Modelling the dynamic structure of biological state-based systems

I. Stamatopoulou\textsuperscript{a},*, P. Kefalas\textsuperscript{b}, M. Gheorghe\textsuperscript{c}

\textsuperscript{a} South-East European Research Centre, 17 Mitropoleos Street, Thessaloniki 54624, Greece
\textsuperscript{b} Department of Computer Science, CITY College, 13 Tsimiski Street, Thessaloniki 54624, Greece
\textsuperscript{c} Department of Computer Science, University of Sheffield, Regent Court, 211 Portobello Street, Sheffield S1 4DP, UK

Received 28 February 2005; received in revised form 8 July 2006; accepted 15 July 2006

Abstract

The paper discusses the modelling aspects of systems with dynamic processes and dynamic structure. A combination of models bringing together the benefits of two paradigms, Population P Systems and Communicating X-machines, is introduced. A simple case study is used in order to illustrate the potential of the combined use of the two methods.

© 2006 Elsevier Ireland Ltd. All rights reserved.

Keywords: Formal modelling; Dynamic reconfiguration; X-machines; Population P Systems; biologically-inspired systems

1. Introduction

Biological systems exhibit highly dynamic processes within a very dynamic environment. The individual components which they consist of demand modelling of evolving data structures and modelling of the control over their internal changing states. In addition, the systems overall imply modelling of their configuration, including the ability to exchange messages between individual components as well as the ability to re-structure their formation over time. Examples of such systems include colonies of ants or bees, flocks of birds, cell tissues, etc. (Dorigo et al., 1996; Gheorghe et al., 2001; Stamatopoulou et al., 2005a).

The complexity of such systems is due to substantial differences in attributes between their components, high computational power required for the processes within these components, non-trivial type or volume of data manipulated by these processes and finally considerable amount of communication in order for these systems to exhibit emergent behaviour. The use of a computational framework that is capable of modelling both the dynamic aspect (change) and the static aspect (data), will facilitate modelling and simulation of such complex systems.

The majority of models created for biological systems are based on an assumed, fixed system structure that is not realistic. The concept of growth, division and differentiation of individual components and the communication between them should be addressed in order to create a complete biological system which is based on rules that are linked to the underlying biological mechanisms allowing the dynamic evolution.

For example, consider the case of a tissue consisting of cells. Each cell has its own type, behaviours and evolution rules that allow it to grow, reproduce and die. The cells may be arranged in some two- or three-dimensional space. This layout implies the way they interact with other cells in the local neighbourhood through the exchange of biological constructs, e.g. chemicals. The structure of the tissue, that is the physical arrangement of the cells, changes over time, thus imposing a change in the cells’ interactions.
In the last years attempts have been made to devise computational models in the form of generative devices (Păun, 2000, 2002; Banatre and Le Metayer, 1990; Adleman, 1994; Holcombe, 2001) or hybrid machines (Duan et al., 2000; Holcombe et al., 2003). All these attempts were made to capture the diverse requirements of modelling biological or biology-inspired systems.

In this paper we propose the combination of two formal methods, X-machines and Population P Systems, in order to take advantage of the appealing characteristics of them both and facilitate the modelling of dynamically evolving biological systems. Section 2 of this paper describes a simple example that will be used as a paradigm model throughout the paper. Section 3 presents the theory of X-machines and Communicating X-machines, which are able to statically define a system consisting of state-based components that are able to communicate. Section 4 defines the operators that are suitable for changing the configuration of a Communicating X-machine system. Population P Systems are presented in Section 5. Finally, our proposal is discussed in Section 6, which presents how Communicating X-machines may be used at the component level with a Population P System model working at a higher level in order to deal with the dynamic evolution of a system.

2. Modelling a tissue of cells

Modelling tissues of cells points out a very characteristic example of a dynamic system. Tissues consist of individual cells whose internal structure and configuration change over time as they evolve; new cells are born, others die while their physical movement in space imposes changes in their communication with other cells.

