A Tool for Using the SBML Format to Represent P Systems which Model Biological Reaction Networks

Isabel Nepomuceno, Juan Antonio Nepomuceno,
Francisco José Romero–Campero

Research Group on Natural Computing
Department of Computer Science and Artificial Intelligence
University of Sevilla
Avda. Reina Mercedes s/n, 41012 Sevilla, Spain
E-mail: isabel@lsi.us.es, janepochamorro@gmail.com, fran@us.es

Summary. In this paper we present a software tool to represent P systems modelling signalling networks of biochemical reactions using SBML (*Systems Biology Markup Language*), a machine-readable format for describing qualitative and quantitative models of biochemical networks. CLIPS (*C Language Integrated Production System*), a tool which provides a complete environment for the construction of rule and/or object based expert systems, has been used to simulated membrane system. Our tool acts as a translator from SBML to CLIPS; that is, besides providing an environment for writing SBML code it also parses this code and generates automatically the CLIPS code that simulates the membrane system represented in SBML.

1 Introduction

Membrane Computing is an emergent branch of Natural Computing, introduced by Păun in [9], which considers the different processes taking place into living cells as computing processes. It is a cross-disciplinary field, involving Formal Languages, Computer Science, Cellular Biology, etc; since its beginning it has received important attention from the scientific community and many different variants from different approaches and different goals have been introduced and studied. Actually, membrane systems have been considered as a fast *Emerging Research Front in Computer Science* by the Institute for Scientific Information, USA, and [8] was mentioned in [4] as a highly cited paper in October 2003.

Roughly speaking, a P system consists of a cell-like membrane structure which represents the hierarchical structure of the cell, multisets of objects that are placed in the compartments of the membrane structure to represent chemical substances and evolution rules which abstract the chemical reactions and processes that take

place inside the living cell. Usually the rules are applied in a synchronous non-deterministic maximally parallel manner. However, other semantics have been proposed, such as a bounded parallelism, a probabilistic application of the rules, an asynchronous evolution of the system, etc. Besides, variants that consider a population of cells (membrane systems) arranged in a graph representing a tissue like structure have been studied as well.

Most variants of these systems have been proved to be computationally complete, that is, equivalent in power to Turing machines and computationally efficient, that is, able to solve NP-complete problems in polynomial time by trading time for space; for details we refer to [10].

Summing up, P systems are a kind of rewriting systems that move a step further by representing the hierarchical structure of the living cell. Due to the fact that this new model of computation is inspired by the functioning of the cell it is natural to study and use these systems as a kind of specification language and framework for the modelling of different cellular processes and natural living systems. Several variants of membrane systems have already been used to model biological phenomena (see the volume [2]); recently a continuous variant has been introduced in [11] and a dynamical probabilistic approach has been studied in [1]. These recent developments of different variants used to model biological phenomena inside the framework of P systems make necessary information standards to facilitate the portability/comparison of the different models in order to study "when and how" one is better than the other.

In this paper we present SBML as a convenient machine-readable format for describing qualitative and quantitative models of biochemical networks developed within the framework of P systems. We also provide a software tool intended to be an environment for writing SBML code and a translator from SBML to executable code. Up to now we have only developed a translator from a SBLM description of a model to CLIPS code simulating the model using a P system.

This paper is organized as follows. In the next section we briefly show how P system have been used to model biological phenomena. In Section 3 we describe SBML, *Systems Biology Markup Language*, as a standard specification language and we discuss some of the characteristics which make it suitable for representing P systems. Our software tool is presented in Section 4; finally, conclusion and future work are given in the last section.

2 Modelling Biological Processes in the Framework of P Systems

Up to now, most of mathematical models of biological processes have been based on using differential equations. In this framework, the variation of the concentration of each chemical substance is modelled as a global process. This approach makes difficult the development of modular and scalable designs. That is, in order to extend a previous model, one has to start from scratch and rewrite a whole new

system of differential equations. Moreover, the modelling of the interactions at a molecular level does not scale to the cellular or tissue level. Using membrane systems we focus on the membrane structure and on the local interactions between different chemical substances, which are represented by rules that are applicable in different regions. This computational approach makes possible the extension of previous models by simply adding new rules, new objects and also new membranes without making major changes in the previous models. Furthermore, our approach is scalable, after having modelled the interactions at a molecular level we can study the evolution of the system as a whole to reach the cellular level and finally, in order to get insight into the dynamics at a tissue level we can consider a population of identical systems that have been previously designed and that can communicate through the environment or through specific links.

In this sense the most important characteristics of our approach are modularity and easy extensibility. Beside these features we mention easy understandability and programmability of the models developed within this framework.

