Neural Networks 24 (2011) 693–698

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Simulation Platform: A cloud-based online simulation environment

Tadashi Yamazaki [a], Hidetoshi Ikeno [b], Yoshihiro Okumura [c], Shunji Satoh [d], Yoshimi Kamiyama [e], Yutaka Hirata [f], Keiichiro Inagaki [g], Akito Ishihara [h], Takayuki Kannon [i], Shiro Usui [c,g,i,*]

[a] *RIKEN BSI-TOYOTA Collaboration Center, RIKEN Brain Science Institute, 2-1 Hirosawa, Wako, Saitama 351-0198, Japan*
[b] *School of Human Science and Environment, University of Hyogo, 1-1-12 Shinzaike-Honcho, Himeji, Hyogo 670-0092, Japan*
[c] *Neuroinformatics Japan Center, RIKEN Brain Science Institute, 2-1 Hirosawa, Wako, Saitama 351-0198, Japan*
[d] *Graduate School of Informatics Systems, University of Electro-Communications, 1-5-1 Chofugaoka, Chofu, Tokyo 182-8585, Japan*
[e] *School of Information Science and Technology, Aichi Prefectural University, 1522-3 Kumabari-Ibaragabasama, Nagakute, Aichi 480-1198, Japan*
[f] *Faculty of Engineering, Chubu University, 1200 Matsumoto-cho, Kasugai, Aichi 486-8501, Japan*
[g] *Computational Science Research Program, RIKEN, 2-1 Hirosawa, Wako, Saitama 351-0198, Japan*
[h] *School of Information Science and Technology, Chukyo University, 101 Tokodachi Kaizu-cho, Toyota, Aichi 470-0393, Japan*
[i] *Laboratory for Neuroinformatics, RIKEN Brain Science Institute, 2-1 Hirosawa, Wako, Saitama 351-0198, Japan*

A R T I C L E I N F O

Article history:
Received 29 December 2010
Revised and accepted 14 June 2011

Keywords:
Neuroinformatics
Online simulation
Cloud computing
Virtual machine

A B S T R A C T

For multi-scale and multi-modal neural modeling, it is needed to handle multiple neural models described at different levels seamlessly. Database technology will become more important for these studies, specifically for downloading and handling the neural models seamlessly and effortlessly. To date, conventional neuroinformatics databases have solely been designed to archive model files, but the databases should provide a chance for users to validate the models before downloading them. In this paper, we report our on-going project to develop a cloud-based web service for online simulation called "Simulation Platform". Simulation Platform is a cloud of virtual machines running GNU/Linux. On a virtual machine, various software including developer tools such as compilers and libraries, popular neural simulators such as GENESIS, NEURON and NEST, and scientific software such as Gnuplot, R and Octave, are pre-installed. When a user posts a request, a virtual machine is assigned to the user, and the simulation starts on that machine. The user remotely accesses to the machine through a web browser and carries out the simulation, without the need to install any software but a web browser on the user's own computer. Therefore, Simulation Platform is expected to eliminate impediments to handle multiple neural models that require multiple software.

1. Introduction

Higher-order brain functions emerge from the hierarchically organized network composed of a massive number of neurons mutually connected, not from individual neurons acting independently. That is, even if each element is a simple computational unit, the connected network can exhibit greater computational capability. The same argument seems valid for neural models, which are developed at various levels from molecule to system, from milliseconds to hours, and from reflex to cognition. Even if each model functions little, the integrated model could demonstrate such higher-order functions. As multi-scale and multi-modal neural modeling, we focus on building an integrated model by combining multiple models together.

To date, a number of neural models at different scales and modalities have been proposed, and many of them are archived and available at neuroscience databases such as ModelDB[1] (Hines, Morse, Migliore, Carnevale, & Shepherd, 2004) and J-Node Platforms[2] (Usui et al., 2008). The latter is composed of eight databases and is organized by Neuroinformatics Japan Center, which is the Japan Node (J-Node) of the International Neuroinformatics Coordinating Facility. J-Node Platforms serve as online databases, and a companion project PLATO (Kannon, Inagaki, Kamiji, Makimura, & Usui, 2011; Usui, 2010) aims to provide the scheme of model integration.