For reasons of exposition the model presented here is rather abstract and follows various underlying assumptions that entail an overall simplicity (e.g. discrimination of possible movements and directions on two-dimensional space, placement of one cell per coordinate at any one time, etc.). This is because we are not fully dedicated to capturing the realistic view of the processes taking place inside a cell but rather focus on main characteristics in their behaviour and explore how these can be captured by a model.

Consider a simple cell that is born by division, matures and eventually dies. Its death can only be a consequence of time so it is assumed that cells have a predefined life span and that there are no deadly interactions that may end their lives prematurely. Similarly, a specific amount of time has to elapse from a cell’s birth before it can become mature. A mature cell is able to reproduce unless certain circumstances make it infertile (e.g. receiving unsuitable biological constructs). The cells can move freely in four possible directions on two-dimensional space. Neighbouring cells communicate by exchanging, under specific conditions, biological elements. Naturally, the cell may exhibit other more complex behaviours as well (Kefalas, 2002). Additionally, though time plays an important role in the way a cell evolves, we choose to deal with it implicitly in our model (Kefalas et al., 2005).

In summary the cells’ behaviours that we take into consideration are the following: (a) a cell can move around freely in empty space, (b) a cell will mature when the time comes, (c) a mature cell can reproduce by division in the presence of suitable biological constructs within it, (d) a mature cell may become infertile upon receiving unsuitable biological constructs from neighbouring cells or the environment, (e) mature and infertile cells expel chemicals to the environment or send them to neighbouring cells, and (f) a cell dies when the time comes.

The free movement of the cells within the space constantly changes the physical configuration of the entire tissue and therefore affects their communication. Fig. 1 shows such an example of the evolving structure of the system.

Modelling the system requires specifying individual cells (i.e. data, states, behaviours, etc.) as well as capturing its dynamic configuration. For the modelling of a cell as an individual entity we propose the use of the X-machine formal method (Eilenberg, 1974; Holcombe, 1988) which can sufficiently capture both the data of the cell as well as the control that portrays the changes of its...
internal state. Additionally, the communication between the cells can be modelled with the use of Communicating X-machines.

X-machines can be thought to apply in similar cases where StateCharts (Harel, 1987) and other similar notations do. In principle, X-machines are considered a generalisation of models written in similar formalisms since concepts devised and findings proven for X-machines form a solid theoretical framework, which can be adapted to other, more tool-oriented methods. In the following section the theory of X-machines is briefly put forward and part of the model of the cell is indicatively presented.

3. Communicating X-machines

An X-machine (Eilenberg, 1974) resembles a finite state machine with the difference that there is a memory attached to it and that functions do not operate on input symbols but on input and memory values and produce output, possibly modifying the memory value. The X-machines formal method (Holcombe, 1988) forms the basis for a specification/modelling language that is rather intuitive while at the same time formal descriptions of data types and functions can be written in any known mathematical notation. The definition of the particular X-machine that is used specifically for the purposes of modelling follows.

**Definition 1.** A stream X-machine (Holcombe and Ipate, 1998) is defined as an eight-tuple:

\[ X = (\Sigma, \Gamma, Q, M, \Phi, F, q_0, m_0) \]

where \( \Sigma \) and \( \Gamma \) are the input and output alphabets, respectively; \( Q \) the finite set of states; \( M \) the (possibly) infinite set called memory; \( \Phi \), the type of the machine \( X \), is a set of partial functions \( \varphi \) that map an input and a memory state to an output and a possibly different memory state, \( \varphi : \Sigma \times M \to \Gamma \times M \); \( F \) the next state partial function, \( F : Q \times \Phi \to Q \), which given a state and a function from the type \( \Phi \) determines the next state. \( F \) is often described as a state transition diagram. Finally \( q_0 \) and \( m_0 \) are the initial state and initial memory, respectively.

Hereafter, the term X-machine refers to a stream X-machine. Fig. 2 depicts the state transition diagram defined by \( F \) for the cell model.