Our work has the paper [11] as a started point. In this paper it was mentioned the necessity of information standards to facilitate the circulation and study of different models, like [1], [2], [7], [11], proposed for different biological phenomena. SBML was pointed as a good candidate to achieve this goal. Usual variants of P systems are discrete models of computation where in every step the rules are applied in a maximal way an integer number of times. A continuous P system evolves applying a maximal set of rules a positive number of times determined by a certain function, see [11]. In order to implement/simulate continuous P systems on real computers we need to develop approximation methods which consist basically in a discretization of the continuous evolution of the systems taking a small interval of time Δt. In this manner evolutions of continuous P systems are approximated by computations of usual P systems working in a bounded parallel manner. For details we refer to [11].

The model presented in [11] of modelling EGFR signalling cascade using continuous P systems was implemented using CLIPS. CLIPS is a an expert systems tool which provides a complete environment for the construction of the rule and/or object based systems.

3 SBML: Systems Biology Markup Language

Advances in biotechnology are leading to larger, more complex quantitative models describing systems of biochemical reactions. The complexity of these models is such that information standards are necessary if these models are to be shared, evaluated and developed cooperatively. Lately, membranes systems have been proved to be a suitable framework to develop local, modular and topological models of biological processes. These models have been designed using different variants of P systems and they have been implemented in different P systems simulators written in different programming languages, like CLIPS, JAVA, PROLOG, etc.

SBML, *Systems Biology Markup Language*, is a machine-readable format for describing qualitative and quantitative models of biological systems, see [14]. SBML provides a standard biochemical network model of representation and it promotes inter-operability between tools. SBML is based on XML, *eXtensible Markup Language*, which is a standard language for describing markups languages. In general, markups languages capture particular information in text for using it: for future processing, for storing it, for exchanging it with other systems, etc. SBML is still under development. Up to now two levels have been released. Level 1 and 2 capture fundamental features common to all biochemical network models; the second level in a richer manner than the first one. Level 3 is about to be available; it is developed in order to capture several aspects like hierarchical models, spatial features, kinetic constants, etc.

An SBML document has the following structure:

```
<?xml version="1.0" encoding="UTF-8"?>
 <sbml xmlns="http://www.sbml.org/sbml/level2"
                                        level ="2" version ="1">
   <model id = " ">
      <listOfFunctionDefinitions>
                 ...
      </listOfFunctionDefinitions>
      <listOfUnitDefinitions>
                 ...
      </listOfUnitDefinitions>
      <listOfCompartments>
                 ...
      </listOfCompartments>
      <listOfSpecies>
                 ...
      </listOfSpecies>
      <listOfParameters>
                 ...
      </listOfParameters>
      <listOfRules>
                 ...
      <listOfRules>
      <listOfReactions>
                 ...
      </listOfReactions>
      <listOfEvents>
                 ...
      </listOfEvents>
   </model>
</sbml>
```

The outermost portion of an SBML document consists of the definition of Sbml for SBML Level 2 Version 1. The XML namespace URI for SBML Level 2 is "http://www.sbml.org/sbml/level2" and the character encoding for SBML is UTF-8.

In the next line an identifier is associated with `model` and next the different components are written. All these components are optional and we will only use `listOfParameters`, `listOfCompartments`, `listOfSpecies` and `listOfReactions`.

Recall that a *continuous* P *system* is a construct, $\mathbf{\Pi} = (\Sigma, \mu, w_1, \ldots, w_n, \mathcal{R}, \mathcal{K})$, where:

1. $n \geq 1$ is the degree of the system (number of membranes);
2. $\Sigma = \{c_1, \ldots, c_m\}$ is the alphabet of *objects*;
3. μ is a *membrane structure* consisting of n membranes labelled with $1, \ldots, n$.
4. $w_1, \ldots, w_n$ are continuous multisets associated with each membrane of the membrane structure μ
5. $\mathcal{R}$ is a finite set of *rules* of the form:

$$u\,[\,v\,]_i \;\rightarrow\; u'[\,v'\,]_i,$$

   where $u, v, u', v' \in \Sigma^*$, and $1 \leq i \leq n$.
6. $\mathcal{K}$ is the *rate of application function* which associates with each rule and multiplicity of the objects in μ the rate of application of the rule:

$$\mathcal{K} : \mathcal{R} \times \mathcal{M}_{n\times m}(\mathbf{R}^+) \rightarrow \mathbf{R}^+,$$

   where $\mathcal{M}_{n\times m}(\mathbf{R}^+)$ is the set of matrixes of order $n \times m$ over $\mathbf{R}^+$.

