuits. Among the projects I was involved in included: handwriting recognition, ultrasonic wave analysis, document analysis, image flow visualization, and a number of other pattern analysis projects.

From these various pattern analysis projects, there appeared to be a common set of strategies and methods that could be abstracted across all fields. That thought led to my Ph.D. thesis at M.I.T. entitled *Information Cybernetics*. In 1993, while in the middle of my Ph.D. work, I was invited to participate in a very interesting pattern analysis competition. This competition was not scientific in origin; it was from the U.S. Government. In particular, the Executive Office of the White House, with then President Bill Clinton, was looking for intelligent ways using computers to sort their E-Mail. At that time, the White House was receiving nearly

5,000 E-Mail per day, and this was before the introduction of the World Wide Web. There were 147 different categories. Student interns at the White House were manually reading each E-Mail and assigning them into one of the 147 categories. Categories included drugs, education, death threats, etc. The White House was interested in automatic filtering and sorting of E-Mail for two reasons.

I approached the White House competition with little knowledge natural language processing. My approach was engineering, where I used a hybrid method of employing nearly 19 different methods spanning feature extraction, clustering, to supervised and unsupervised learning. I was the only graduate student involved. The other five competitors were private and publicly traded companies. I won the competition.

I took time off in M.I.T. in 1994 to start EchoMail, a company for pattern analysis of electronic mail. We grew the company to nearly 300 employees worldwide by year 2000. I remember hearing from Swamy once during that time asking me for a letter of recommendation as he was taking a new job in the Midwest. It was weird for me to write a letter of recommendation for him. He had written several for me many decades before. Following that interaction I had not heard from him. It was my mom who a few years later informed me of his passing.

Hearing of Swamy's passing was sad. He had always been there and now was no more. I wondered who was with him, what his life had been during those past 10 years when I had lost touch. I recall Swamy and I sharing at our first dinner conversation thoughts about ancient Indian science, its mystical and scientific aspects, including the concept of Soul. There was an unstated agreement that Soul never dies. As I heard about Swamy's passing, I flashed back to that dinner conversation and knew his spirit would and had never died.

As I look back from the time of my meeting Swamy till today, 2010, my entire history of work, particularly my success with EchoMail, development of CytoSolve, and my recent research at M.I.T. to discover patterns of coherence that bridge traditional systems of medicine with modern systems biology, can all be traced to that dinner conversation with Swamy. As I write this last sentence, I am deeply moved and wish Swamy was here so we could have dinner again and I could thank him for the wondrous gift he gave a 16-year old nearly 30 years ago.

References

1. Hood L, Heath JR, Phelps ME, Lin B. Systems biology and new technologies enable predictive and preventative medicine. Science. 2004;306:640–3.
2. Ideker T, Lauffenburger D. Building with a scaffold: emerging strategies for high- to low-level cellular modeling. Trends Biotechnol. 2003;21:255–62.
3. Kitano H. Computational systems biology. Nature. 2002;420:206–10.
4. Palsson BO, Price ND, Papin JA. Development of network-based pathway definitions: the need to analyze real metabolic networks. Trends in Biotechnology. 2003;21:195–8.
5. Tomita M, Hashimoto K, Takahashi K, Shimizu TS, Matsuzaki Y, Miyoshi F, Saito K, Tanida S, Yugi K, Venter JC, Hutchison CA 3rd. E-CELL: software environment for whole-cell simulation. Bioinformatics. 1999;15:72–84.

