PROCEEDINGS OF
THE 2006 INTERNATIONAL CONFERENCE ON
BIOINFORMATICS & COMPUTATIONAL BIOLOGY

BIOCOMP'06

Editors

Hamid R. Arabnia
Homayoun Valafar

Associate Editors

Hikmet Budak, Matthias Dehmer
Frank Emmert-Streib, Faramarz Valafar
Mary Qu Yang, Salim Zabir

Las Vegas, Nevada, USA
June 26-29, 2006
©CSREA Press

SPiC-MAS: a multiagent simulation of stochastic π-calculus models

M. Bourhis, V. Rodin, G. Querrec
European Center for Virtual Reality, Brest, France

Abstract

The biological systems' study is booming. Some scientists use well known mathematical tools such as differential equations while others use graphs or other formalisms. Since the 2000's, new models are more and more used: process algebras. These algebras allow to model the biological systems at the scale of the components. Another individual-centered paradigm, very used in computer science, is the multiagent paradigm. We present in this paper a simulation method of stochastic π-calculus models with a multiagent platform. The implementation of this method led us to the realization of the SPiC-MAS software. The robustness and the adaptability of MultiAgent Systems allow to the user of this software to do genuine experiments in the computing and virtual world: "In Virtuo".

Keywords: stochastic π-calculus; multiagent system; biological simulations; In Virtuo experiment

I. Introduction

To study a real phenomenon, the scientists build a simplified representation of it called "model" . It is however often impossible, physically or for reasons of cost, to confront a model with the reality by experiment. In this work, the studied real phenomena are reactions of chemical or biological nature and the model of these phenomena are scriptural lines of process algebra. The direct confrontation between these two elements is not done immediately. The scientists recourse to simulations then. There are however several means to carry out simulations: the most part of simulation at the present time consist in solving an exact or approximate way equations which underlie the physical models. These simulations are *deterministic*. In this mathematical approach, some equations are enough for example to model the operation of an ecosystem, and their numerical resolution to simulate its evolution. Within the framework of the biological or chemical phenomena, this model allows to describe the kinetics of evolution (creation or disappearance) of a component of the reaction. In these deterministic simulations, the result will always be the same if one initializes the system with the same initial conditions. What allows to notice a simulation of this type, it is the average behavior of the real system. One of the weaknesses of this approach of simulation is its absence of explanatory reach. They note that studied phenomenon converges (or not) towards such or such state but they ignore which are the concerned mechanisms. Contrary to mathematical modelling, *individual-centered* models, or multiagent models, are focused on the means rather than on the aims. Instead of trying to account for the evolution of total parameters, they pose the problem at the scale of the individual. The result of a individual-centered simulation will be the sum of the behaviors of each component or element of the model. If the elements of the model have non-deterministic behaviors (because their behaviors depend on their environment for example), the results will be variable from one simulation to another.
The main objective of this paper is to show a method of simulation of models written in stochastic π-calculus with a multiagent platform. With this intention we will introduce into section II the stochastic π-calculus process algebra which will be the support of description of biochemical models. We will detail in section III the agent paradigm. In section IV, we will present our model and in section V, some applications.

II. π-calculus

The π-calculus [15] is an process algebra. The process algebras are formal languages which allow to make models of distributed computer processes which communicate between them by explicitly named channels and which can be thus synchronized. The ancestors of π-calculus are the algebras CSS (Calculus for Communicating Systems) [14] which allows to represent only the interactive concurrent systems, as the protocols of communication and CSP [12]. But the π-calculus includes in addition the creation of bonds between already existing components. This evolution is called mobility: it allows to model the movement of bonds between the components and also the movement of the components themselves. For that, we will consider the location of a component of an interactive system by the bonds which it has. We determine thus which are the neighbors of this component. If we reason of the kind, the movement or the change of location is represented by the changes of bonds (i.e. these neighbors are not any more the same ones).