Next we show how we can specify P systems using SBML.

In the component `listOfParameters` we collect all the unknown parameters in our model.

Using the component `listOfCompartments` we can specify the membrane structure representing the label of the membrane in the field `id` and the hierarchical relationship between the different membranes using the field `outside`.

```
<listOfCompartments>
   <compartment id= "  " outside= " " >
              ...
<\listOfCompartments>
```

The alphabet and initial multisets of the system are represented in the component `listOfSpecies`. For each object in the alphabet we associate a species with an identifier `id` and in the fields `initialAmount` and `compartment` we represent the initial multiplicity and the membrane where the object is placed in the initial configuration.

```
<listOfSpecies>
   <species id = " " compartment = " " initialAmount = " ">
                    ...
<\listOfSpecies>
```

At level 3 we can choose a specification which allows us to work with the kind of reactions in which we are interesting. We have been working with [3], a proposal of specification of level 3 which contains reactions with kinetic constants.

Rules that are used in [11] are of the following form:

$$u[v]_M \longrightarrow u'[v']_M, \quad KL,$$

with KL the kinetic constant. We need to express the reactants and the kinetic constant associated to the reactions. Following [3] the corresponding reaction of the SBML document which describes it would be:

```
<reaction id=R1 reversible="true">
    <listOfReactants>
        <speciesReference id="u-membrane_father_of_M"
                                                speciesType="u">
        <speciesReference id="v-M" speciesType="s">
    </listOfReactants>
    <listOfProducts>
        <speciesReference id="u'-membrane_father_of_M"
                                               speciesType="u'">
        <speciesReference id="v'-M" speciesType="v'">
    </listOfProducts>
    <kineticLaw>
        <math xmlns="http://www-w3.org/...">
            <apply>
                <times/>
                <cn> KL </cn>
                <ci> u </ci>
                <ci> v </ci>
            </apply>
    </kineticLaw>
</reaction>
```

The main goal of SBML is to provide a good description of biological processes, especially biochemical networks, so machines can read them.

4 A Tool for Working with P Systems Modelling Biological Processes

In this paper we present a modular tool with two different objectives. In the first place, we create a tool with a friendly interface and a P system as engine made in CLIPS. The engine simulates a concrete P system model in the way we explained previously and this engine is a changeable component, that is, the P system implemented in CLIPS is changeable with other P system implemented

in CLIPS or in other language. It has a restriction: if other language is used, the input format of this engine must be in the same format as the input format in CLIPS. In the second place, the tool have a parser module whose objective is to be able to add rules to the concrete model that we like to simulate (the next step is to allow to modify the complete model with its rules), that is to say, changing the engine. In this parser module there exists the possibility to export our model to a file which contains this model in the SBML format with the objective to use it in other tool that accepts the SBML format.

The tool is built as an architecture model of software development used in interactive systems. The application, following the MVC (Model-View-Controller) architecture, is composed of three different components. In the first one, *Model Component*, we have the engine of simulator of the EGFR signalling cascade using continuous membrane systems made in CLIPS, see [11], the translator of models (from CLIPS to SBML and from SBML to CLIPS), functional qualities and type abstract data. The second component, *View*, has the user guide interface responsible to showing the results. Finally, the third component, *Controller*, establishes the relation between the engine CLIPS and the Java class by means of a dll library. Figure 1 represents the hierarchical structure and the relation between the different components designed in this application.

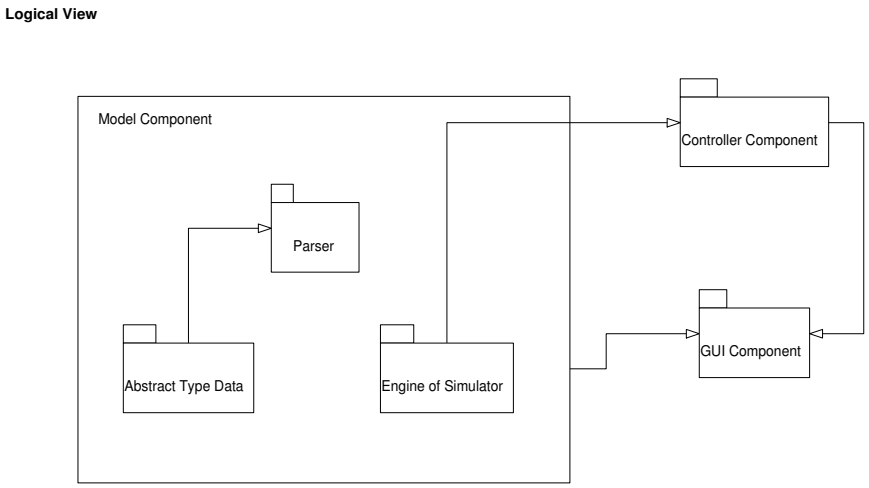

Fig. 1. Hierarchical structure and relations among the different components of the application. We underline the package (sets of classes) parser, which contains all functionalities to identify the SBML format and transforms it in an input file of the model. The engine simulator package contains several CLIPS files. The controller package contains libraries to communicate among CLIPS and Java.

