ISBN: 978-972-8924-60-7 © 2008 IADIS

CELLULAR PETRI NETS

J.M. Maestre, E.F. Camacho
Departamento de Ingeniería de Sistemas y Automática
Universidad de Sevilla
Camino de los Descubrimientos s/n, 41092, Sevilla, Spain

ABSTRACT

This paper proposes a software framework to model hybrid systems based on an extension of the Petri Net paradigm. Biology inspires changes proposed to the original paradigm. The main difference between this approach and others that can be found in literature is that token attributes may have their own dynamics, just like if they were living cells inside a more complex system. Object Oriented Programming is, at first, applied at this point so that these attributes are packaged in the token, so that most of the token complexity is hidden from the Petri net point of view. This grey box approach adds other important features that enhance Petri net modeling possibilities, like the continuous flow of discrete tokens. Cellular processes or sensor networks are some examples of systems that can take advantage of the new framework.

KEYWORDS

Petri Nets, Biology, Cellular token.

1. INTRODUCTION

Hybrid systems modeling have become a very important research topic in control engineering and other disciplines [1]. In this paper the most common definition of hybrid system will be used; that is, a system is hybrid if and only if its dynamics cannot be represented by means of purely continuous or discrete models and a mix of both is needed for a correct representation. This definition is not very rigorous, but it leaves enough freedom to apply hybrid modeling whenever it is needed.

During the last years a lot of mathematical formalisms have been defined in order to model hybrid dynamics in a compact way. The perfect modeling language doesn't exist but there are some valuable characteristics that we should look for in a language [2]. A good modeling tool should be:

- Descriptive: it should capture both continuous and discrete dynamics, the interactions between them, allow non deterministic models, etc.
- Scalable: bigger models can be built by composition of smaller ones.
- Modular: the yield of a composed system can be expressed as a result of the yield of its individual components. This way designs can be easily improved.

Although there have been many advances in the modeling field, some issues are not properly covered by the actual modeling tools. This is the case, for example, of distributed systems whose components have inner complex dynamics. The approaches to this problem lead to complex models or to ad hoc solutions. In [4] modified Petri Nets are proposed to relieve this lack of modeling options but some of the enhancements proposed in this paper are missed. The purpose of this paper is to set up a basis for a common framework that could ease the development of models and simulation for hybrid systems.

1.1 Paradigm Inspiration

Several extensions to the original Petri Nets language have been proposed to this date (e.g.:[4], [8] ,[9], [10]). Some of these have tried to expand the original model so that it could cover the hybrid phenomena. In [4] the application of object-oriented paradigm to HDN (Hybrid Dynamical Nets, a variation of Hybrid Petri Nets) is suggested to encapsulate subnets within object frames so that scalable models can be easily done. In [8] objects are embedded inside Petri nets. In this case, when a transition fires, token methods are executed to change their attributes. In [9] and [10] the famous Coloured Petri Nets are shown and some of their object-oriented properties studied.

The new Petri Net paradigm uses biology as inspiration source. Basic concepts such as places or transitions will be redefined and also compared to very popular biological terms. The comparison will be made in a very intuitive way. We will consider a token as a living cell.

1.2 Main Contributions

The key contribution of this paper is the definition of a framework that allows tokens to have a dynamic behavior, i.e., token whose attributes may change independently with time. This allows to model objects which have attributes such as time valid windows or net-independent inner dynamics. This allows that a great variety of modelling problems can be studied under the Petri Net perspective. It also considers the continuous flow of discrete tokens through the net, so that a token can be in two places at the same time.

1.3 Outline of the Paper

The starting language for this article is Petri nets. In first place the main concepts of these nets will be revisited ([5],[6]) The extensions to the original model will be defined in order to handle hybrid models. Finally, a example will be shown.

2. CELLULAR PETRI NETS MODEL

2.1 Petri Nets Revisited

Petri nets have two types of nodes: places and transitions. Places are represented by circles and transitions by bars. These two kinds of nodes are connected by arcs (arcs are directed and connect either a place to a transition or a transition to a place). Besides, the number of places and transitions must be finite and non zero. A formal definition of this model is given by:

A Petri net is a 5 tuple R= <P, T, Pre, Post, m > where:

- $P=\{p_1,p_2,\ldots,p_n\}$ is a finite, not empty, set of places.
- $T=\{t_1,t_2,\ldots,t_m\}$ is a finite, not empty, set of transitions.
- $P \cap T = \varnothing$, that is, the sets P and T are disjoint.
- Pre: $PxT \rightarrow R$ is the input incidence application. In other words, Pre(p_i, t_j) is the "weight" of the arc from the place p_i to the transition t_j.
- Post: $PxT \rightarrow R$ is the output incidence application. Post(p_i, t_j) is the "weight" of the arc from the transition t_j to the place p_i.
- m: $P \rightarrow (r_1,r_2,\ldots,r_n)$ is the initial marking.