Corresponding authors. Address: Centre Européen de Réalité Virtuelle, 25 rue Claude Chappe, BP 38, F-29280 Plouzané, France. Phone: +33 (0)2 98 05 89 78; Fax: +33 (0)2 98 50 89 79. E-mail: {bourhis, rodin, gquerrec}@enib.fr

All in all, π-calculus allows the modelling of the communication of several processes through a network whose topology changes dynamically. They are the processes themselves which change the topology of the network by which they communicate.

A. *Syntax of π-calculus*

The syntax is the whole of the elements of vocabulary which are used in a π-calculus model:

- *processes* : they are indicated by a word beginning with an uppercase letter.
- *names* : These names are in fact simple variables or channels of communication. Their visibility can be total in the model (these are public names) or be restricted to certain processes (these are private names). They are indicated by a word beginning with a lower case.
- *operators* given here by increasing set of priorities:
  - + : non-deterministic choice between processes or actions.
  - . : sequentiality
  - | : parallelism
- *actions* : polyadic π-calculus here will be considered: the content of communications can be just as well one name as a vector composed of several names.
  - emission : $channel$! ($name1$, $name2$, ...). Send on the channel $channel$ the informations $name1$, $name2$, ...
  - reception : $channel$? ($name1$, $name2$, ...). Receives on the channel $channel$ the informations $name1$, $name2$, ...
  - restriction (creation of private names) : $new(name, ...)$. Create a name $name$ belonging to local process and to all his descendants.
  - call : $Process$. Call process $Process$

B. *Operational semantics of π-calculus*

Operational semantics describes the behavior or the evolution of the process. It allows to know what is the result of the interaction between several processes. With this intention, it uses rules of reduction. Main examples are the rule of communication and the rule of choice. Let us consider two processes $P1$ and $P2$:

- *Communication* : if $a\ !\ (y).P1 \mid a\ ?\ (x).\ P2$ then the name y could be communicated by the channel a. The system becomes the following:
  $$P1 \mid P2[x \leftarrow y]^1$$
- *Non-deterministic choice* : let the following system : $A \mid D$ with
  $$\begin{aligned} A &= x!().B + y?().C \\ D &= x?().E + y!().F \end{aligned}$$
  it can be reduced in an equiprobable way in $B \mid E$ or $C \mid F$.

C. *Stochastic π-calculus*

*1) **Limits of the π-calculus**:* The π-calculus as other algebras which we have already named allows to describe a system only in a qualitative way. To integrate quantitative aspects and performances into the process algebras, a parameter is associated at each action. This parameter is a probabilistic distribution. Generally the algebras thus built i.e. the stochastic algebras, are going to follow Gillespie's algorithm example [10]. Each action will have a certain probability of being carried out and, if need be, will require a certain duration to be carried out. Stochastic π-calculus was introduced in 1995 by Priami [18].

*2) **Gillespie's algorithm**:* Gillespie's work concerned the stochastic simulations of chemical reactions. Gillespie observed the reactions at a microscopic scale and studied the collisions between molecules. These collisions are well the founder elements of the chemical reactions. Then he could crossed another scale of observation, the mesoscopic scale. At this scale, the collisions are not observed any more one by one but the average quantity of collisions can just distinguished. Thus, more the average quantity of the collisions is supposed to be large, more one molecule will be likely to collide with other one, and more that will be done quickly. The vision of Gillespie is a probabilistic vision. The collision between two molecules will be a probabilistic law according to the number of molecules. And it is the same reasoning for time. The time during which no collision occurs is also a probabilistic law depending on the quantities of molecules present in the system. Gillespie could define two mathematical rules: one concerns the identification of reaction the molecules which will react and the other concerns the probable quantity of time during which the system will remain stable (when no collision occurs) [10]. Let:

- h_i : the number of distinct combinations between the molecules which can react for the reaction i at a given moment.
- c_i : The kinetic coefficient of the reaction i.
- a_i : the product $h_i * c_i$.
- a_0 : the sum of all a_i.

