Proceedings of the 29th Annual International
Conference of the IEEE EMBS
Cité Internationale, Lyon, France
August 23-26, 2007.

A Platform for *in silico* Modeling of Physiological Systems

Toshihiro Kawazu, Masao Nakanishi, Yasuyuki Suzuki, Shinsuke Odai and Taishin Nomura

***Abstract*— This paper describes our challenge of developing base systems necessary and useful for a physiome typed platform for physiological modeling and simulation across multiple scales of time and size through diverse hierarchies of living organisms. Our base systems include a CellML compatible standardized description of structure and dynamic equations (rules for time evolution) of biological organisms in general. A database that can deal with the standardized description and APIs for large scale model constructions and their simulations are also considered.**

I. Introduction

THE integration of modern biosciences and engineering is moving the world towards a new generation of life science where physiological and pathological information from the living human body can be quantitatively described *in silico* across multiple scales of time and size and through diverse hierarchies of organization - from molecules to cells and organs to individuals. The physiome project and systems biology represent such emerging sciences. In particular, the term “physiome” has been coined to represent human structure and function as a whole [1,2]. The challenge is to understand and quantitatively integrate not only structure and function of biological entities such as ion channel proteins and enzymes on a single spatio-temporal scale, but also functional relationships between entities across multiple scales. This integrative approach will eventually allow us to understand the mechanisms underlying biological functions that will emerge through the dynamics of each element and large aggregations of the elements. This means that we will be able to explore *the logics of proteins and cells* through the modeling of nano and microscopic dynamics governed by the first principles of physics, *the logics of cells, organs, and individuals* through the phenomenological modeling of meso and macroscopic dynamics of systems as the aggregation of the nano and microscopic objects, and finally, *the metalogics* that will bridge between the different scales and hierarchies, leading to the establishment of a platform, called *in silico* human. The *in silico* human is designed as both model databases and simulators. It will be released, in the mean time, into the web public domain. The *in silico* human will eventually liberate us from spatio-temporal constraints to freely explore the human body, leading to the qualitative logic necessary to understand human physiology and pathology.

This paper describes a part of our challenge of developing base systems necessary for the *in silico* human, with intending possible cooperations with similar projects such as JSim/XSIM in NSR physiome[3], CellML and its related APIs in IUPS physiome[4], and simBio at Kyoto[5] that are promoting worldwide efforts. Our base systems include a standardized description of structure as well as dynamic equations (rules for time evolution) of biological organisms, database functions based on the standardized description, and APIs for large scale model constructions and their simulations.

Manuscript received April 2, 2007.
T. Kawazu, M. Nakanishi, Y. Suzuki, S. Odai, and T. Nomura are with Graduate School of Engineering Science and The center for advanced medical engineering and informatics at Osaka University, Toyonaka, Osaka, 560-8531 Japan. (phone: +81-6-6850-6532; fax: +81-6-6850-6557;e-mail: kawazu@bpe.es.osaka-u.ac.jp,nakanishi@bpe.es.osaka-u.ac.jp,suzuki@bpe.es.osaka-u.ac.jp,odai@bpe.es.osaka-u.ac.jp,taishin@bpe.es.osaka-u.ac.jp).

II. Outline of the Platform

A. A standardized description; Structure and Dynamics

A target biological organism, either its structure or dynamics is modeled as *a module*, which is a base class object of the developed system. A module can be characterized basically by its *ID number* which is unique throughout the system, *name*, *edges*, *state*, *parameters*, *dynamic-rules* and *morphology* among others. *The edges* are class objects that represent *the structural and functional relationships* among modules. The structural relationships defined in the system are “*include*” and “*constituent*” representing hierarchical or parent-child relationships, and “*attachment*” to represent two or more modules are glued each other. A single or a set of modules may form a module by operating class methods, *encapsulation* and *opening ports* through which two or more modules are *functionally related* with each other by the *functional links*. *Ports* of an encapsulated module are packed in a class object which has properties describing information necessary enough for the use of the module. *The state* and *dynamic-rules* of a module are responsible for dynamics of the module. The *state* may evolve or be updated as time based on the property specified in the *dynamic-rules* which utilize the *states* of other