* Corresponding author at: Neuroinformatics Japan Center, RIKEN Brain Science Institute, 2-1 Hirosawa, Wako, Saitama 351-0198, Japan. Tel.: +81 48 467 7491; fax: +81 4467 7498.
E-mail address: usuishiro@riken.jp (S. Usui).

1 http://senselab.med.yale.edu/modeldb/.
2 http://neuroinf.jp/.

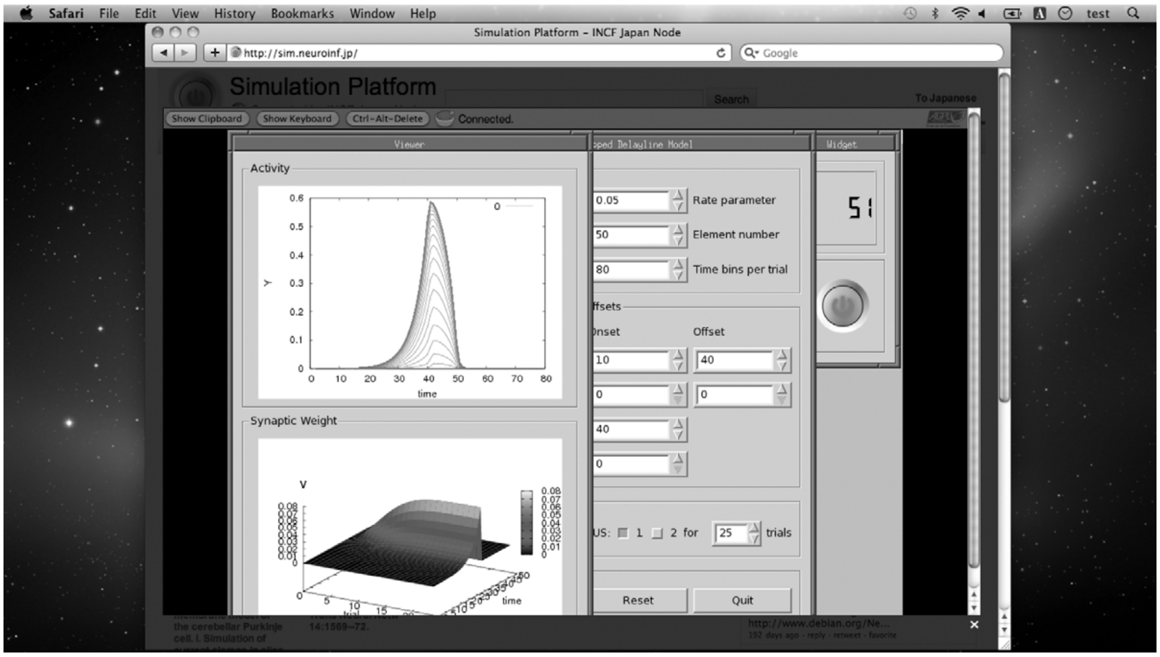

Fig. 1. A screenshot of a web browser during a simulation of a tapped delayline model for cerebellar timing mechanisms (Moore et al., 1989). The model script is found at Cerebellar Platform.[3]

For the integration, we have to validate that all component models of an integrated model work properly beforehand. To validate a component model, we have to download it from a database, extract, read instructions, compile if a model is written in a general programming language such as C, install an appropriate neural simulator if a model is written for a simulator such as GENESIS (Bower & Beeman, 1998), NEURON (Carnevale & Hines, 2006) and NEST (Gewaltig & Diesmann, 2007), and finally we are ready to carry out a computer simulation. We have to take these tedious steps for each component model, even for just a simple test. We need a better way to test a model effortlessly. Once we develop such a system that allows us to validate each component model online, we will be able to search, validate, download component models, and build an integrated model by putting them together seamlessly.