To determine the next reaction which will occur between elementary quantities of the reactive molecules (these quantities correspond to the stoechiometric coefficients of the reaction), it is necessary to draw a random number and uniformly distributed between 0 and 1. This number is noted r_2. The reaction which will occur is the reaction m such as

$$\textstyle\sum_{i=1}^{m-1} a_i \ < \ r_2 * a_0 \ \leq \ \sum_{i=1}^{m} a_i$$

Secondly, to calculate the period of time linked to this elementary reaction, another random number should be drawn: r_1. The period of time t is:

$$t \ = \ (1/a_0) * ln(1/r_1)$$

[1] $[x \leftarrow y]$ means the replacement of all the occurrences x in P2 by y.

*3) **Stochastic π-calculus:*** Gillespie's work was therefore introduced into π-calculus. But, its application differs: the probability of communication via a channel replaced the probability of a single collision in a chemical reaction. To take into account this new mechanism, it is necessary to add an additional element of syntax: a rate. This rate is a positive real number and is associated with each channel of communication. It corresponds to the coefficient of kinetics of the algorithm of Gillespie. Whereas in π-calculus, all the communications come true with equal chances, in its stochastic version, this rate will be related to the probability of achievement of a communication. The integration of this probabilistic algorithm in π-calculus will serve first to choose the communication which will occur at a given moment among all the possible communications then to calculate the time of realization of this communication.

D. *Application of the stochastic π-calculus at the biology*

Since 2001, stochastic π-calculus is also used for biological modelling. Indeed, the researchers noticed that the biological networks share a great number of functional aspects with the computer networks. Models of computer science (such as networks) enough studied and understood well can give useful proof for biology [21]. It becomes possible to exploit the already existing process algebras to model the biological processes in a formal way. So, every molecules, cells or other biological systems can be seen as processes of π-calculus [22]. These stochastic π-calculus models offer several advantages: the composition and modularity (it is possible to assemble large models from smaller by re-using components) and the permutation between various levels of abstraction (model of whole topological networks or models of the interactions at a molecular scale). Unlike the models of biological systems with differential equations, in this stochastic simulation, each biochemical element which can react is therefore considered individually: the stochastic π-calculus model describes the behavior of one element at the same time and the result is the parallelization of all the behaviors of individuals. Given that each individual evolves of its own way (he has variable chance to interact at a given moment), the simulation of the model will not provide us a unique result. The result of such simulation provides us *one* of the possible behaviors of the modelled system.

Some computer tools were already developed to carry out simulations of models written in stochatic π-calculus. We can quote BioSPi [23] and SPiM [16].

The biochemical models are individual-centered models. They are copied on the models of concurrent and communicating computer processes. Today all these concepts belong to the *agent* paradigm.

III. The agents for the biology

A. *Agents and multiagent system*

The first work on the multiagent system (M.A.S.) dates from 1980s under the influence of Artificial intelligence, bio-sciences, robotics and development of the distributed computing systems.

*1) **Agent:*** The concept of agent covers several acceptances according to research domain and authors. However, all put the emphasis on the concept of autonomous entities endowed with capacities of communication. According to Ferber [8], we call agent a physical or computer entity able to perceive and act on its environment. Moreover, an agent can communicate with the other agents of simulation but follows an individual goal. An agent has competences and can possibly reproduce. For Demazeau [6], a real or virtual entity can be described as agent when it is able to control whole or part of its operation (perception, reasoning, actions). For Tisseau [24], each agent can be assimilated to a three strokes engine:

1) *perception* : the agent perceives its immediate environment using specialized sensors,
2) *decision* : it decides what it must do taking into account its internal state, the values of its sensors and its intentions,
3) *action* : it acts by modifying its internal state and its immediate environment.