6. Pennisi E. A low number wins the GeneSweep pool. Science. 2003;300:1484.
7. Hodgkin J. What does a worm want with 20, 000 genes? Genome Biology. 2001;2:1–4.
8. Putnam NH, Srivastava M, Hellsten U, Dirks B, Chapman J, Salamov A, Rokshar DS. Sea anemone genome reveals the gene repertoire and genomic organization of the Eumetazoan ancestor. Berkeley: Lawrence Berkeley National Laboratory; 2007.
9. Peri S, Navarro JD, Amanchy R, Kristiansen TZ, Jonnalagadda C, Surendranath V, Niranjan V, Muthusamy B, Gandhi TKB, Gronborg M, Ibarrola N, Deshpande N, Shanker K, Shivashankar HN, Pandey A. Development of human protein reference database as an initial platform for approaching systems biology in humans. Genome Research. 2003;13:2363–71.
10. Laue Mv. Kritische Bemerkungen zu den Deutungen der Photoframme von Friedich und Knipping. Physikalische Zeitschrift. 1913;14:421–3.
11. Patwardhan B, Warude D, Pushpangadan P, Bhatt N. Ayurveda and traditional Chinese medicine: a comparative overview. Oxford Journals Medicine Evidence-based Complementary and Alternative Medicine. 2005;2:465–73.
12. Subbarayappa BV. Siddha medicine: an overview. Lancet. 1997;350:1841–4.
13. Cannon WB. The wisdom of the body. New York: Norton; 1933.
14. Wiener N. Cybernetics or control and communication in the animal machine. Cambridge: The MIT Press; 1948.
15. Watson JD, Crick FH. Molecular structure of nucleic acids: a structure of deoxyribose nucleic acid. Nature. 1953;171:737–8.
16. Kitano H. Perspectives on systems biology. New Generation Computing. 2000;18:199–216.
17. Kitano H. Foundations of systems biology. Cambridge: The MIT Press; 2001.
18. Cuellar AA, Lloyd CM, Nielsen PF, Bullivant DP, Nickerson DP, Hunter PJ. An overview of CellML 1.1, a biological model description language. SIMULATION. 2003;79:740–7.
19. Le Novere N, Bornstein B, Broicher A, Courtot M, Donizelli M, Dharuri H, Li L, Sauro H, Schilstra M, Shapiro B, Snoep JL, Hucka M. BioModels database: a free, centralized database of curated, published, quantitative kinetic models of biochemical and cellular systems. Nucleic Acids Res. 2006;34:D689–91.
20. Snoep JL, Bruggeman F, Olivier BG, Westerhoff HV. Towards building the silicon cell: a modular approach. Biosystems. 2006;83:207–16.
21. Raczynski S. Differential inclusions in system simulation. Transactions of the Society for Computer Simulation. 1996;13:47–54.
22. Bulatwicz TF. Support for model coupling: an interface-based approach. Eugene: Department of Computer and Information Science, University of Oregon; 2006. p. 216.
23. Krueger CW. Software reuse. ACM Computing Surveys (CSUR). 1992;24:131–83.
24. Rajlich V, Wilde N. The role of concepts in program comprehension. In: 2002 international workshop on program comprehension. Los Alamitos, CA: IEEE Computer Society Press; 2002. p. 271–8.
25. Robinson S, Nance RE, Paul RJ, Pidd M, Taylor SJE. Simulation model reuse: definitions. benefits and obstacles., Simulation Modelling Practice and Theory. 2004;12:479–94.
26. Gianchandani EP, Brautigan DL, Papin JA. Systems analyses characterize integrated functions of biochemical networks. Trends in Biochemical Sciences. 2006;31:284–91.
27. Palsson B. Two-dimensional annotation of genomes. Nat Biotechnol. 2004;22:1218–9.
28. Papin JA, Hunter T, Palsson BO, Subramaniam S. Reconstruction of cellular signalling networks and analysis of their properties. Nat Rev Mol Cell Biol. 2005;6:99–111.
29. Hunter P, Borg T. Integration from proteins to organs: the Human Physiome project. Nature Reviews Molecular Cell Biology. 2003;4:237–43.
30. Hood L, Perlmutter RM. The impact of systems approaches on biological problems in drug discovery. Nat Biotechnol. 2004;22:1215–7.
31. Aderem A. Systems biology: its practice and challenges. Cell. 2005;121:511–3.
32. Takahashi K, Kaizu K, Hu B, Tomita M. A multi-algorithm, multi-timescale method for cell simulation. Bioinformatics. 2004;20:538–46.
33. Cerami EG, Bader GD, Gross BE, Sander C. cPath: open source software for collecting, storing, and querying biological pathways. BMC Bioinformatics. 2006;7:497.