2.2 Definition

Cellular Petri nets have two types of nodes: places and transitions. As usual, places are represented by circles and transitions by bars. These two kinds of nodes are connected by arcs (either from place to transition or from transition to place). Besides, the number of places and transitions must be finite and non zero.

A Cellular Petri Net is a tuple R= <D, F, C, P, T, Pre, Post, m> where:

- D is a finite set of non empty types.
- F is a finite set of functions. A function can be defined in terms of C and time variable, t.
- C is a finite set of cells. A cell encapsulates types and functions.
- $P=\{p_1,p_2,\ldots,p_n\}$ is a finite, not empty, set of places. A place encapsulates types and functions.
- $T=\{t_1,t_2,\ldots,t_m\}$ is a finite, not empty, set of transitions. A transition encapsulates types, functions, input guard functions Pre and output cells specifications Post.

- $P \cap T = \emptyset$, that is, the sets P and T are disjointed.
- Pre: PxT→Guard_Function is the input incidence application, that is, Pre(p_i, t_j) is an application from input arcs to a guard function from the place p_i to the transition t_j. It is a condition that specifies if a token fulfills the condition to fire the associated transition.
- Post: PxT→Cell_Specification is the output incidence application. Post(p_i, t_j)is an application from output arcs to a cell specification function of the arc from the transition t_j to the Place p_i. It is a set of conditions that the cells entering a place necessarily fulfill.
- m: C,P,T→DxF is the initial marking, that is, the initial distribution of instanced cells with their appropriate types and functions set to their initial values.

2.3 Biological Analogy

The first comment we should make is about the tokens. In Cellular Petri nets a token can be seen as its biological equivalent: cells. Cells have inherent attributes as in other paradigms such as Colored Petri Nets, but they have their own independent dynamics functions as well that evolve independent and asynchronously with respect to the network. The idea of embedding function in the objects can be found in [11], but with a scope of modeling systems where synchronization is important.

Many phenomena can be modeled directly thanks to this identification between marks and cells. The first and obvious idea is to model cells that evolve with time. For instance: Apoptosis (programmed cellular death) can be easily represented. Industrial examples can be found as well. This is the case of FMS (*Flexible Manufacturing Systems*) where the evolution of some token properties can be useful.

The concept of place should be commented too. Places in a cellular Petri net must be understood as higher level entities than cells. Places are containers for cells that have some properties associated. They can be seen like Petri dishes (invented by Julius Richard Petri, not Carl Adam Petri) where cells's state evolve. The evolution of the cells may vary as a function of the place's properties.

As in Hybrid Petri Nets [5] transitions are objects that work like source, sink or purely transition. Two ways of transition can be defined: discrete and continuous. The first one is the classical mechanism and the second is a variation of the continuous transitions of [5]: it is the *continuous flow* of a discrete entity (cell) through it. Note that transitions constitute a natural way of implementing cellular processes such as mitosis or cellular fusion. For these actions the incumbent arcs determine the requisites that the input/output cells have to verify. The transition defines the procedure that determines the output cells in function of the input ones.

2.4 Implementation

As we employ Object Oriented Programming concepts for the definition of the model, Object Oriented languages are the natural candidates to implement the proposed modeling methodology. However the proposed model is more as a common framework than a fixed group of rules that define a language. There is absolute freedom to carry out the implementation as long as the concepts are respected. For instance, a beautiful possibility would be to define the inner dynamics of the cells with simple low level Petri Nets. In this case, when the whole cellular Petri net is observed, we would have a very curious image: Petri nets "circulating" inside a bigger Petri net.

Nevertheless, we can provide some very general programming guidelines to perform simulations in a language such us C++. Token, places, transitions and arcs should be defined as different classes. Different timers can be employed so that all the dynamics in the CPN are covered. At the highest level a timer would indicate when the transitions can be fired. At a lower level other timers determine when the methods that implement the self evolution of the cells are called. Note that this implies that our models will be necessary timed oriented in a certain way.