The definition of the cell model is:

\[ Q = \{\text{newborn, mature, infertile, dead}\}, \]
\[ \Sigma = \{\text{matureNow, reproduceNow, dieNow, sendConstruct}\} \cup \{(X, Y) | X, Y \in \mathbb{N}\} \cup \text{BioConstructs} \]

where BioConstructs is the set of all possible biological elements that may be present in the system and exchanged among the cells and \((X, Y)\) represent the coordinates of positions in the space. The inputs in the first set mark the passing of the required time for the respective actions.

\( \Gamma \) is a set of messages output by the functions:

\[ \Gamma = \{\text{cellMatures, cellReproduces, cellDies, cellBecomesInfertile, cellMoves, cellSentConstruct, cellReceivedConstruct}\} \]

\[ M = ((X, Y), C, U) \quad \text{where} \ C, U \subseteq \text{BioConstructs} \]

\( X, Y \) are natural numbers denoting the coordinates of the position of the cell on the plane, \( C \) the multi-set of the biological constructs contained in the cell and \( U \) is the
set of biological constructs with implications that lead to the cell’s infertility.

The initial memory could be, for example \( m_0 = (16, 29), \{a, a, b, c\}, \{x, y, z\} \) where \( a, b, c, x, y, z \in \text{BioConstructs} \) and the initial state \( q_0 = \text{newborn} \).

The set of functions, \( \Phi \), contains:

\[
grows(\text{matureNow}, ((X, Y), (C, U))) =
\begin{cases}
\text{cellMatures}, ((X, Y), (C, U))
\end{cases}
\]

\[
\text{receiveConstruct}(k, ((X, Y), (C, U))) =
\begin{cases}
\text{cellReceivedConstruct}, ((X, Y), (C \cup \{k\}, U))
\end{cases}
\]

\[
\text{becomeInfertile}(k, ((X, Y), (C, U))) =
\begin{cases}
\text{cellBecomesInfertile}, ((X, Y), (C \cup \{k\}, U))
\end{cases}
\]

\[
\text{move}((\text{neighbours}((NX, NY)), (X, Y), (C, U))) =
\begin{cases}
\text{cellMoves}, ((NX, NY), (C, U))
\end{cases}
\]

For the modelling of tissues, where more than one cell needs to be specified, Communicating X-machines possessing extra features, such as hierarchical decomposition and communication, are used. A Communicating X-machine model consists of several X-machines that are able to exchange messages. This involves the modelling of the participating components as separate X-machines and the definition of the rules specifying their communication.

**Definition 2.** A Communicating X-machine system \( Z \) is a tuple:

\[
Z = ((C_i^k),_{i=1,...,n}, R)
\]

where \( C_i^k \) is the \( i \)-th Communicating X-machine component, and \( R \) is a relation defining the communication among the components, \( R \subseteq C^i \times C^k \) and \( C^i = \{C_1, \ldots, C_n\} \). A tuple \( (C_i, C_k^k) \in R \) denotes that the X-machine component \( C_i^k \) can output a message to a corresponding input stream of X-machine component \( C_k^k \) for any \( i, k \in \{1, \ldots, n\}, i \neq k \).

A Communicating X-machine component (CXM for short) can be derived by incorporating into an X-machine information about how it is to communicate with other X-machines that participate in the system. Exchange of messages among the components is achieved by redirecting one component’s function output to be received as input by a function of another machine. In order to define the communication interface of an X-machine two things have to be stated: (a) which of its functions receive their inputs from which machines, and (b) which of its functions send their outputs to other machines.