34. Liu ET. Systems biology, integrative biology, predictive biology. Cell. 2005;121:505–6.
35. Hwang D, Smith JJ, Leslie DM, Weston AD, Rust AG, Ramsey S, de Atauri P, Siegel AF, Bolouri H, Aitchison JD, Hood L. A data integration methodology for systems biology: experimental verification. Proc Natl Acad Sci USA. 2005;102:17302–7.
36. Endy D, Brent R. Modelling cellular behaviour. Nature. 2001;409:391–5.
37. Sauro HM, Hucka M, Finney A, Wellock C, Bolouri H, Doyle J, Kitano H. Next generation simulation tools: the systems biology Workbench and BioSPICE integration. Omics. 2003;7:355–72.
38. Lindon JC, Holmes E, Nicholson JK. Metabonomics techniques and applications to pharmaceutical research & development. Pharm Res. 2006;23:1075–88.
39. Pecou E. Splitting the dynamics of large biochemical interaction networks. J Theor Biol. 2005;232:375–84.
40. Pennisi E. How will big pictures emerge from a sea of biological data? Science. 2005;309:94.
41. Bader JS, Chant J. Systems biology. When proteomes collide, Science. 2006;311:187–8.
42. Arkin AP, Fletcher DA. Fast, cheap and somewhat in control. Genome Biol. 2006;7:114.
43. Vaidehi N, Goddard WA. Atom-level simulation and modeling of biomacromolecules. In: Bower JM, Bolouri H, editors. Computational modeling of genetic and biochemical networks. Cambridge: MIT Press; 2001. p. 161–3.
44. White J. Two protein interactions are intractable using molecular dynamics. In: Ayyadurai S, editor. Personal communication. Cambridge; 2007.
45. Stultz C. Intractability of using atom-by-atom molecular dynamics for modeling biological pathways. In: Ayyadurai S, editor. Personal communication. Cambridge; 2007.
46. Keller EF. A clash of two cultures. Nature. 2007;445:603.
47. Noble D. Systems biology and the heart. Biosystems. 2006;83:75–80.
48. Duarte NC, Becker SA, Jamshidi N, Thiele I, Mo ML, Vo TD, Srivas R, Palsson BO. Global reconstruction of the human metabolic network based on genomic and bibliomic data. Proc Natl Acad Sci USA. 2007;104:1777–82.
49. Bhalla US. Understanding complex signaling networks through models and metaphors. Prog Biophys Mol Biol. 2003;81:45–65.
50. Hornberg JJ, Bruggeman FJ, Westerhoff HV, Lankelma J. Cancer: a systems biology disease. Biosystems. 2006;83:81–90.
51. Klipp E, Liebermeister W. Mathematical modeling of intracellular signaling pathways. BMC Neurosci. 2006;7(Suppl 1):S10.
52. Asthagiri AR, Lauffenburger DA. Bioengineering models of cell signaling. Annu Rev Biomed Eng. 2000;2:31–53.
53. Ma'ayan A, Jenkins SL, Neves S, Hasseldine A, Grace E, Dubin-Thaler B, Eungdamrong NJ, Weng G, Ram PT, Rice JJ, Kershenbaum A, Stolovitzky GA, Blitzer RD, Iyengar R. Formation of regulatory patterns during signal propagation in a Mammalian cellular network. Science. 2005;309:1078–83.
54. Lauffenburger DA. Cell signaling pathways as control modules: complexity for simplicity? Proc Natl Acad Sci USA. 2000;97:5031–3.
55. Bassingthwaighte JB, Chizeck HJ, Atlas LE, Qian H. Multiscale modeling of cardiac cellular energetics. Ann N Y Acad Sci. 2005;1047:395–424.
56. Alon U. Biological networks: the tinkerer as an engineer. Science. 2003;301:1866–7.
57. Kitney R, Dollery C. Systems biology: a vision for engineering and medicine. In: Klipp E, Liebermeister W, editor. Engineering. 2007.
58. Oltvai ZN, Barabasi AL. Systems biology. Life's complexity pyramid, Science. 2002;298:763–4.
59. Seeman NC, Belcher AM. Emulating biology: building nanostructures from the bottom up. Proc Natl Acad Sci USA. 2002;99(Supplement 2):6451–5.
60. Morgan JJ, Surovtsev IV, Lindahl PA. A framework for whole-cell mathematical modeling. J Theor Biol. 2004;231:581–96.
61. Sauro HM, Kholodenko BN. Quantitative analysis of signaling networks. Prog Biophys Mol Biol. 2004;86:5–43.

