BIOINFORMATICS

Vol. 19 no. 13 2003, pages 1727–1729
DOI: 10.1093/bioinformatics/btg221

E-Cell 2: Multi-platform E-Cell simulation system

K. Takahashi[1], N. Ishikawa[2], Y. Sadamoto[2], H. Sasamoto[2], S. Ohta[2], A. Shiozawa[2], F. Miyoshi[1], Y. Naito[1], Y. Nakayama[1] and M. Tomita[2,]*

[1]Laboratory for Bioinformatics, Institute for Advanced Biosciences, Keio University, Fujisawa 252-8520, Japan and [2]Bioscience Division, Mitsui Knowledge Industry Co., Ltd., Harmony Tower 21st Floor 32-2, Higashinakano 1-chome, Nakano-ku, Tokyo 164-8721, Japan

Received on December 19, 2002; revised on March 1, 2003; accepted on March 17, 2003

ABSTRACT

Summary: A new version of the E-Cell simulation system, which runs on Windows as well as Linux, has been released as free software under the terms of the GNU General Public License.

Availability: Downloadable from http://www.E-Cell.org

Contact: mt@sfc.keio.ac.jp

The E-Cell system is a generic software package for cell simulation. Given a set of reaction rules and initial values, users can run simulations and observe dynamic changes in quantities and concentrations of intra- and extracellular metabolites and substances through graphical user interfaces (GUI). Activities of biochemical reactions can be monitored, and amounts of substances can be altered (increased/decreased) by the users at any time during the simulation. E-Cell system makes it possible to conduct *in silico* experiments (Tomita *et al*., 1999).

E-Cell is based on an object-oriented modeling theory, structured Substance–Reactor Model (SRM), in which simulation models are constructed with three fundamental object classes, Substance, Reactor and System. Substances represent state variables, Reactors represent operations on the state variables, and Systems represent logical and/or physical compartments containing other objects. Users can define and dynamically load their own subclasses of any of the three fundamental classes in C++ programming language. The newly defined object classes can be used to incorporate new data structures and computational procedures in the simulation models.

The first version of E-Cell (E-Cell 1) was initially developed in 1996 at Keio University. E-Cell 1 is written entirely in C++ and runs on Linux operating system, and has been used for numerous modeling projects. The development of E-Cell version 2 (E-Cell 2) had started in 2000 in order to continue the development over the previous version. One outcome of the development is its portability. Unlike the previous version, which is specialized in Linux, E-Cell 2 compiles and runs on several different platforms such as Windows and Linux. Presently, E-Cell version 2.25 works stably on several operating systems including Windows98, WindowsNT4.0, Windows2000, and WindowsXP.

In order to reduce compiler dependency and optimize simulation performance, internal data structure has been re-designed resulting in a concise and more maintainable implementation. The GUI has been reconstructed in the Java language to provide a uniform GUI environment on different platforms including Windows and Linux. The C++ simulation core and the Java GUI is coupled by using Java Native Interface (JNI).

E-Cell 2 allows the users to incrementally add Reactor subclasses, plug-in modules compute cellular phenomena such as chemical reactions, as Windows Dynamic Link Library (DLL) files. The distributed package of version 2.25 contains 18 different classes of standard Reactors, such as for Michaelis–Menten formula and generalized chemical equilibrium. User-defined reactors can be developed and added with Borland C++ Compiler version 5.5.1. In addition to an interactive GUI mode, E-Cell 2 runs in a batch mode, performing simulations efficiently without a need to synchronize with the GUI (see Figure 1). The reactor definition files and the simulation rule files are 100% compatible with E-Cell 1. In addition to a standard time-value data logging as in the previous version, E-Cell 2 can record max, min, and average value with a user-specified logging interval. This feature enables the users to detect small time-scale fluctuations without causing a big logging overhead. To ensure correctness of the simulation after the thorough reorganization of the simulator kernel, it has been carefully verified that both versions produce completely the same result to machine epsilon.

Despite the development described in this paper, there still remain several difficulties regarding the simulation of realistic cell models (Takahashi *et al*., 2002). Another

*To whom correspondence should be addressed.