Here, we introduce our on-going project on launching a cloud-based web service called "Simulation Platform",[4] as one of the J-Node Platforms (Usui et al., 2010, 2009). Using Simulation Platform, a user can carry out a model simulation and data analysis without the need to install any software on the user's own computer (a screenshot is shown in Fig. 1). The user is simply asked to specify the model script to Simulation Platform using a web browser. Then, the simulation starts on one of virtual machines in the cloud automatically, and the user can control the virtual machine remotely and interactively through the web browser. Hence, Simulation Platform is expected to reduce difficulties in computer simulations.

2. System overview of Simulation Platform

Fig. 2 depicts the overall structure of Simulation Platform. Simulation Platform is composed of a set of computational nodes[5] as backends and an administrative node as a frontend. All nodes run CentOS 5.5, a GNU/Linux-based system.[6] The details of these machines are shown in Table 1(A).

Table 1
Specification of (A) a computational node and (B) a virtual machine.

(A)	
Model	HP ProLiant DL360 G5
CPU	Dual Core Xeon Processor 5160 3 GHz
RAM	8 GB (PC2-5300 FB-DIMM)
HDD	146 GB 2.5 inch SAS × 2 (RAID1)
Network	NC373i Gigabit Adapter × 2
OS	CentOS 5.5 x86_64
Kernel version	2.6.18
(B)	
Model	Generic x86_64 PC
CPU	Generic x86_64 CPU
RAM	1 GB
HDD	4 GB
Network	PCnet-FASTIII Gigabit Adapter
OS	CentOS 5.5 x86_64
Kernel version	2.6.18

Four virtual machines, also based on CentOS 5.5, run on each computational node.[7] For virtualization, we use a free software (Oracle VirtualBox 3.1.8[8]). The details of a virtual machine are shown in Table 1(B). On each virtual machine, most developer tools such as compilers and libraries, popular neural simulators such as GENESIS, NEURON and NEST, and scientific software such as Gnuplot, R and Octave, are pre-installed. Table 2 lists some of the installed software. A virtual machine can execute a computer simulation upon a user's request. The desktop of the virtual machine appears on the screen of the user's web browser via virtual-network-computing (VNC) protocol, through which the user can interact with the virtual machine remotely.

The administrative node accepts a user's request, assigns a virtual machine to the user, and proxies the connection between

3 http://cerebellum.neuroinf.jp/modules/xoonips/detail.php?item_id=350.
4 http://sim.neuroinf.jp/.
5 As of March 2011, we have four computational nodes.
6 http://www.centos.org/.
7 We have 16 virtual machines as of March 2011.
8 http://www.virtualbox.org/.

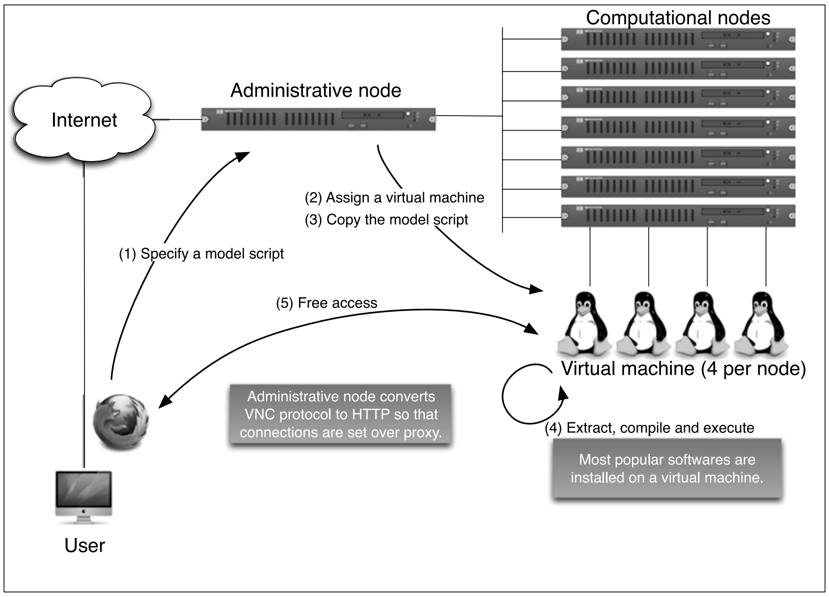
Computational nodes
Administrative node
Internet
(2) Assign a virtual machine
(3) Copy the model script
(1) Specify a model script
(5) Free access
Virtual machine (4 per node)
Administrative node converts VNC protocol to HTTP so that connections are set over proxy.
(4) Extract, compile and execute
Most popular softwares are installed on a virtual machine.
User