The model presented to simulate the EGFR signalling cascade using continuous membrane system has been implementing by means of CLIPS. It is a productive development and expert system tool which provides a complete environment for the construction of rules and/or object based expert system. The engine of the simulator in CLIPS works independently of the rest of Java package of this tool, that is, the CLIPS package is changeable with other model implemented in CLIPS. In this way we can expand our model and use yet this tool, for example, to model the PI3K phosphorylation.

In the Model Component, see Figure 1, there is the parser package[1]. A parser is a program that takes a set of sentences as an input and finds its syntactic structure according to a given grammar; then, the parser transforms this syntactic structure in other kind of structure text. This parser is independent of the model implemented by CLIPS (in this case the EGFR signaling cascade), that is, we can use this parser package for any type of topological and modular model working in the engine of CLIPS simulator. We have implemented three kinds of parsers. The first one, *ParserClispToJava*, takes a list of reactions written in SBML syntax and identifies the markup language to translate it, and then it builds a list of reactions as a list of rules implemented in Java. This data type is converted in the syntax used in a input file of a list of reactions that understand the model implemented in CLIPS. The second parser, *ParserSbmlToJava*, is the opposite to the one above; it translates a list of reactions written in a syntax used in the CLIPS engine to a list of rules implemented in Java and this is translated to a SBML model. Finally, we have implemented a third parser, *ParserEasyRules*, to translate a simple rule written in the style of membrane computing [2], easier to write for the user, and parses this in SBML or CLIPS format. See Figure 2 for details.

The View Component includes all classes that contain the GUI (Guide User Interface). This is the part of the program which allows the user to interact with the application. The Swing Java Package is used in the construction of this GUI.

Finally the Controller Component contains the management-reception of events generated between the GUI and the simulator engine. In this moment this interaction is made only in the direction from the GUI to the engine, that is, we can transform the format of the input list of rules of the model implemented and run the engine CLIPS. The second direction, the presentation of the results (the evolution of a number of key proteins in the EGFR signalling cascade) is under development.

In figure 3 we present the interface of the application and we explain how to use it.

[1] A package is a group of classes whit a similar functionality.

[2] This easy syntax to write a reaction is:

$$u[v]_{memb} \rightarrow u'[v']_{memb} Kineticlaw$$

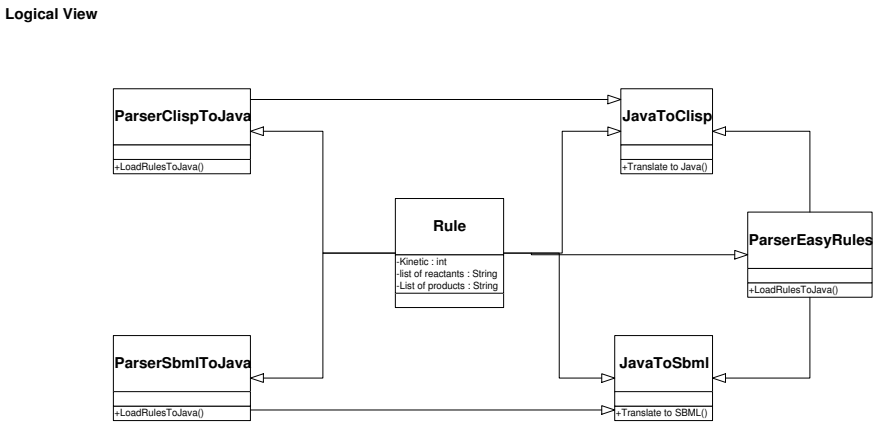

Fig. 2. Parser package: the set of classes. ParserSBMLtoJAVA identifies and recognizes the structure of the sentences according to a given grammar presented in the SBML Level 3 and build an input file in format to be read by the engine. ParserClispToJava makes the same in the inverse direction.

5 Conclusions and Future Work

In the following version of this tool there will be presented the evolution of a number of key proteins on the signalling network in a integrated way. We plan to use the Scientific Graphics Toolkit (written by Donald W. Denbo from the NOAA/PMEL/EPIC group) for creating interactive graphics applications.