Information about the first can be contained in a set \( I\Phi_i \) that consists of tuples of function names and the names of the components they receive their input from:

\[
I\Phi_i \subseteq \Phi_i \times C^k \text{ where } C^i \text{ is the set of all } C_i^i.
\]

A similar construct \( O\Phi_i \) may contain the information regarding which functions of a machine send messages to other machines. However, it has to be ensured that the message that a CXM \( C_i^k \) sends to another X-machine \( C_k^k \) belongs to the input alphabet \( \Sigma_k \) of the latter. Therefore, before a message can be sent, the values that it will contain need to be transformed into a format that can be readable by the receiver. Due to this restriction \( O\Phi_i \) needs to contain three-tuples holding not only functions and the components they send their messages to but also the kind of transformation \( T \) that needs to take place for the message to be acceptable by the receiver:

\[
O\Phi_i = \{(\phi_i, C_i^k, T_{\phi_i}C_i^k) | \phi_i \in \Phi_i \land C_i^k \subseteq C^k \land T_{\phi_i}C_i^k : \Sigma_i \times I\Gamma_i \times M_i \rightarrow \Sigma_k\}
\]

Graphically on the state transition diagram we denote the acceptance of input from another component by a solid circle along with the name \( C_i^k \) of the CXM that sends it. Similarly, a solid diamond with the name \( C_k^k \) denotes that output is sent to the \( C_k^k \) CXM. An abstract example of the communication between two CXMs is depicted in Fig. 3. It has to be noted that though a function \( \phi \) may only read from one component at a time, it is possible that it sends its output to more than one components.

**Definition 3.** A Communicating X-machine component \( C_i^i \) is defined as a 10-tuple:

\[
C_i^i = (\Sigma_i, I\Gamma_i, Q_i, M_i, \Phi_i, F_i, q_{0i}, m_{0i}, I\Phi_i, O\Phi_i)
\]

![Fig. 3. Abstract example of the communication between two Communicating X-machine components.](image)
where \( \Sigma_i, \Gamma_i, Q_i, M_i, \Phi_i, F_i, q_{0i}, m_{0i} \) are the same as in the definition of an usual X-machine, and \( I\Phi_i \) and \( O\Phi_i \) comprise the communication interface of the machine.

4. Reconfiguring the structure

The above allows the definition of systems of a static configuration. However, most biological systems are highly dynamic and this requires that their structure and the communication among its components are constantly changing. To this end the following operators have been fully defined in (Kefalas et al., 2005).

Definition 4. The attachment operator \( \text{ATT} \) is defined such that an existing CXM is able to join a set of other existing components and as a result establishes communication with them (Fig. 1a).

Definition 5. The detachment operator \( \text{DET} \) is defined such that an existing CXM can be detached from a set of other existing components and as a result all communication channels between them are removed (Fig. 1b).

Definition 6. The generation operator \( \text{GEN} \) is defined such that a new CXM is created and introduced into the system. If communication is required with other components, it is established (Fig. 1c).

Definition 7. The destruction operator \( \text{DES} \) is defined such that an existing CXM is able to join a set of other existing components and as a result all communication channels between them are removed (Fig. 1d).

The Communicating X-machine system can change its configuration by the operation of a meta-level system that will be responsible for the application of the operators above. For this reason, this meta-level system should possess the following information at any time:

- The communicating system tuple:
  \[ Z = ((C_1^x, \ldots, C_i^x, \ldots, C_n^x), R); \]

- The current system state \( \text{SZ} \) of \( Z \). \( \text{SZ} \) is defined as a set of tuples \( \text{SZ} = \{(q_c, M_c, \varphi_c); i, \ldots \} \) with each tuple corresponding to the computation state of a component \( C_i^x \): the current state \( q_c \) in which \( C_i^x \) is at, the current memory \( M_c \) of \( C_i^x \) and the last function \( \varphi_c \) that has been applied;

- Definitions of the types of components that may exist in the system. These act as genetic codes (GC) for the components of the system whereby GC is a set of tuples of the form, \( \langle \Sigma, \Gamma, Q, M, \Phi, F, q_0, m_0, \Phi_R, \Phi_W \rangle \) where the first eight elements are as in the definition of the X-machine and the last two are the sets of functions that may be involved in communication with other components (i.e. \( \Phi_R \) functions may read from communicating streams and \( \Phi_W \) functions may write to communicating streams).