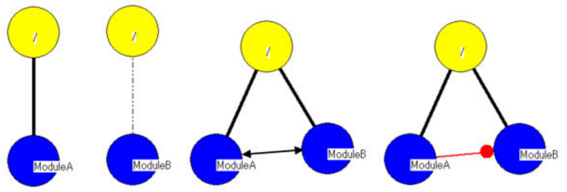


Fig. 1. Diagrammatic representations of structure and functional relationships in the system. From the left to right, it is the “constituent”, “include-like” hierarchical relationships, the “attachment”, and the “functional” relationships. The last two relations do not represent hierarchical structure. The functional relationship has its direction, starting at the non-marked site and terminated at the open circle, along which state update notification signals are transmitted in dynamic simulations.

modules that influence the state update of the module through the *functional links*. See Fig. 2 for the corresponding GUI examples.

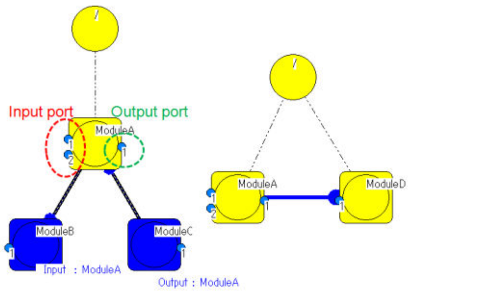


Fig. 2. Encapsulation of modules. The module A in the left panel has internal constitutions (B and C). It is encapsulated with two input ports and one output port. The output port provides a state of either A, B or C. On the right, two encapsulated modules A and D are functionally linked by which the state update of D is influenced. The port properties can be edited to modify and create a new module.

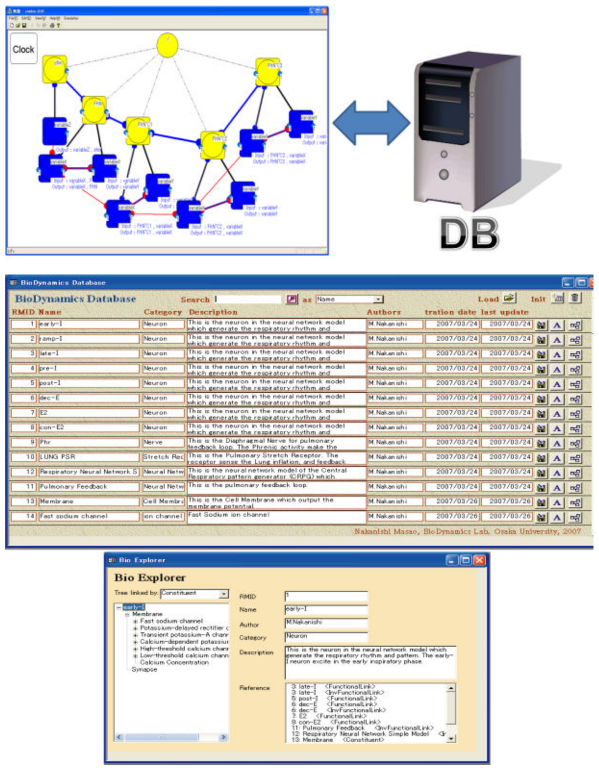
The *parameters* and *morphology* properties of a module specify, respectively, the values and descriptions of constant parameters used for the module and the geometrical shape of the module. The morphological data can be specified by several data format such as vrml and VCAD volume data. It can be used merely for 3D visualization, but may also essentially be used for dynamic simulation.

B. *Database Functions*

Every module will be registered in a database that can deal with all of *the standardized model description* defined above, so that the user can search/find and then download a desired model into his/her PC to simulate and/or reuse the model in his/her modeling and simulations (Fig. 3).