In the present version of our software, we can interact in two directions: exporting the CLIPS model to SBML for reusing it in others tools, and importing new rules in the SBML format to add them to CLIPS model. In a following version we will permit not only to import rules, but also biological models or new P systems model completely.

References

1. D. Besozzi, G. Mauri, D. Pescini, C. Zandron: Analysis and simulation of dynamics in probabilistic P systems. Submitted, 2005.
2. G. Ciobanu, Gh. Păun, M.J. Pérez-Jiménez, eds.: *Applications of Membrane Computing*. Springer-Verlag, Berlin, 2005.
3. A. Finney: Systems Biology Markup Language (SBML) Level 3 Proposal: Multi-component Species Features, 2004. Available via the World Wide Web at: `http://www.sbml.org/workshops/ninth/supplementary/multi-component-species.pdf`
4. ISI web page: `http://esi-topics.com/erf/october2003.html`
5. JAVA web page: `http://java.sun.com`
6. I.A. Nepomuceno-Chamorro: A Java simulator for membrane computing. *J.UCS*, 10, 5 (2004), 620–629.

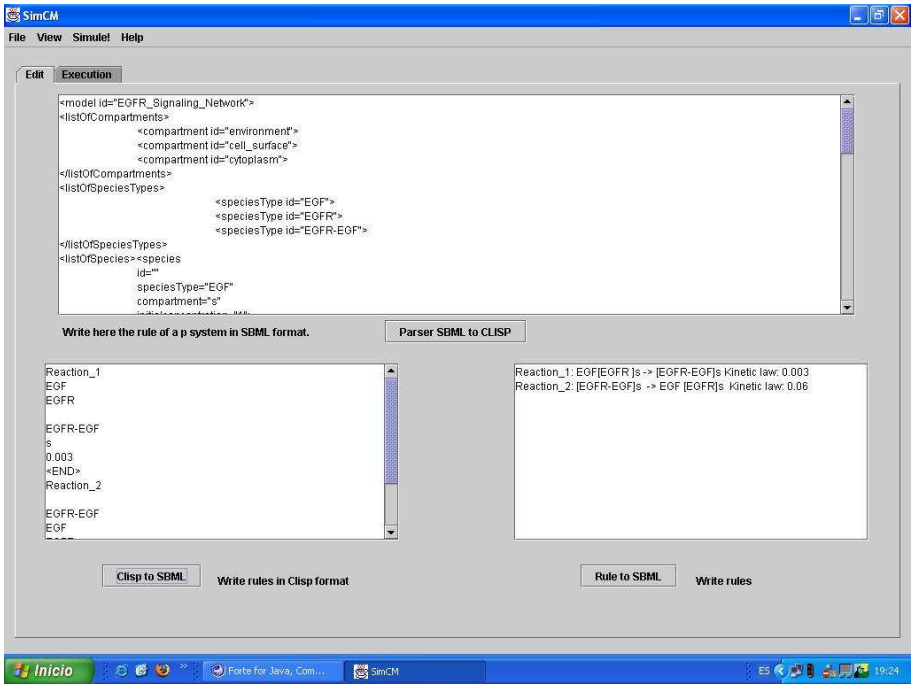

Fig. 3. The main screen of the program. One can see de barmenu that allows to handle the tool. The main screen is divided into two basic panels: "Edit", where one can use the SBML format to add rules to the model or export it to SBML, and "Execution", which shows an execution of the simulator (in a following version of the program, this will show us the output file as we have said before).

7. T.Y. Nishida: Simulations of photosynthesis by a K-subset transforming system with membranes. *Fundamenta Informaticae*, 49, 1 (2002), 101–113.
8. A. Păun, Gh. Păun: The power of communication: P systems with symport/antiport. *New Generation Computing*, 20, 13 (2002), 295–305.
9. Gh. Păun: Computing with membranes. *Journal of Computer Sciences and Systems Sciences*. 61, 1 (2000), 108–143.
10. Gh. Păun: *Membrane Computing. An Introduction*. Springer-Verlag, Berlin, 2002.
11. M.J. Pérez-Jiménez, F.J. Romero-Campero: Modelling EGFR signalling cascade using continuous membrane systems. *Third Workshop on Computational Methods in Systems Biology*, Edinburgh, 2005.
12. PSim, Java tool developed by Group for Models of Natural Computing (MNC), University of Verona, Italy
13. The P systems web page: `http://psystems.disco.unimib.it/`
14. The SBML web page: `http://sbml.org/index.psp`