So, in a general case, the structure of a cellular Petri net object should be like this:

Class CPNObject:

{

Properties:

Private: inner variables defined for programming reasons or not accessible from the point of view of the net.

Public: variables that determine the evolution of the object in the net.

Methods:

Private: functions that determine the evolution of the net in some way.

}

In [12] an implementation of a reduced version of the CPN can be found. The simulator was programmed in C++ and it allows communication with MATLAB through DDE to enhance its modeling capabilities. The definition of the CPN is stored in a queue of object places that point and/or are pointed by elements of a queue of transitions. A timer regulate the execution of the simulation: every object that can change with time has an associate time interval that determine when its methods are called to update their state or the CPN state.

2.5 Examples

2.5.1 Structured Superstore

This example illustrates possible applications of this modeling framework: a structured superstore. This is the case, for instance, of IKEA. In this kind of superstores the customers must follow a preestablished path. In fact, when one of these stores' map is analyzed, it is surprising its similarity with a Petri net.

Figure 1. IKEA Map

In this example, the different sections of the store are modeled like places and the customers like tokens. The transitions regulate the token flow from one section to another. At the end of the superstore, a problem of shared resources take place because there is a limited number of cashes to serve a much bigger number of clients. Clients' properties can vary during their route through the superstore as they are modeled like cellular tokens. Such properties could be, for example, shopping speed, numbers of bought items, readiness to go to the next section or probability of unexpected buyings. Different profiles with time varying properties can be used to model the clients' behavior. This allows a great variety of possibilities for simulation purposes.

For the set of performed simulations a simple model was implemented: different shopping probabilities have been assigned for every section in the shop, i.e. to each place. Three different types of customer profile have been defined and their behavior is influenced by the place properties where they are located. For example, the numbers of items that the customer buys in a determined section are modeled as the result of a rectangular random variable. The limits of the random variable are defined by the section where the client is. Shopping time is calculated as a function of the items bought in every section plus a random extra time.

$$IB(i) = PC(i) \cdot RECT(0, NM_i)$$

$$\text{NI}=\sum_{i=1}^{n} IB(i)$$

$$\text{ST}=\sum_{i=1}^{n}\left(IB(i) \cdot TPI(i) + RECT(0, ET_i)\right)$$

Where IB(i) stands for items picked in section i, PC(i) is the probability of buying items of a customer for section i, NMi is the maximum number of items that customer can buy in section i and ETi is the maximum extra time a client can stay in section i. Finally, NI and ST are respectively the total numbers of items bought in the n sections of the superstore and the shopping time from the beginning until the client reaches the cash queue.

This model can be used for control purposes as well. Let us imagine that every shopping cart or basket is equipped with a wireless transceiver and a little processor, so that it can inform the control network the shopping burden that it contains. This can be done by resorting to technologies like Zigbee (technology for wireless communication with low-power transceivers based on IEEE 802.15.4 standard) and RFID (*Radio Frequency Identification*). The model can be used for real time monitoring and the collected data employed to find an optimal way to distribute the available resources (cash, shopping assistants…).

We have performed the simulations for the Cellular Petri Net of the IKEA superstore in Seville. The calculations were made to study the clients' shopping time and their average number of items bought. Three different types of clients were defined for simulation: saver, normal and compulsive buyer. Each profile has its own features, but the main ones are related with their shopping capabilities, such as average time employed to buy an item, probabilities of buying items as a function of the section, etc. These parameters are dynamical, that is, they evolve for example as a function of shopping time or other variables such as the number of cells in a section. As can be seen, the possibilities are endless for the designer.