*2) **Multiagent systems:*** When a great number of these agents are joined together to work in a common environment, they form a multiagent system. Multiagent simulations allow to model complex situations and they are individual-centered. But like any modelling, the multiagent approach simplifies the studied phenomenon. On the other hand, this kind of simulation allows to mainly respect the complexity of phenomon by authorizing a diversity of components, a diversity of structures and a diversity of interactions. Such simulations allow the observation of structures and behaviors which emerge at the collective level from the individual and local interactions of the agents. For Demazeau [6], a MAS can be described by its four facets, the vowels AEIO, which are the initial ones of *Agent*, of *Environment* (the support of the agents' actions), of *Interaction* (the interactions between the agents have a role) and of *Organization* (which is produced by the layout of the relations between the entities). Tisseau [24] suggests to add to it the concept of *User* (vowel U) so that the operator takes an active part in the simulation. It will allow to carry out real experimentations of the model. It is the experimenter himself who gives sense to the simulation by wanting to test its hypotheses and by interacting with the agents of simulation and unsettling them to test the reactivity of the model. In the field of biology like elsewhere, the user must formed integral part of the numerical simulation, as much as he is in the usual biochemical experiments (called In Vivo or In Vitro). Tisseau suggests the term of *In Virtuo* experimentation, to qualify such simulations executed by computers, in a virtual world populated with autonomous entities.

B. ARéVi : multiagent platform

The platform ARéVi (Workshop of Virtual Reality) [1] is a multiagent platform for the virtual reality developed in the C.E.R.V. (European Center for Virtual Reality). It is a toolbox to create applications of virtual reality. It takes into account various techniques of graphic rendering, space sound and manages various peripherals to allow the interaction between the users and the virtual world. The core of this platform [11] allows the implementation of multiagent systems. This core offers the possiblity to implement autonomous agents which communicate by messages. So, each agent has properties, know-how, a letterbox and activities. The scheduling of the activities have the following characteristics: it is asynchronous, chaotic and without handing-over. For each cycle of simulation, the scheduler will carry out all the activities whose the delay of wait is reached and that only once. The execution of these activities is done in a sequential way, the ones after the others. The order in which they executed is random. Moreover, inside the same cycle, the last launched activities can exploit and reason on the information brought by the first launched activities (asynchronism).

Several biological models have already been simulated on this platform. The agents represent cells [5], biochemical reactions [13], [20] or more complex phenomena [7].

The continuation of the article will be dedicated to the realization of a software performing multiagent simulations which recreate the most accurately possible the behaviors of the stochastic π-calculus processes. This software will be based on the multiagent platform AReVi.

IV. Model

To answer the posted objective, i.e. the multiagent simulation of a stochastic π-calculus model, it is first of all necessary to achieve an exhaustive analysis of the model. Then, it is necessary to retranscribe as truthfully as possible the mechanisms of simulation of stochastic π-calculus in the agents of the MAS. The direct implementation of these methods will lead us to the realization of a software of simulation. Its first module will make the *analysis* and its second module will *generate* the agent oriented computer code that will be exploitable by the multiagent platform ARéVi.

A. Stochastic π-calculus model analysis

To be able to carry out a simulation in conformity with the model, it is necessary to save all the characteristics of the model: the first module of the software extracts, controls and saves all the information contained in the model. The input of this module is simply a textual file containing the model i.e. the list of descriptions of each process of the π-calculus model (see the examples given figures 2 and 3). This description uses syntax employed higher. This syntax is close to that defined by Milner (only some characters were substituted: ! for the emission; ? for the reception and new in instead of ν), contrary to the syntax used in the tools like BioSPi [23] and SPiM [16]. No prior declarations of the channels of communication and initial parameters of simulation are added in this file. The control of this text is certainly not exhaustive but already allows to check if:

- the words and the signs employed in the model are those of stochastic π-calculus.
- the layout of the words of the model respects the syntax of stochastic π-calculus models (for that, the software searches a layout which is defined by grammatical rules).
- if a called process exists.
- the number of arguments passed during a call corresponds to the number of waited arguments.
- the same channel is always used with the same rate.

These detected errors will be easily and clearly sent to the user who will then be able to make corrections at his model. After collecting the information on the stochastic π-calculus model, it is necessary to find the means of carrying out a simulation of this model by using the capacities and the resources of a MAS. Now, we will detail the methods which we worked out to be able to simulate a stochastic π-calculus model with a MAS. The two fundamental points of this work, on which a simulation of stochastic π-calculus model rests, are the scheduling of the actions of the processes during the time and the algorithm of Gillespie.