We have seen so far that modelling with X-machines can be a straightforward process that provides us with a number of advantages but as the definition of such a meta-level system, which will take over the reconfiguration of dynamic systems, can prove to be a rather cumbersome task, alternative choices toward this direction have been looked into. One of them, Population P Systems, naturally possess the trait of restructuring the formation of the participating components on their own thus offering what Communicating X-machine systems lack. Their modelling potential has been looked into in Stamatopoulou et al. (2005a) in an effort to specify a flocking behaviour. Although defining the rules underlying the way the system evolves has been uncomplicated, Population P Systems were not as straightforward in specifying the internal structure and the internal states of the individuals.

Communicating X-machines and Population P Systems have complementary characteristics in the sense that the former are more suitable when it comes to modelling systems of a static structure whereas the latter are more adequate regarding the reconfiguration of an evolving collection of units. As a result, the idea is that we take advantage of the appealing characteristics of them both in a combination by which communicating components are defined as a Communicating X-machine system whose structure will be controlled by a Population P System at a higher level.

The theory related to Population P Systems is presented in the next section while the idea how the two methods may be used in conjunction follows.

5. Population P Systems with active cells

A natural generalisation of the P system model (Păun, 2000) can be obtained by considering P systems whose structure is defined as an arbitrary graph. Each node in the graph represents a membrane, which gets assigned a multi-set of objects and a set of rules for modifying these objects and communicating them alongside the edges of the graph (Păun, 2002). These networks of communicating membranes are also known as Population P Systems (PPS) (Fig. 4) because, from a biological point view, they can be interpreted as an abstract model of bio-entities aggregated together in more complex bio-units. In this respect the model also addresses the case of vari-
ous colonies of more complex organisms like ants, bees, etc.

These populations of individuals are usually far from being stable; mechanisms enabling new individuals to be introduced, for updating the links between them or for removing some individuals play a fundamental role in the evolution of a biological system as a population of interacting components.

The model of PPS is augmented with an operation of cell division as a mechanism to introduce new cells in the system, and with an operation of cell death as a mechanism to remove cells from the system. In addition to this, an operation of cell differentiation is considered that allows the type of the cells to be changed, thus varying in this way the sets of rules that can be used inside the cells.

**Definition 8.** A Population P System with active cells (Bernandini and Gheorghe, 2004) is a construct:

\[
P = (V, K, \gamma, \alpha, w_E, C_1^p, C_2^p, \ldots, C_n^p, R)
\]

where \( V \) is a finite alphabet of symbols called objects; \( K \) a finite alphabet of symbols, which define different types of cells; \( \gamma = \{1, 2, \ldots, n\}, A \), with \( A \subseteq \{i, j\} \mid 1 \leq i \neq j \leq n \), is a finite undirected graph; \( \alpha \) is a finite set of bond-making rules of the form \((t, x_1; x_2, p)\), with \( x_1, x_2 \in V^* \), and \( t, p \in K \); \( w_E \in V^* \) a finite multi-set of objects initially assigned to the environment; \( C_i^p = (w_i, t_i) \), for each \( 1 \leq i \leq n \), is a cell with \( w_1 \in V^* \) a finite multi-set of objects, and \( t_i \in K \) the type of cell \( i \); \( R \) is a finite set of rules dealing with communication, object transformation, cell differentiation, cell division and cell death.

Communication rules are of the form \((a; b, \text{in})_t, (a; b, \text{enter})_t, (b, \text{exit})_t\), for \( a \in V \cup \{\lambda\}, b \in V, t \in K \) and allow the exchange of objects between neighbouring cells, or a cell and the environment, according to the cell type and the existing bonds among the cells. The first rule means that in the presence of an object \( a \) inside a cell of type \( t \) an object \( b \) can be obtained by a neighbouring cell non-deterministically chosen. The second rule is similar to the first with the exception that object \( b \) is not obtained by a neighbouring cell but by the environment. Lastly, the third rule denotes that if an object \( b \) is present it can be expelled out into the environment.