Since a registered module in the database includes all information in *the standardized model description* of the module and more additional metadata such as the original journal references, and the date and name of the registration, once the user found a desired module by "search," one can explore the module fully in detail. When the user creates a new model by modifying a pre-registered model, by connecting multiple pre-registered models to obtain a large scale model, or by originally constructing a brand-new model, one can upload and register it to the database. The model registration and creation policies have not been established, but will be in the mean time at the time of its release in the public domain.

As expected from the model exploring function of the database, it is not used only for the model search and download, but also for construction of a new model. In particular, the database function to search a desired module and examining its port properties and functional links of that module should help the user who tries to gather and connect a number of modules to obtain a new large scale simulation model. In such a case, the API described below cooperates with the database to support the modeling process.



BioDynamics Database

BioDynamics Database

Search as Name Load Init

RMID	Name	Category	Description	Authors	tration date	last update
1	early-I	Neuron	This is the neuron in the neural network model which generate the respiratory rhythm and	M.Nakanishi	2007/03/24	2007/03/24
2	ramp-I	Neuron	This is the neuron in the neural network model which generate the respiratory rhythm and	M.Nakanishi	2007/03/24	2007/03/24
3	late-I	Neuron	This is the neuron in the neural network model which generate the respiratory rhythm and	M.Nakanishi	2007/03/24	2007/03/24
4	pre-I	Neuron	This is the neuron in the neural network model which generate the respiratory rhythm and	M.Nakanishi	2007/03/24	2007/03/24
5	post-I	Neuron	This is the neuron in the neural network model which generate the respiratory rhythm and	M.Nakanishi	2007/03/24	2007/03/24
6	dec-E	Neuron	This is the neuron in the neural network model which generate the respiratory rhythm and	M.Nakanishi	2007/03/24	2007/03/24
7	E2	Neuron	This is the neuron in the neural network model which generate the respiratory rhythm and	M.Nakanishi	2007/03/24	2007/03/24
8	con-E2	Neuron	This is the neuron in the neural network model which generate the respiratory rhythm and	M.Nakanishi	2007/03/24	2007/03/24
9	Phr	Nerve	This is the Diaphragmal Nerve for pulmonary feedback loop. The Phrenic activity make the	M.Nakanishi	2007/03/24	2007/03/24
10	LUNG PSR	Stretch Rec	This is the Pulmonary Stretch Receptor. The receptor sense the Lung inflation, and feedback	M.Nakanishi	2007/03/24	2007/03/24
12	Respiratory Neural Network S	Neural Netw	This is the neural network model of the Central Respiratory pattern generator (CRPG) which	M.Nakanishi	2007/03/24	2007/03/24
11	Pulmonary Feedback	Neural Netw	This is the pulmonary feedback loop.	M.Nakanishi	2007/03/24	2007/03/24
13	Membrane	Cell Membr	This is the Cell Membrane which output the membrane potential.	M.Nakanishi	2007/03/26	2007/03/26
14	Fast sodium channel	ion channel	Fast Sodium ion channel	M.Nakanishi	2007/03/26	2007/03/26

Nakanishi Masao, BioDynamics Lab. Osaka University, 2007

Bio Explorer

Bio Explorer

Tree linked by: Constituent

- early-I
  - Membrane
    - Fast sodium channel
    - Potassium-delayed rectifier c
    - Transient potassium-A chann
    - Calcium-dependent potassiu
    - High-threshold calcium chann
    - Low-threshold calcium chann
    - Calcium Concentration
  - Synapse

RMID	1
Name	early-I
Author	M.Nakanishi
Category	Neuron
Description	This is the neuron in the neural network model which generate the respiratory rhythm and pattern. The early-I neuron excite in the early inspiratory phase.
Reference	3: late-I <FunctionalLink> 3: late-I <InvFunctionalLink> 5: post-I <FunctionalLink> 6: dec-E <FunctionalLink> 6: dec-E <InvFunctionalLink> 7: E2 <FunctionalLink> 8: con-E2 <FunctionalLink> 11: Pulmonary Feedback <InvFunctionalLink> 12: Respiratory Neural Network Simple Model <I 13: Membrane <Constituent>

Fig. 3. The model database. Every module will be registered in a database that can deal with all of *the standardized model description* defined above, so that the user can search/find and then download a desired model into his/her PC to simulate and/or reuse the model in his/her modeling and simulations.