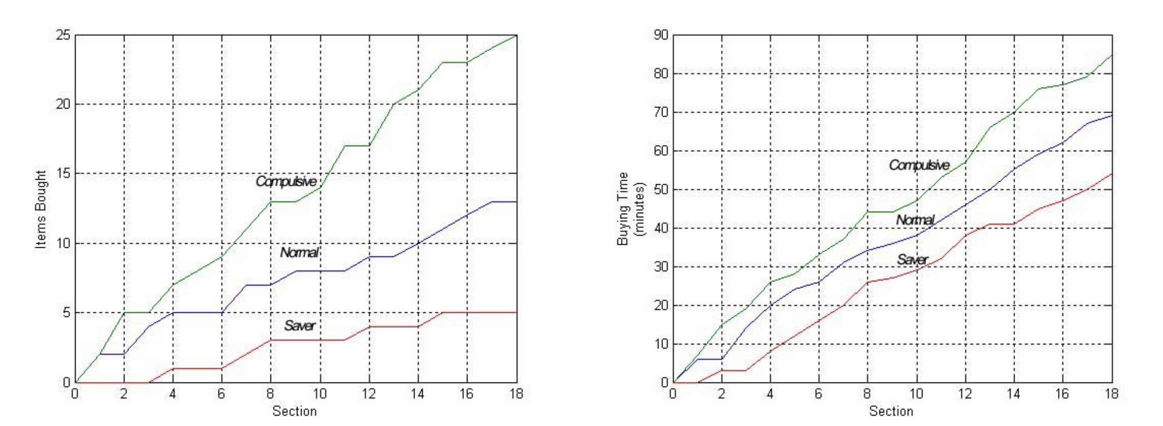

Figure 2. Left: Items Bought. Right: Buying time.

Some results of the simulations are presented in figure 2. This figure shows the average accumulated number of items bought and buying time employed for every type of client as a function of the section of the superstore. It can be seen how these graphics increase with the section.

Finally, in a more general case, any kind of superstore could be modeled with just applying the existing models for similar problems, like urban traffic networks, where the net structure is more complex than in this case.

3. CONCLUSION

A new Petri net framework to model hybrid systems has been proposed. The extension for the original Petri nets paradigm has a double inspiration source: object-oriented paradigm and biology. The cells' inner dynamics are encapsulated in the tokens, so that the complexity is hidden from the Petri net point of view. The advantages of this grey box approach to model hybrid phenomena result from the combination between the conceptual simplicity of Petri nets and complex dynamics modeling power of suitable programming languages such as C++.

This new perspective must be considered as a middle level Petri net paradigm. The reason for this is that it puts together aspects that belong to both high level and low level Petri nets. On one hand, the tokens have inner attributes and dynamics, but on the other hand, the complex dynamics are embedded into the tokens, that are alive by themselves.

There are grounds for thinking that the new paradigm could be helpful for fields like Medicine, Biology or distributed sensor monitoring. Besides, the new paradigm creates a bridge between two separate investigation fields like Petri Nets and multi agent systems. All these subjects can take advantage of its built in features so that new problems can be modeled or new and simpler models can be made for the existing ones.

REFERENCES

Jan Lunze, 2002. What is a Hybrid System?. Modeling, Analysis and Design of Hybrid Systems.Springer Verlag, Berlin, Germany.

Lygeros J et al, 2003. Dynamical properties of hybrid automata. *Automatic Control, IEEE Transactions on*

Volume 48, Issue 1, Jan 2003 Page(s): 2 - 17.

R Goebel et al., 2004. Hybrid systems: generalized solutions and robust stability }. *IFAC Symp. on Nonlinear Control*

Systems, Stuttgart, Germany.

R. Drath, 2002. Description of Hybrid Systems by Modified Petri Nets. *Modelling, Analysis and Design of Hybrid Systems*}.Springer Verlag, Berlin, Germany.

R. David and H. Alla, 2005. Discrete, Continuous and Hybrid Petri Nets. Springer Verlag, Berlin, Germany.

R.David and H. Alla, 1994. Petri Nets for Modeling of Dynamic Systems- A Survey}. *Automatica Vol. 30.* No 2. pp. 175-202.

I. Demongodin and N.T. Koussoulas, 1998. Modelling of Hybrid Control Systems using Petri Nets. *Proc. 3rd Int. Conf. ADPM'98.*

Ralf Wieting, 1996. Hybrid High-Level Nets. *Proceedings of the 1996 Winter Simulation Conference.*

K Jensen, 1997. Coloured Petri Nets–Basic Concepts, Analysis Methods and Practical Use. *Monographs in Theoretical Computer Science*. Springer-Verlag, Berlin, Germany.

Becker U., Moldt D., 1993 . Object-oriented concepts for coloured Petri net. *Systems, Man and Cybernetics.*

O. Biberstein, D. Buchs, N, Guelfi, 1997. CO-OPN/2: A Concurrent Object-Oriented Formalism. *Phd. Thesis.*

J. M. Maestre and E.F. Camacho, 2007. Simulador para Redes de Petri Híbridas. *Actas de las XXVIII Jornadas de Automática.*