Transformation rules are of the form \((a \rightarrow b)_t\), for \( a \in V, b \in V^+, t \in K \) meaning that an object \( a \) is replaced by an object \( b \) within a cell of type \( t \).

**Cell differentiation rules** are of the form \((a)_t \rightarrow (b)_p\), with \( a, b \in V, t, p \in K \) meaning that consumption of an object \( a \) inside a cell of type \( t \) changes the cell, transforming it into a cell of type \( p \). All existing objects remain the same besides \( a \) which is replaced by \( b \).

**Cell division rules** are of the form \((a)_t \rightarrow (b)_t(c)_t\), with \( a, b, c \in V, t \in K \). A cell of type \( t \) containing an object \( a \) is divided into two cells of the same type. One of the new cells has \( a \) replaced by \( b \) while the other by \( c \). All other objects of the originating cell appear unaltered in both new cells.

**Cell death rules** are of the form \((a)_t \rightarrow \dagger\), with \( a \in V, t \in K \) meaning that an object \( a \) inside a cell of type \( t \) causes the removal of the cell from the system.

At each computation cycle, all rules regarding the transformation and communication of objects that may be applied in a cell are applied. Additionally, one out of the applicable cell differentiation, division or death rules, non-deterministically chosen, is also applied in each cell. When computation in all cells has finished, the graph is decomposed and restructured according to the specified bond-making rules that define the conditions under which two cells are able to communicate.

6. Combining the two methods

The rationale behind the combination of the two methods is that:

- a Communicating X-machine models the internal data and change of state of a cell.
- a Population P System models the dynamic configuration of a tissue by applying the reconfiguration operators on the Communicating X-machine system.

In order to achieve that, the objects of the PPS need to capture information regarding the overall state of the Communicating X-machine system. Both systems have the same physical structure at any time of the computation, i.e. their graph representation remains the same. This implies that the PPS cells correspond in a one-to-one manner to the X-machine components. Fig. 5 depicts...
the notion of a PPS working on top but also in parallel
and in complete correspondence with a Communicating
X-machine system.

**Objects**: the alphabet of objects will contain all
possible computation states of the Communicating X-
machines. Each cell of the PPS will contain in its multi-
set of objects the computation state \((q_c, M_c, \phi_c)\) of the
CXM that it corresponds to. The alphabet should also
contain genetic codes, in order to be able to generate
new CXMs upon application of a division rule.

**Types of cells**: the types of cells in the Population P
System may be the same as the types of the X-machines
from which all system components are derived.

**Bond making rules**: the bond-making rules define the
circumstances under which two types of CXMs should
engage in communication. The rules are responsible for
two things: (a) for updating the graph of the PPS, and
(b) for applying the attachment operator \(\text{ATT} \) on the
Communicating X-machine system.

For example, in the case of the tissue, two cells may
start to communicate by exchanging biological elements
when they are physically next to each other. As the position
of the cells is the first element in their memory, the
corresponding bond-making rule would look like:

\[
(\text{cell}, (q_i, ((xpos_i, ypos_i), \ldots, \phi_i)); (q_j, ((xpos_j, ypos_j), \ldots, \phi_j)), \text{cell})
\]

if neighbours \(((xpos_i, ypos_i), (xpos_j, ypos_j))\)

and the attachment operator will link components \(C_i^x\)
and \(C_j^x\).

In correspondence to the bond making rules, there
should also be bond releasing rules that will remove
bonds from the PPS and apply the detachment opera-
tor \(\text{DET} \) on the Communicating X-machine system.

At this point it should be noted that in contrast to the
original definition of the PPS, according to which there
are only bond-making rules and at the end of each cycle
of the computation all bonds are removed and the graph
is created from scratch, our suggestion is that at the end
of each cycle the graph is only updated according to bond
making and bond releasing rules.