Fig. 2. Overall structure of Simulation Platform.

Table 2
Partial list of installed software.

	Name	Version
Programming language	GCC	4.1.2
	OpenJDK	1.6.0
	Python	2.4.3
	Ruby	1.8.5
Library	BLAS	3.0
	FFTW2	2.1.5
	GSL	1.13
	Matplotlib	0.99.1.2
	Numpy	1.2.1
	PyNN	0.6.0
	Scipy	0.6.0
Visualization	Dislin	10.0
	Fiji	20 091 014
	gMocren	4.1.0
	Gnuplot	4.2
	ImageJ	1.0
	Samurai-graph	1.0.7
Data analysis	Octave	3.0.5
	R	2.11.0
	Scilab	5.2.2
Neural simulator	GENESIS	2.3
	Moose-beta	1.2.0
	NEURON	7.1
	NEST	1.9.8558
	Satellite	2.9.5
Other	MCR[a]	2.3

[a] Matlab Compiler Runtime.

the user and the virtual machine. The transaction between the administrative node and the assigned virtual machine is made through a shared folder and network-address-translation (NAT) which are functions of VirtualBox. A shared folder of a virtual machine cannot be accessed from the other virtual machines for security. The administrative node also proxies the VNC transaction between the user's web browser and the assigned virtual machine by translating the VNC protocol to hyper-text-transfer-protocol (HTTP). This allows the user to access the virtual machine via proxy such as Squid Web Proxy Cache.[9] This mechanism is necessary for accessing from security-conscious universities and institutes that block all outbound connections except HTTP. We use a free software for VNC–HTTP translation (guacamole 0.2.6[10]), with a slight modification of the source code to adapt it to our system.

All the transactions between a user and the administrative node are made automatically by a set of custom scripts written in Ruby language (Flanagan & Matsumoto, 2008).

3. The use of Simulation Platform

Fig. 3 shows the diagram of transactions on Simulation Platform in detail. The sequence of the transactions is as follows.

1. A user posts a request to the administrative node to start a simulation of a model. The user uploads the script file of the model or specifies the location of the site that the model is archived.
2. The administrative node receives the request and tries to find a free virtual machine from the virtual machine pool.
3. If all virtual machines are busy, the node returns an error message saying so.
4. If a free virtual machine is found, the administrative node receives the script file of the model from the user, or downloads the file from the site specified by the user.
5. The administrative node copies the script file to the assigned virtual machine through the shared folder. The node also seeks a file called "autorun" for the script file (the details are explained below). If the file is found, it is also copied to the virtual machine.
6. The assigned virtual machine keeps watching the shared folder. When the script file is copied, the virtual machine stops watching and copies the file from the shared folder to the home directory of guest user (on a virtual machine, a computer simulation is executed under the guest account). The virtual machine also seeks the autorun file in the shared folder. If found, the file is executed under a privileged account to prevent the user from interrupting the execution of the file.
7. Simulation starts. The desktop screen of the virtual machine appears on the user's browser. The graphical user interface is

9 http://www.squid-cache.org/.

10 http://guacamole.sourceforge.net/.