III. Model construction: Examples

A. *Construction of a simple differential equation model*

This subsection exemplifies a typical procedure of model construction. The FHN model, a well known simple model of excitable membrane, is used for this purpose. Fig. 4 visualizes the procedure in which a diffusively coupled several FHN models is newly constructed in our module framework. More precisely, the following coupled ordinary differential equations are constructed in the system;

$$\frac{dx_i}{dt} = c\left(x_i - \frac{x_i^3}{3} - y_i + z_i\right) + \sum_{j \in N(i)} d_{ij}(x_j - x_i) \qquad (1)$$
$$\frac{dy_i}{dt} = \frac{1}{c}(x_i - by_i + a)$$

where x_i and y_i represent, respectively, the membrane potential and refractoriness of the i-th excitable membrane model. z_i represents a current stimulation applied to the membrane. The last summed-up term with a diffusion constant d_{ij} of the first line of Equation (1) represents the diffusive interactions between the adjacent cell models (i-th and j-th cell models), which is absent in every single membrane model. There are several ways how we define modules to do this task. Here the dynamic variables of the FHN, namely x_i and y_i,

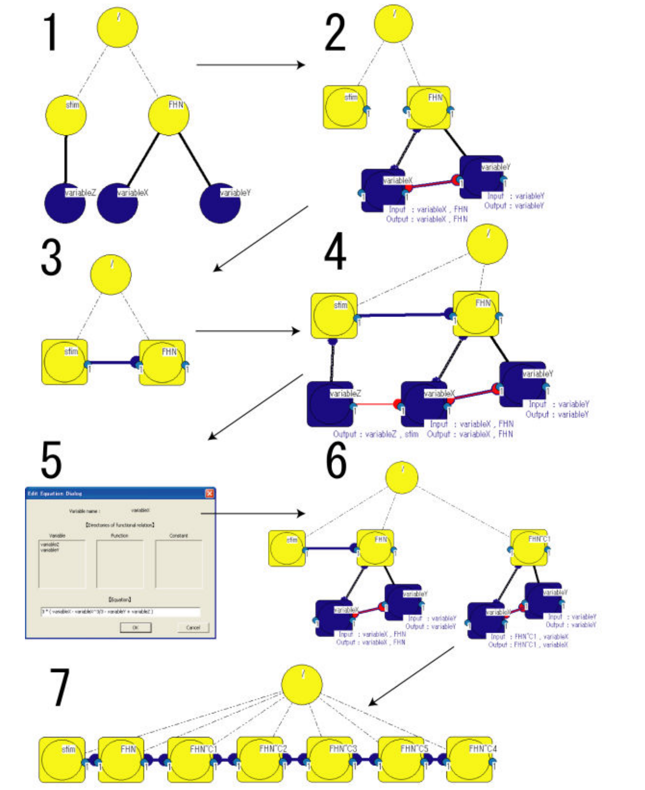


Fig. 4. An example of procedures to construct a simple model described by a set of ordinary differential equations.

and z_i are defined as the elementary modules as shown in the procedure 1 (three blue circles). The right and left modules on the second hierarchy represent, respectively, the single FHN module which consists of two modules as x_i and y_i and the stimulation module as z_i. In 2, *input and output ports* of the FHN and the stimulation modules, as well as those of the elementary modules x_i, y_i and z_i, are defined, through which the elementary modules at the third hierarchy may be *functionally linked* to other modules outside. Each of the FHN and the stimulation modules is encapsulated with those *ports*. The encapsulated modules are now displayed by squares as shown in 2. Moreover, the modules x_i and y_i are *functionally linked* through their *input/output ports* as they are interacting variables. In 3, the stimulation and the single FHN modules are *functionally linked* via their *input/output ports*, by which the *functional link* between the z_i module and x_i module is automatically established as shown in 4. In 5, the property *dynamic-rules* of each of the modules x_i, y_i and z_i are edited using a pop-up window in which the *state* of every functionally linked modules is already listed to be used in the *dynamic-rules*. At this point, one can execute a dynamic simulation of the single FHN model to obtain action potentials using the API in this modeling procedure. One can also register the model into the database. A structured data necessary for the database registration can also be produced by the API. As the continuation of the modeling procedure so