B. Process Agents

The π-calculus models are, as we saw, individual-centered models. They allow to express the behavior of each substance or entity which composes the system to be modelled. The Process agent is a more or less complex agent wich will be just like a stochastic π-calculus process. To be done, the Processus agents will respect the traditional groundwork of the computer agent:

- they are *pro-active* : they must fill a plan of actions in conformity with the π-calculus description. When this plan of actions is filled, they die.
- their phase of *perception* corresponds to know which actions they accomplished and which are potentially to make.
- their phase of *decision* corresponds to determine the actions which will be to launch (and that according to the results of perception).
- their phase of *action* is the realization of the selected actions.

The diagram of the actions of a process is governed by the priorities of the operators wich coordinate the actions. There are $+$, . and $|$, for respectively the choice, the sequentiality and the parallelism. By taking into account the priorities of these operators, we can work out a hierarchy and place the various actions at the leaves of a tree.

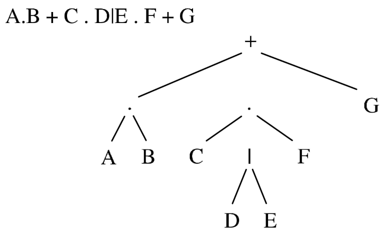

Fig. 1. *Example of tree in wich the state machine works.*

Action plan of Process agents: The Process agents must be just like π-calculus processes which they represent. They therefore have to perform a series of actions which can be communications, internal actions, calls to other agents. So, the agents of ARéVi will have to born, evolve and die as the π-calculus processes. It is therefore easy to understand that for each Call action, an instanciation of a new agent is associated. Also, when an agent has no more action to be performed, he must die.

The objective of this section is to introduce a generic method to determine actions that the Process agent must launch. This method is based on an state machine. The state machine is composed of the arborescent diagram of the actions of a π-calculus process (see figure 1). The state machine will reason on this tree. This tree has the following characteristics: each branch has a state (*finished*, *in progress*, or *not selected*) and each type of node has a particular behavior. These nodes make go up information of the states of the daughter branches towards the mother branches or conversely propagate an order resulting from a mother branch towards the daughter branches. Depending on whether they symbolize the sequentiality, the parallelism or the choice, these nodes do not transmit information on the branchs' state of the identical manner. The Process agent will exploit this state machine to live. The first type of behavior which makes go up information on the states since the branches associated with the actions (the actions represent the leaves of the tree) to the first branches of the tree will be used for the Process agent to report the actions which it made. Of course, that will be done during the *perception* phase of the Process agent. The second type of behavior which propagates an order come from a mother branch towards the daughter branches will be useful during the *decision* phase of the Process agent. At the end, it will only have for module *action* of the agent to launching the actions whose ascending branches are in a state *in progress*. To initialize this state machine that each Process agent has, we put the state of all branches at *not selected* and launch the order *in progress* on the higher branch of the first node.

C. Gillespie's algorithm

In the algorithm of Gillespie (detailed at section II-C.2) applied to stochastic π-calculus, the selection of the channel of communication on which will be achieve the exchange of information is made by taking a random number and by knowing all the channels likely to be chosen. The rigorous implementation of this algorithm implies the use of a data structure that contains all these informations. It is a centralized implementation. But with the intention of keeping the concepts of the agent paradigm, we cannot preserve this model of implementation. A distributed model for the Gillespie's algorithm is needed. In this case, there is not object or agent that supervises the selection. Nevertheless, this new model should not skew the initial algorithm. This distributed model will be integrated into the agents of the MAS which represents the channels (and named Channel agents). The decisional capacity, carried out by the draw of a random number, which was centralized, is transferred to the Channel agent level. So, it is not a supervisor who goes to elect the channel on which the communication will be done. The channels themselves are going to auto-designate. They auto-designate by knowing necessary parameters of course, namely their Actual Rate (value a_i in the initial Gillespie's algorithm) and the sum of Actual Rate.