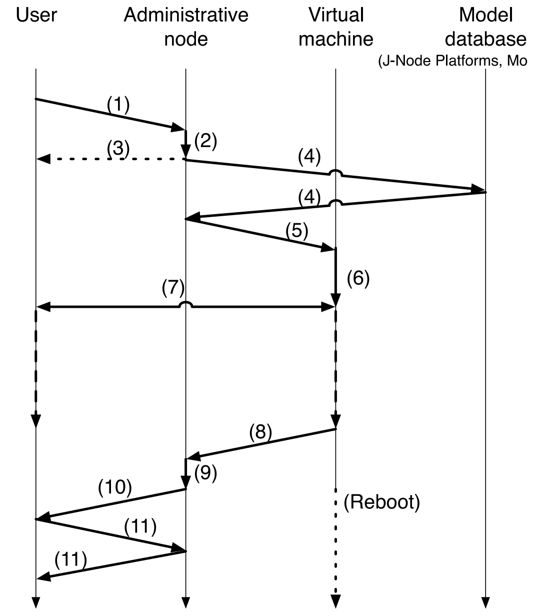

Fig. 3. Diagram of transactions on Simulation Platform. Arrows labeled (1)–(10) represent each of the transactions, which are explained in the text.

implemented by the X Window System.[11] The user can control the virtual machine through the web browser. There is a time limit for the user to use the virtual machine, which is specified for each model in the autorun file. Because the autorun file is executed under a privileged account, the user cannot cancel the time limit.

8. When the virtual machine is timed out or the user clicks the power button shown on the right panel of the screen (Fig. 1), the virtual machine archives all files generated during the simulation and copies the archive file to the shared folder through which the administrative node can access the file.
9. The administrative node removes the virtual machine, creates a new one, and boots it. The new virtual machine does not inherit any traps set by a malicious user. The node also copies the archive file from the shared folder to a publicly open folder that is accessible from a web browser.
10. The administrative node tells the URL of the archive file to the user.
11. The user downloads the archive file and the session is closed.

Fig. 1 represents a screenshot of a web browser running a computer simulation.

3.1. *Auto-start of a simulation*

As explained in the previous section, Simulation Platform provides virtual machines with major software required for a computer simulation. Once a model script is uploaded, Simulation Platform executes a simulation automatically using some of the installed software. However, when it is impossible to identify the software requirements to use, an instruction that describes how to start a simulation of the model has to be prepared a priori.

For this purpose, we provide an "autorun" file, which is a batch file for each model script. An autorun file describes how to deal with the downloaded model script. Fig. 4 is an example of an autorun file, which extracts the downloaded file and an add-on package prepared by the system, compiles the source code, and starts the simulation.

```
#!/bin/sh
# --------------------------------------------
# Simulation Platform Autorun File
# Copyright (C) 2010- RIKEN.
# Some rights reserved.
# --------------------------------------------

echo "Extracting zip file ..."
unzip tapped_delayline.zip

echo "Entering the directory ..."
cd ./tapped_delayline
mv tapped_delayline.c tapped_delayline.c.orig
mv Makefile Makefile.orig

echo "Extracting addons ..."
unzip /share/addons.zip

echo "Compiling the program ..."
make clean
make

./autorun.rb
```

Fig. 4. Autorun file for the tapped delayline model as in Fig. 1.

The administrative node seeks the autorun file after receiving the script file of a model. If found, the administrative node copies the autorun file to the assigned virtual machine as well as the model script. The virtual machine, in turn, seeks the autorun file and executes it if found. Otherwise, the virtual machine invokes a shell terminal and waits the user's command.

We have set up autorun files for most model scripts archived at J-Node Platform. We are preparing autorun files for models archived at other databases such as ModelDB (Hines et al., 2004).[12]

4. Discussion

Multi-scale and multi-modal neural modeling requires the following three steps: collecting models, validating each model, and integrating these models. J-Node has been working on all of them comprehensively by virtue of Neuroinformatics infrastructure and technology, namely, by harvesting J-Node Platforms, Simulation Platform, and PLATO, respectively. In other words, these projects rarely support building a single multi-scale/multi-modal model from the ground up. We, however, do not think this spoils the usefulness of these projects. The core component of Simulation Platform is a virtual machine that equips many software that are necessary for modeling and simulations in neuroscience. Using our virtual machine, scientists can validate each component of an

11 http://www.x.org/.