**Transformation rules**: object transformation rules are
responsible for updating a cell of the PPS with the corre-
spending CXM computation state. The transformation,
instead of being a simple rewrite rule, may be defined by
the computation that takes place in the CXM.

**Cell division rules**: according to the X-machine com-
putation state, a new cell appears in the PPS containing
the objects of the initial cell (i.e. all the genetic codes of
the machines) and an object representing the initial
computation state as it is defined by the genetic code.
A cell division rule is also responsible for the applica-
tion of the generation operator \(\text{GEN} \) on the underlying
Communicating X-machine system.

**Cell death rules**: finally, cell death rules, depending
on the computation state of the corresponding machine,
will make cells disappear from the PPS as well as de-
struct the corresponding CXM after the application of
the \(\text{DES} \) operator. For our tissue example, the cell death
rule would look like \(((\text{dead}. \ldots) \rightarrow \uparrow)_{\text{cell}} \) and the application of the \(\text{DES} \) operator will remove the component \(C_i^x\) from the system.

7. Discussion

Both X-machines and Population P Systems have
sound theoretical foundations and act as formal specifi-
cation languages. On the practical aspect, the X-machine
Description Language (XMDL) (Kapeti and Kefalas,
2000) has been defined facilitating the formal descrip-
tion of X-machine models and serving as a basis for
the development of tools, such as the X-system (Kefalas
et al., 2003), that allow the syntactical check and au-
tomatic animation of the X-machine models. Addition-
ally, X-machines have further techniques supporting the
modelling activity. \(\text{AnnCTL(Eleftherakis, 2003)} \) is an ex-
tension of CTL which facilitates formal verification of
desired system properties. This is accomplished with the
use of two additional operators that deal with the mem-
ory structure of the machines. Also, the generation of
a complete test set for X-machine models is feasible
through the use of a method, based on Chow’s \(W\)-method
(Holcombe and Ipate, 1998).

A similar notation to XMDL, PPSDL, has been de-
vised for Population P Systems while work has also been
done on a tool, PPS-system, that allows the animation
of models described in PPSDL. In order to investigate
the practical benefits of such a modelling approach, the
tumor growth example, borrowed from the NetLogo li-
brary (Wilensky, 1999), has been modelled in PPSDL
indicating there is promising ground for further research
(Stamatopoulou et al., 2005b). Finally, work has been
initiated on the transformation of XMDL to the NetL-
ogo language (Peneva and Kefalas, 2005) such that visual representation of the simulation of models described in XMDL becomes more straightforward.

Overall, this might be considered as a first attempt towards a kind of hybridisation of the two paradigms and there are various issues arising from this approach which require further investigation. First of all, a complete modelling attempt has to be performed in order to investigate the practical value of the approach. A formal definition of notation and precise semantics of the rule set should also be provided. Thirdly, under the new definition, it may be possible to transfer the communication between components at the upper level, that of the Population P System, instead of being explicit in the lower level communicating system. This may require some further thoughts because it may have implications in modelling, since the communication at the upper-level may entail some non-determinism which is not currently implied by the Communicating X-machine system.

Finally, it is worth noticing that it may be in principle possible to have any kind of model at the second layer, such as a collection of P systems, instead of a Communicating X-machine system. This will require the definition of the corresponding operators for the communicating model at the lower level but it will lead towards the use of any combinations of modelling methods.

8. Conclusions

In this paper we introduce the notion of combining two different computational paradigms: Population P Systems and Communicating X-machines to better facilitate the modelling of systems of a dynamic structure. A simple example has been used to illustrate the potential benefits of this new approach.

Future work will be directed towards the formal definition of this proposed hybrid framework. Furthermore, high in the list of priorities is the attempt towards creating a tool that will facilitate the animation of such hybrid models, most likely based on the existing tools X-system and PPS-system. This in combination with a complete modelling attempt of a case study would greatly contribute towards establishing the validity of the approach.

References