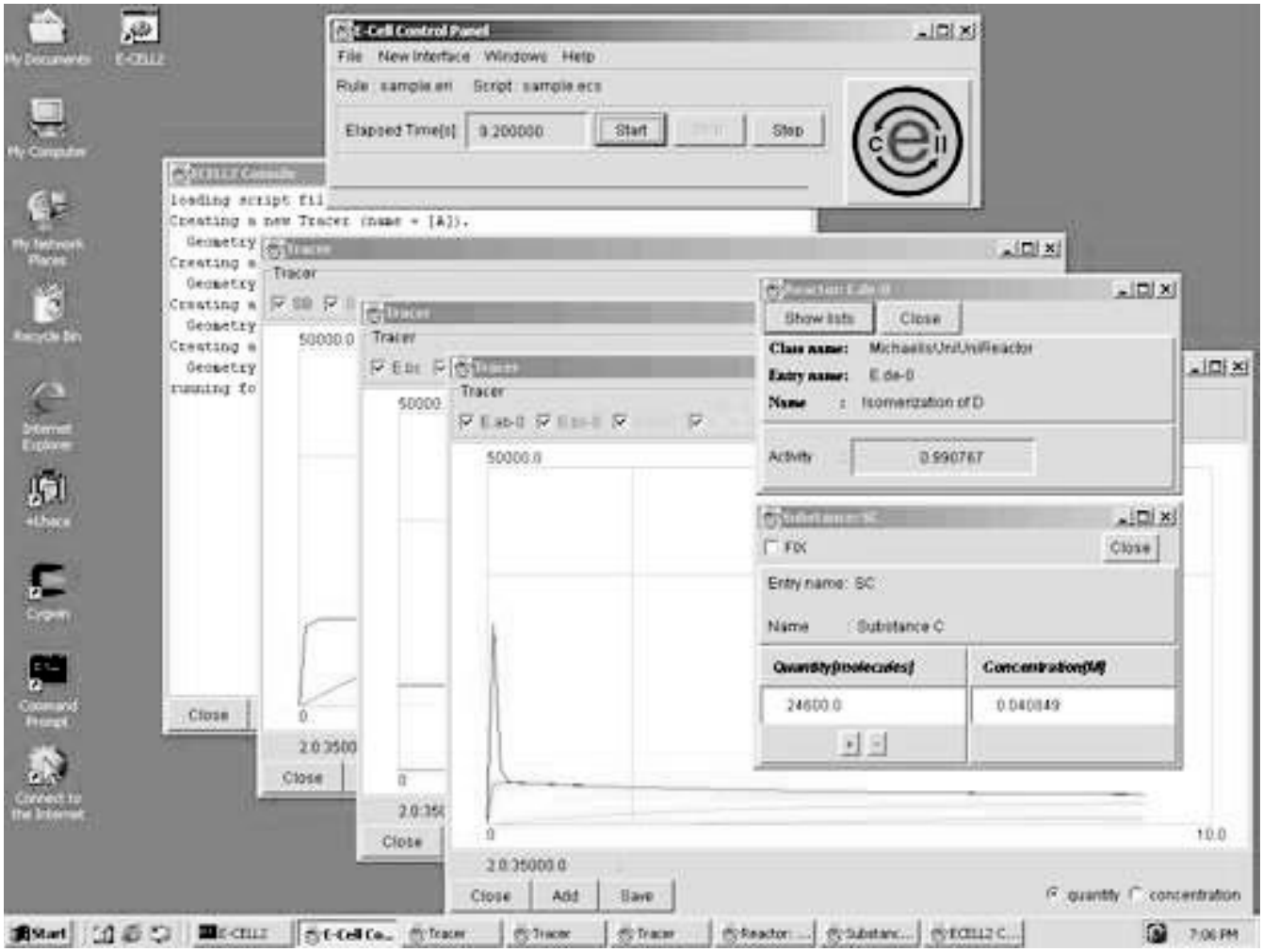

Fig. 1. Snapshot of E-Cell 2 simulation system. Top: the main control panel. Upper middle: a Reactor window showing an activity value of a Reactor object. Lower middle: a Substance window, by which the user can alter the quantity of a Substance. Hiding in the back: the message window, which displays messages from the software. The windows with plots (Tracer windows) present time-series of Substance quantities and Reactor activities. Plots can be magnified on demand.

version of the software, E-Cell Version 3 (E-Cell 3), is concurrently being developed. E-Cell 3 is capable of running various different algorithms simultaneously in a single simulation. Suitable algorithms can be used for different submodels of various cellular phenomena at different levels of abstraction and in different time-scales. The new version provides a compact front end API in C++ and Python, allowing development of front-end software in a modular way. These features of multi-algorithm simulation and software modularity are necessary for large-scale cell simulation (Tomita, 2001, 2002).

E-Cell 2 is distributed as free software under the terms of the GNU General Public License. The package includes an automated installer and a model development environment consisting of several software components such as the Borland C++ Compiler, Java Development Kit, and Cygwin (a UNIX-like shell and library layer for Windows). Several example models including a human erythrocyte metabolism model developed by Nakayama *et al.* are bundled with the package.

ACKNOWLEDGEMENTS

We thank Kenta Hashimoto, Katsuyuki Yugi, Yuri Matsuzaki, Takeshi Sakurada and other members of the E-Cell project; it would have been impossible to develop E-Cell 1 and E-Cell 2 without their comments and feedbacks.

This work was supported by Japan Science and Technology Agency (JST), and in part by the Ministry of Agriculture,

Forestry and Fisheries of Japan (Rice Genome Project SY-2103) and New Energy and Industrial Technology Development and Organization (NEDO) of the Ministry of Economy, Trade and Industry of Japan (Development of a Technological Infrastructure for Industrial Bioprocess Project).

REFERENCES

Tomita,M., Hashimoto,K., Takahashi,K., Shimizu,T., Matsuzaki,Y., Miyoshi,F., Saito,K., Tanida,S., Yugi,K., Venter,J.C. and Hutchison,C. (1999) E-Cell: software environment for whole cell simulation. *Bioinformatics*, **15**(1), 72–84.

Takahashi,K., Yugi,K., Hashimoto,K., Yamada,Y., Pickett,C., and Tomita,M. (2002) Computational challenges in cell simulation. *IEEE Intell. Syst.*, **17**(5), 64–71.

Tomita,M. (2001) Whole cell simulation: a grand challenge of the 21st century. *Trends Biotechnol.*, **19**(6), 205–210.

Tomita,M. (2002) Towards computer aided design (CAD) of useful microorganisms. *Bioinformatics*, **17**, 1091–1092.