62. Mishra J, Bhalla US. Simulations of inositol phosphate metabolism and its interaction with InsP(3)-mediated calcium release. Biophys J. 2002;83:1298–316.
63. Klipp E, Nordlander B, Kruger R, Gennemark P, Hohmann S. Integrative model of the response of yeast to osmotic shock. Nat Biotechnol. 2005;23:975–82.
64. Hunter P, Smith N, Fernandez J, Tawhai M. Integration from proteins to organs: the IUPS Physiome Project. Mech Ageing Dev. 2005;126:187–92.
65. Alvarez-Vasquez F, Sims KJ, Hannun YA, Voit EO. Integration of kinetic information on yeast sphingolipid metabolism in dynamical pathway models. J Theor Biol. 2004;226:265–91.
66. Oda K, Kimura T, Matsuoka Y, Funahashi A, Muramatsu M, Kitano H. Map of the TLR signaling network. AfCS Research Reports. 2004;2:1–12.
67. Kitano H, Funahashi A, Matsuoka Y, Oda K. Using process diagrams for the graphical representation of biological networks. Nat Biotechnol. 2005;23:961–6.
68. Schulz M, Uhlendorf J, Klipp E, Liebermeister W. SBMLmerge, a system for combining biochemical network models. Genome Inform. 2006;17:62–71.
69. Akarsu E, Fox F, Furmanski W, Haupt T. WebFlow-high-level programming environment and visual authoring toolkit for high performance distributed computing. In: Proceedings of supercomputing '98: high performance networking and computing. IEEE Computer Society; 1998. p. 1–7.
70. Whelan G, Castleton KJ, Buck JW, Hoopes BL, Pelton MA, Strenge DL, Gelston GM, Kickert RN. Concepts of a framework for risk analysis in multimedia environmental systems (FRAMES). In: Laboratory PNN, editor. PNNL-11748. Richland: Pacific Northwest National Laboratory; 1997.
71. Neteler M, Mitasova H. Open source GIS: A GRASS GIS approach. Springer: Boston; 2004.
72. Dewey CF. In: Ayyadurai S, editor. Personal communication; 2006.
73. Quackenbush J, Stoeckert C, Ball C, Brazma A, Gentleman R, Huber W, Irizarry R, Salit M, Sherlock G, Spellman P, Winegarden N. Top-down standards will not serve systems biology. Nature. 2006;440:24.
74. Le Novere N, Finney A, Hucka M, Bhalla US, Campagne F, Collado-Vides J, Crampin EJ, Halstead M, Klipp E, Mendes P, Nielsen P, Sauro H, Shapiro B, Snoep JL, Spence HD, Wanner BL. Minimum information requested in the annotation of biochemical models (MIRIAM). Nat Biotechnol. 2005;23:1509–15.
75. Gilbert D, Fuss H, Gu X, Orton R, Robinson S, Vyshemirsky V, Kurth MJ, Downes CS, Dubitzky W. Computational methodologies for modelling, analysis and simulation of signalling networks. Brief Bioinform. 2006;7:339–53.
76. Brooks F. The mythical man month: essays in software engineering. Reading, MA: Addison Wesley; 1975.
77. Ayyadurai VAS, Dewey CF. CytoSolve: A scalable computational method for dynamic integration of multiple molecular pathway models. Cell Mol Bioeng. 2010; doi: 10.1007/s12195-010-0143-x.
78. Sato M, Taniguchi T, Tanaka N. The interferon system and interferon regulatory factor transcription factors—studies from gene knockout mice. Cytokine & Growth Factor Reviews. 2001;12:133–42.
79. Bocharaov. Mathematical model of antiviral immune response III. Influenza A virus infection. J Theor Biol. 1994;167:323–9.
80. Zi Z, Cho K, Sung M. In silico identification of the key components and steps in IFN-gamma induced JAK-STAT. FEBS. 2005;579:1101–8.
81. Hancioglu, B., Swigon, D., Clermont, G. A dynamical model of human immune response to influenza A virus infection. J Theor Biol. 2007;167:323–60.
82. Yamada S. Computer modeling of JAK/STAT signal transduction pathway. Genome Inform. 2001;12:282–3.
83. Cella M. Maturation, activation, and protection of dendritic cells induced by double-stranded RNA. J Exper Med. 1999;189:821–9.

84. Cooley M. Cytokine activity after human bone marrow transplantation: production of interferons by peripheral blood mononuclear cells from recipients of HLA-Identical sibling bone marrow transplants. J Immunol. 1987;138:3688–94.
85. Takauji. CpG-DNA-induced IFN- production involves p38 MAPKdependent STAT1 phosphorylation in human plasmacytoid dendritic cell precursors. J Luekocyte Biol. 2002;72:1011–19.
86. Kimmel AR, Parent CA. The signal to move: D. discoideum go orienteering. Science. 2003;300:1525–7.
87. Lauffenburger DA. Four M's of systems biology. Cambridge: MIT; 2003.