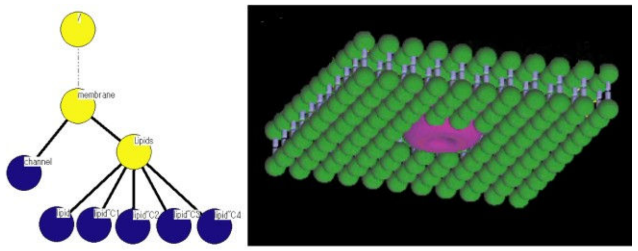


Fig. 5. A multi-agent type model of a cell membrane with biophysical morphology. Structure of the model, involving the channel and lipid bi-layer, is shown on the left panel, where the model description is the same as the case for the model with differential equations. The 3D morphology of the membrane shown on the right panel is obtained by setting a *morphology* data into the channel and every lipid. Note that the number of the lipids on the left and right does not correspond just for a display space reason.

far, or after downloading the model constructed here from the database, the diffusively coupled FHN models can easily be constructed. It can be done by copying the single FHN module to reach the situation shown in 6. Repeating the copy, *functionally linking* between the adjacent FHN modules, and editing the *dynamic-rules* property as in the procedure 5 will lead to the construction and simulation of the coupled system described in Equation (1). Note that we can employ another way of modeling for the corresponding coupled system, where the diffusive coupling term is defined as the module. Then only copying the coupling module without editing *dynamic-property* of the FHN module is enough to obtain the coupled FHN system. The coupled FHN system can also be registered in the database as in the single FHN case. Note also that, if we employ the way that uses the coupling module and no modification of the *dynamic-rules* of the single FHN module, the registered data for the coupled FHN system maintains the registered data for the single FHN so that the user can search the single FHN and pick it up from the data for the coupled FHN to reuse it for other modeling purposes.

In addition to the modeling procedure described here, the system has an API that can parse CellML formatted model descriptions into the system, by which the model's structure can be represented as in Fig. 4 and one can simulate the model's dynamics by just pressing an executing button.

B. *Construction multi-agent type model with morphology*

Another type of modeling example is shown in Fig. 5, where a multi-agent type model of a cell membrane with biophysical morphology is considered. In this example, physical structure of the model, involving the channel protein and the lipid bi-layer, is modeled by using our standardized description. The 3D morphology of the membrane shown on the right panel of Fig. 5 is obtained by setting a *morphology* property into modules representing the channel protein and every lipid. By introducing a number of ions as modules, and setting the *dynamic-rules* describing the positional change of each ion as well as the changes in the permeability of the channel for the ion, we can simulate dynamics of a single channel current experiment in the electrophysiology of cell membranes. (Figures are not shown for this example.)

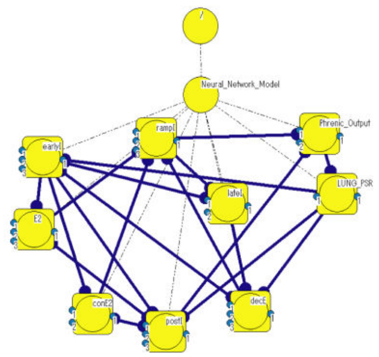


Fig. 6. Basic architecture of a motor pattern generating neural network model. Every neuron model in the network is encapsulated module. Functional links representing synaptic connections among the neuron models enable the network to behave as the pattern generator. 32 sets of this network are used to construct a relatively large scale model of the spinal neural network.