Each one tries to auto-designate by choosing its own random number and uniformly distributed. In that way, it is possible that several Channel agents auto-designate or none. In that case, when there are not one and single Channel agent which auto-designates, we are going to repeat this algorithm without keeping anything of the previous failure. So, when there is success, the probality of such or such channel is in agreement with the probability defined by Gillespie: for the channel i the probability of being elected is

$$P = \frac{ActualRate_i}{\sum_{j=0}^{j=N} ActualRate_j}$$

This algorithm distributed, as we said, restarts if it fails. But, is there a risk that this algorhithm never find a solution? An example in [4] exploits a similar algorithm: the election by iteration of a leader by using a random variable. It is shown the election of a leader with a probability 1 will not exceed e iterations on average. e is the constant of Euler which is worth approximately 2.72. Also, we can notice it by an empirical way. The average value of the iterations in the election of the Channel agent is lower than 3.

Another fundamental point to build this distributed algorithm is the consideration of the way in which the Channel agents interact between them. They send messages of course and that in several steps:

- *Step 1* : first of all, the Channel agents are listening to requests of the Process agents. These requests are transmissions in point-to-point. They must record all the requests to be able to count them and to find the value of Actual Rate ($ActualRate = nbSenders \times nbReceivers \times rate$). They broadcast this value to the other Channel agents.

- *Step 2* : they make the sum of Actual Rate which were transmitted to them. It allows them to find the coefficient a_0 of Gillespie's algorithm (see the description of this algorithm in the section II-C.2).
- *Step 3* : each Channel agent draws a random number. If this number is lower than the ratio between its Actual Rate and the sum of Actual Rate, it is considered as auto-designated and informs all other Channel agents about it by the broadcasting of this information. At this time, if the Channel agents receive a single message (i.e. only one agent is auto-designated), they consider that the election of the agent finished. Otherwise, they start again the step 3.
- *Step 4* : the Channel agent indicated will be able to choose a sender Process agent and a receiver Process agent among those wich sent requests to it. This choice is transmitted to all Process agents. Like that, the Process agents chosen can perform their communication inter-process while the others are informed of the failure of their request.
- *Step 5* : the last step consists in calculating time associated with the communication inter-process according to the equation given by Gillespie. This calculation is done by the elected Channel agent which will broadcast it to all other Channel agents. The reception of this time will have the effect of restarting the activity of the Channel agents in some period of time: the time of Gillespie.

The life cycle of an agent Channel is in two steps: first, it treats the requests which it receives, in cooperation with other agents Channel. During this step, they are entirely synchronized. It is necessary to be able to make the election. On the other hand, the Channel agents which did not receive any requests are not concerned by the algorithm of Gillespie. The second step is a step of sleep. It is at this time that the time of simulation is incremented. At the end of the period of sleep (which lasts the time of Gillespie), the Channel agents take their activities back and treat the new messages.

By definition, the agents are autonomous entities which are not coupled the ones with the others. The addition of one or more agents in the simulation is completely possible and accomplishable. At any time, the user can unsettle the simulation in progress by removing or by adding agents. This characteristic amounts to put the user in the loop of the simulation of the models as Tisseau suggested it (section III-A.2). Thanks to this flexibility of handling, the user can test his model with complete freedom and easiness.

```
Enzyme         =   new (release , catalyse)
                   (bind ! (catalyse, release), 1) .
                   BoundEnzyme

BoundEnzyme    =   (release ! (), 1)     . Enzyme +
                   (catalyse   ! (), 30). Enzyme

Substrate      =   (bind ? (cat, rel), 1) . BoundSubstrate

BoundSubstrate =   (rel ? (), 1) . Substrate +
                   (cat ? (), 30). Product

Product        =   Product
```

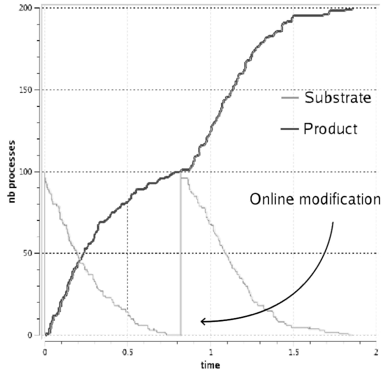

Fig. 2. *Simulations of the stochastic π-calculus model of the enzymatic reaction with disruption of the user.*