12 As of March 2011, we have prepared more than 100 autorun files.

integrated model seamlessly and effortlessly without the need to install any software on their own computers. In this way, Simulation Platform eliminates the tedious, time-consuming and sometimes painful installation process of software.

A related project, the Debian Neuroscience Repository (or simply called NeuroDebian[13]) has been launched recently. The project firstly aims to provide binary packages of neuroscience-related software and tools for the Debian GNU/Linux Operating System.[14] The project distributes a virtual machine image of a Debian system that contains many neuroscience-related packages. Their concept is very close to ours, because our virtual machine is a CentOS system containing a number of neuroscience-related packages. We plan to translate the Debian packages that they provide to CentOS packages for our system. Simulation Platform also provides infrastructure and computer resources to execute the virtual machine remotely, which seems not possible for NeuroDebian project. Hence, Simulation Platform may collaborate with NeuroDebian to take advantages of each system in future. Considering accessibility and sustainability of various software, we must install the latest version of software on a virtual machine while preserving older versions of the same software for long-term compatibility. CentOS is capable of managing multiple versions of the same software, so that we can simply install the older packages as well as the latest one.

Simulation Platform relies on only standard HTML5 and Java. Therefore, all modern HTML5-compliant web browsers installed on most computers including tablets and smartphones can be used to access it, while providing the same user experience as long as these browsers are compatible with each other. Owing to the VNC–HTTP translation technology, Simulation Platform can be accessed from any locations where outbound connections to the HTTP port is allowed (and most providers and institutions allow outbound connections to the HTTP port). Therefore, Simulation Platform provides ubiquitous computer resources for online simulation.

We point out that Simulation Platform has tremendous applications not only for multi-scale and multi-modal neural modeling, but also for more general use in various academic fields. One potential application would be a review system of research articles. To date, a reviewer of a modeling paper cannot access the model during the review process. The reviewer has to foresee whether the model works properly and exactly as written in that paper and must be convinced that the result is reproducible, but this is an extremely difficult task. The best solution is to force researchers to open the source code of the model upon submission, but it seems, at least now, very hard to convince them to do so. The source code is everything of the model. Opening the source code may cause a number of side effects such as a takeover of the model by competitors and a violation of copyrights and patents. Simulation Platform may provide a solution. Upon submission, the researcher is asked to register the source code and an autorun file to Simulation Platform. Simulation Platform provides the test environment for the model while preventing access to the source code itself. The reviewer can explore the model dynamics more intensively, whereas the source code is protected from the competitors of the researcher. Although the current Simulation Platform does not have such functionality to hide the source code, we will implement them in near future.

Another potential usage of Simulation Platform would be as a rich online article. To date, an online article is simply an HTML (hyper-text-markup-language) version of the article with hyperlinks to external journal sites for references. Thus, online articles provide little functionality compared with traditional printed articles. Simulation Platform can enhance online articles by adding functionality to play movies, manipulate images, and execute computer simulations on the fly. For example, in the near future, we will be reading articles using handheld computers such as tablet devices, which allow a reader to tap figures in the article. When the reader taps a figure, a web browser opens, connects to Simulation Platform, the simulation that created the figure starts, and the figure is reproduced on the fly. Thus, Simulation Platform may make the content of an article more reliable and attractive.

Simulation Platform aims to provide an instant simulation environment for occasional use. On the contrary, users might want to have a computing environment with much higher performance for heavy use in their laboratories. For this purpose, we are also preparing a grid computing environment and several model programs, which is based on open source software Globus toolkit (Foster, 2006) and Ganglia.[15] We will integrate the grid computing environment with the current Simulation Platform in the future and will provide more flexible computing environment for both occasional and heavy use.