IV. Application to Neural Network Simulation

This section is devoted to describe a relatively complete example of the use of our platform. The platform is applied to construction of a spinal neural network that can reproduce a rhythmic motor pattern generation. The neural network model consists of hundreds of biophysical spinal interneurons and their electrical excitation pattern includes alternating burst-like action potential generations representing left and right as well as flexor and extensor coordination. The single interneuron models and the basic network architecture are borrowed from the published literatures[6,7]. The basic neural network consists of seven interneurons and two peripheral sensory systems. Each interneuron model is described by 29 sets of ordinary differential equations, and it is constructed using our API as in the case for the FHN, and then encapsulated with the three input *ports* and one output *port*. After simulating each of the single neuron models to examine it behaves appropriately, they are registered into the database.

The single neuron model and the models of peripheral sensory systems are used to construct the basic neural network model. The synaptic connections among the neurons are easily established by linking the input and output *ports* defined for the registered modules using the API. Fig. 7 shows simulated dynamics, which reproduced the alternating burst activities of the neurons. This example shows that the developed system can facilitate model constructions efficiently.

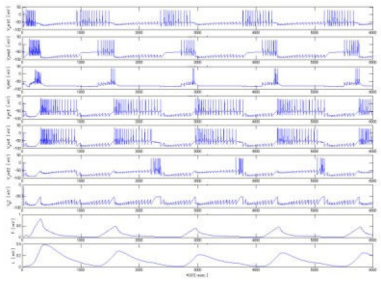


Fig. 7. Simulated alternating bursting activities of seven inter neurons and the peripheral sensory systems in the basic neural network model. Seven traces from the top are the action potentials of the neuron models.

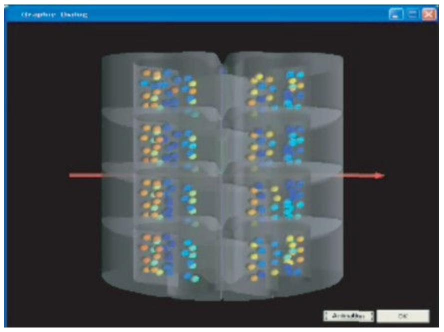


Fig. 8. The simulation result of the model which had three-dimensional structure. The many neurons (sphere) are arranged in the spinal cord.

The base neural network is then also encapsulated with several input and output *ports* to construct a spinal network model of larger size. A morphology property including a shape of each single neuron and its 3D position are given, and 32 copies of the base neural network are prepared, and they are spatially arranged in the spinal segments as shown in Fig. 8. Each spinal segment is divided into the left and right sub-segments, and each sub-segment has 4 basic neural networks. The basic neural networks at the left and right sub-segments are connected by reciprocal inhibitions, and those arranged in the vertical directions are connected by uni-directed inhibitions. In this case, since the dimension of the model is relatively large, the dynamic simulation of the model was performed on a PC cluster with 32 nodes using MPI. Simulated dynamics of the large network model can be visualized as in Fig. 8, where the membrane potentials of the neurons are colored coded.

V. Conclusion

Base systems necessary for a physiome typed platform for physiological modeling and simulation are developed. Several examples of model constructions and their dynamic simulation as well as the database registration and reuse of the registered model are described to show the developed system can be used as a platform in the physiome typed project.

References

[1] James B. Bassingthwaighte, *Annals of Biomedical Engineering*, Vol. 28, pp. 1043-1058, 2000
[2] Peter J. Hunter and Thomas K. Borg. Integration from proteins to organs: The Physiome Project. *Nature Review Mol Cell Biol*, 4(3), pp. 237-243, 2003
[3] JSim [Online]. Available: http://physiome.org/jsim/index.html. (URL)
[4] CellML API. [Online]. Available: http://www.cellml.org (URL)
[5] simBio. [Online]. Available: http://www.sim-bio.org (URL)
[6] I A Rybak, J F Paton, and J S Schwaber. "Modeling neural mechanisms for genesis ofrespiratory rhythm and pattern." I. Models of respiratory neurons. J Neurophysiol, Vol. 77, No. 4, pp. 1994, 2006, Apr 1997.
[7] I A Rybak, J F Paton, and J S Schwaber. "Modeling neural mechanisms for genesis of respiratory rhythm and pattern. II. Network models of the central respiratory pattern generator." J Neurophysiol, Vol. 77, No. 4, pp. 2007, 2026, Apr 1997.