V. Application

The software resulting from the implementation of these models is named SPiC-MAS for *Stochastic Pi Calculus - MultiAgent Simulation* [2]. It allows to make multiagent simulations of stochastic π-calculus models. These models are the simple scripts of stochastic π-calculus recorded in textual files. This software rests on the analyzers Flex and Bison and on the multiagent platform ARéVi. The graphical user interface allows to choose a stochastic π-calculus model and to do the compiling of it. This step of compiling will translate the model into oriented agent code for ARéVi. Moreover, this interface allows to launch simulation by choosing the initial conditions (the number of π-calculus processes at the beginning of the simulation) and especially to interact with simulation (adjustment of the quantities of π-calculus processes) at any time. Let us see now some stochastic π-calculus models and possible results of their simulations obtained with SPiC-MAS.

A. Generic enzymatic reaction

During this reaction, a substrate will bind to the active site of an enzyme. This union will form the complex enzyme-substrate. At this time, the enzyme can activate the substrate which is transformed then into product or release the substrate just as it is. At the end of the reaction, the initial quantity of enzyme must be found and all the substrates are transformed into product. In the simulation carried out with *SPiC-MAS* (figure 2), once all substrates of the beginning are reacted, we add a

```
Mg        =      (ionize1   ! (), 10 ) . Mg_plus

Mg_plus   =      (ionize2   ! (), 100) . Mg_plus2 +
                 (deionize1 ? (), 50 ) . Mg

Mg_plus2 =       (deionize2 ? (), 5  ) . Mg_plus

Cl        =      (ionize1   ? (), 10 ) . Cl_minus +
                 (ionize2   ? (), 100) . Cl_minus

Cl_minus =       (deionize1 ! (), 50 ) . Cl       +
                 (deionize2 ! (), 5  ) . Cl
```

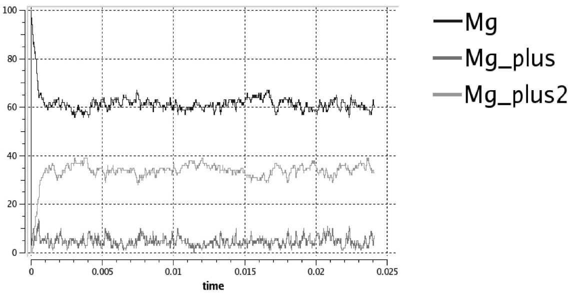

Fig. 3. *Simulation of stochastic π-calculus model of chemical reaction.*

certain quantity of substrate. That shows well the user can modify the quantity of agents (i.e. π-calculus processes) at any time and can observe the new behavior of the system.

B. Chemical reaction

This example was already used by the designers of BioSPi [19] and also by the designers of SPiM [3]. It represents the reaction between molecules of magnesium and chlorine. The system is usually described by the following equation: $Mg + 2\ Cl \leftrightarrow Mg^{++} + 2\ Cl^{-}$. Its stochastic π-calculus model used by *SPiC-MAS* is shown on figure 3.

C. Other examples

Other simulations were carried out with our software. We can quote the "Evolved Gene Network" model [9] and [17], as well as the "Gene Regulation by Positive Feedback" model [16] and [19].

VI. Discussions and conclusion

This proposal for simulations of stochastic π-calculus models by a multiagent system is viable and gives similar results to other simulators [23] and [16]. However, from the point of view of computer performances, it is not optimum. Indeed, the agents of the MAS have sometimes lifespan very short - their instanciations and their destruction mobilize some resources and their interactions by the communication of messages is more expensive than a simple call of a function for instance. But this software can be worthy of interest because it is ergonomic, its use is intuitive and it is interactive: the user has a precise return of the state of the simulation (thanks to an appropriate GUI) and the user endorses experimenter's role by unsettling simulations in progress.

References

[1] http://sourceforge.net/projects/arevi/.

[2] http://www.cerv.fr/∼bourhis/spicmas/.