Finally, we would like to describe the tentative schedule of the development of Simulation Platform briefly. In 2011, we plan to officially launch our service. We have started collaboration with J-Node Platforms by offering the online simulation function. We will double the number of VMs by the end of this year. In 2012, we plan to improve the user interface. For example, owing to HTML5 technology, we will implement a feature that changes the desktop size dynamically to be fitted with the screen size, which will be nice for full-screen browsers such as Safari on iPhone and iPad. We will extend our collaboration to the world-wide neuroinformatics databases. We will also double the number of VMs once again. In 2013, we will start to distribute the hosting of computational nodes across various research institutes and universities in the world to scale out the system. At this stage, Simulation Platform will be truly a world-wide distributed service.

We expect that Simulation Platform will further support research, education and publication in neuroscience fields. We have opened a gallery for demonstration.[16] Please visit and try.

Acknowledgments

We would like to thank Nilton Liuji Kamiji for valuable comments on the manuscript. We appreciate reviewers' proposals of new functionality in Simulation Platform.

References

Bower, J., & Beeman, D. (1998). *The book of genesis: exploring realistic neural models with the general neural simulation system* (2nd ed.) New York: Springer-Verlag.

Carnevale, T., & Hines, M. (2006). *The neuron book*. Cambridge University Press.

Flanagan, D., & Matsumoto, Y. (2008). *The Ruby programming language*. O'Reilly Media.

Foster, I. (2006). Globus toolkit version 4: software for service-oriented systems. In *LNCS*: *Vol. 3779*. *IFIP international conference on network and parallel computing* (pp. 2–13).

Gewaltig, M.-O., & Diesmann, M. (2007). Nest (neural simulation tool). *Scholarpedia*, *2*, 1430.

Hines, M., Morse, T., Migliore, M., Carnevale, T., & Shepherd, G. (2004). Model DB: a database to support computational neuroscience. *Journal of Computational Neuroscience*, *17*, 7–11.

Kannon, T., Inagaki, K., Kamiji, N., Makimura, K., & Usui, S. (2011). PLATO: data-oriented approach to collaborative large-scale brain system modeling. *Neural Networks*, in this issue (doi:10.1016/j.neunet.2011.06.011).

Moore, J., Desmond, J., & Berthier, N. (1989). Adaptively timed conditioned responses and the cerebellum: a neural network approach. *Biological Cybernetics*, *62*, 17–28.

13 http://neuro.debian.net/.

14 http://www.debian.org/.

15 http://ganglia.sourceforge.net/.

16 http://sim.neuroinf.jp/.

Usui, S. (2010). PLATO: platform for collaborative brain system modeling. In *Proceedings of 2010 IEEE world congress on computational intelligence. WCCI 2010* (p. plenary talk).

Usui, S., Furuichi, T., Miyakawa, H., Ikeno, H., Nagao, S., Iijima, T., Kamiyama, Y., Isa, T., Suzuki, R., & Ishikane, H. (2008). Japanese neuroinformatics node and platforms. In M. Ishikawa, K. Doya, H. Matsumoto, & T. Yamanaka (Eds.), *LNCS*: *Vol. 4985*. *14th International conference on neural information processing*, ICONIP 2007, (pp. 884–894).

Usui, S., Yamazaki, T., Ikeno, H., Okumura, Y., Satoh, S., Kamiyama, Y., Hirata, Y., Inagaki, K., Ishihara, A., Kannon, T., Kamiji, N., & Akazawa, F. (2010). Simulation platform: model simulation on the cloud. In *Front. neurosci. conference abstract: neuroinformatics 2010*doi:10.3389/conf.fnins.2010.13.00100.

Usui, S., Yamazaki, T., Ikeno, H., Okumura, Y., Satoh, S., Kamiyama, Y., Hirata, Y., Inagaki, K., Kannon, T., Kamiji, N., & Ishihara, A. (2009). Simulation platform: a test environment of computational models via web. In *Front. Neurosci. Conference Abstract: Neuroinformatics 2009*doi:10.3389/conf.neuro.11.2009.08.047.