[3] http://www.doc.ic.ac.uk/∼anp/spim/chemical.pdf.

[4] H. Attiya and J. Welch. *Distributed Computing : Fundamentals, Simulations and Advanced Topics*. McGrawHill, 1998.

[5] P. Ballet, J.-F. Abgrall, V. Rodin, and J. Tisseau. Simulation of thrombin generation during plasmatic coagulation and primary hemostasis. In *SMC'00, IEEE International Conference on System, Man & Cybernetics*, volume 1, pages 131–136, Nashville (USA), 8–11 October 2000.

[6] Y. Demazeau. From interactions to collective behaviour in agent-based systems. In *First European conference on cognitive science*, pages 117–132, Saint Malo, France, April 1995.

[7] G. Desmeulles, G. Querrec, P. Redou, S. Kerdélo, L. Misery, V. Rodin, and J. Tisseau. The virtual reality applied to biology understanding: The *in virtuo* experimentation. *Expert Systems with Applications, Elsevier*, 30:82–92, 2006.

[8] J. Ferber. *Multi-Agent System: An Introduction to Distributed Artificial Intelligence*. Addison Wesley, Reading, MA (USA), 1999.

[9] P. François and V. Hakim. Design of genetic networks with specified functions by evolution in silico. *Proceedings of the National Academy of Sciences of the United States of America*, 101:580–585, 2004.

[10] D.T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81(25):2340–2361, 1977.

[11] F. Harrouet, J. Tisseau, P. Reignier, and P. Chevaillier. oRis: un environnement de simulation interactive multi-agents. *RSTI–TSI*, 21(4):499–524, 2002.

[12] C. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.

[13] S. Kerdélo, J.F. Abgrall, M. Parenthoën, and J. Tisseau. Multi-agent systems: a useful tool for the modelization and simulation of the blood coagulation cascade. In *AAMAS 2002, Workshop Bioinformatics And Multi-Agent Systems (BIXMAS'02)*, pages 33–36, Bologna, Italy, July 2002.

[14] R. Milner. *Communication and Concurency*. Prentice Hall, 1989.

[15] R. Milner. A calculus of mobile processes, 1 & 2. *Information and Computation*, 100(1):1–77, 1992.

[16] A. Phillips and L. Cardelli. A correct abstract machine for the stochastic π-calculus. *Electronic Notes in Theoretical Computer Science*, 2004. http://www.doc.ic.ac.uk/∼anp/spim.

[17] A. Phillips and L. Cardelli. A graphical representation for the stochastic π-calculus. In *Bioconcur'05*, August 2005.

[18] C. Priami. Stochastic π-calculus. *The Computer Journal*, 38(7), 1995.

[19] C. Priami, A. Regev, E. Shapiro, and W. Silverman. Application of a stochastic name-passing calculus to representation and simulation of molecular processes. 80(1):25–31, October 2001.

[20] G. Querrec, V. Rodin, J.F. Abgrall, S. Kerdélo, and J. Tisseau. Uses of multiagents systems for simulation of mapk pathway. In *BIBE 2003, Third IEEE International Symposium on Bioinformatics and BioEngineering*, pages 421–425, Bethesda (USA), March 2003.

[21] A. Regev and E. Shapiro. Cellular abstractions: Cells as computation. *Nature*, 419:343, 2002.

[22] A. Regev and E. Shapiro. The π-calculus as an abstraction for biomolecular systems. *Modelling in Molecular Biology*, pages 219–266, 2004.

[23] A. Regev, W. Silverman, and E. Shapiro. Representation and simulation of biochemical process using the π-calculus process algebra. In *Proceedings of the Pacific Symposium of Biocomputing 2001*, volume 6, pages 459–470, 2001. http://www.wisdom.weizmann.ac.il/∼biospi/index_main.html.

[24] J. Tisseau. *Virtual Reality* –in virtuo *autonomy*–. Accredition to direct research, field computer science, Université de Rennes I, Rennes (France), December 2001. http://www.enib.fr/∼tisseau/doc/hdr/hdrJTuk.pdf.