Toward a New Formalism of the Petri Nets: Agents Petri Nets

Marzougui Borhen[1] and Mahfoudhi Adel[1]

[1] CES Laboratory E.N.I.S BP W 3038,
Sfax, Tunsia
Marzougui_bor@yahoo.fr
Adel.Mahfoudhi@fss.rnu.tn

Abstract. *This paper belongs to the domain of the engineering of the Petri Net (RdP). It aims at defining a new formalism for the modelling at the multi agents system (MAS). This formalism is based on agents called Agents Petri Nets (APN). That's why, the definitions that manipulate the internal state of the agent and its behavior are proposed. Indeed, the other formalisms are no more capable of modelizing the systems of large size and the interaction between the entities that compose them. The proposed mathematical definitions help us in modelizing in a rigorous manner and without ambiguity the interactive systems. To validate our approach, we will deal with simple examples.*

Keywords. *Petri Net, Formalism, Multi agents Systems, Agent, Agents Petri Nets.*

1. Introduction

Since the appearance of the Petri Net [1], the inventors have not stopped proposing new models, either to improve an already existing formalism or to create a new model. These formalisms resulted from studied system types. They permitted to make the conception more natural, more intuitive and more familiar by Petri Net. Indeed, the Petri Net can be considered as tools of modelling both graphic and mathematical. To modelize and analyze the discreet system, particularly the system competitors, parallel and not determinists, it is necessary to choose the appropriate type of RdP to be used. This type must be capable of modelizing with a rigorous manner the systems of large size as the systems multi agents. Such systems permit to coordinate the intelligent agent behavior interacting and communicating in an environment to achieve some tasks or to solve some problems [2].

According to [3], the modelling of an SMA proves to be applicable to represent the actions of the agents and their consequences in the environment that can be complex and of an autonomous evolution. Indeed the complexity of the studied system is increasing. The precision, reliability and the hardiness have become difficult factors to reach. Therefore, the integration of mathematical tool offers an exact way, in presence of graphic tools, to succeed the conception of these systems, particularly the systems multi agents. The objective of the present paper consists in proposing a new type of Petri Net based on the agents that helps us understand the functioning of the system multi agents. The previous works of the Petri Net concentrated on their uses and not on the creation of the new models as the works of [4], [5], [6], [7]. The research of a new model has been ignored for a long time. However, there were some works that took into account the extension of some classic types of the Petri Net to reach a more or less generic model to satisfy a need of modelling. In order to describe the behavior and the interactions of the entities of the system or the constraints on the variable characteristics of the system, we should make a dynamic modelling. This modelling must be achieved by an adequate formalism that will be presented in our work.

This paper is organized as follows: concerning the second section, it describes the formal methods. As for the third section, it invokes the system multi agents and their modellings by the Petri Net. In the fourth section, explain the limits of the classic RdPs. In the fifth section, our new model title Agents Petri Net will be presented and the definitions formulating this formalism will be interpreted. Then in the sixth section, a way of correspondence between the two SMA approaches and RPA will be given. Finally, in seventh section, this document will be concluded, giving some perspectives.

2. Formal Method

The formal methods have been used to assure a level of precision, consistency and quite elevated accurateness. They are based on mathematical foundations to decrease the risks of uncertainty and ambiguousness. In the phase of the software conception, the formal methods helps a particular language to express the properties descended of the problem specifications very rigorously [8].

However, these methods use the notations and the specific concepts that often generate a weak legibility and a difficulty of integration in the processes of development and certification [9]. The formal specifications are expressed in languages using the syntaxes and the very precise and strict semantics. The automatic validations result from a strong theoretical basis, so the integration of several formal methods is indeed difficult [10].

For the approaches employed at the level of the system specification, one can move toward two approaches: one classic and the other formal. Concerning the classic approach, it starts with a functional specification phase, then the general conception followed by a detailed conception and ends with the obtaining of the code. As for the formal approach, it starts with a general model that will be verified mathematically to get a refined primary model that undergoes other verification. This treatment is repeated until the obtaining of a refined and detailed model that drives to the coding.

A formal specification can bring many advantages. The methods are richer and permit to represent the static and dynamic aspects of the system better [10]. It is possible to classify the methods of formal specification into two groups. The first is the approaches based on the states as Z [11], object - Z [12], B [13], VDM [14], etc. Such approaches represent the system by two parts: the first is static, which permits the description of constituent and their states, while the second is dynamic, which describes the changes of states.

The second is the approaches based on the events they describe the system by processes or the communicating independent entities. Among these approaches one can mention: the Petri Nets [1], LOTOS [15], CSP [16], CCS [17], etc.

3. Multi Agents System and Petri Net:

A multi agents system permits to coordinate the behavior of agents interacting and communicating in an environment to achieve some tasks or to solve some problems [18], [19] and [20]. It allows the complex task decomposition in coins tasks which facilitates its development, test and updating.

The modelling of an SMA requires a verifying tool, in the first place, features and properties of agents, then, those of the system itself. The different applications of the multi agents system raise four domains: the resolution of problems by emergency, simulation, control of complex system and the environments of interaction man-machine (IHM). That's why, several works have been achieved on the formalization of the SMA by different formal methods as those of work [3], [21] [22], [23] and [24].

Little formalisms have been defined as the automatons of finished states, which prove to be inefficient when one must take into account the aspects of parallelism [5] and the algebraic models of difference equation that are as inefficient but essentially at the level of the representation of the agents in interactions. Then, one must have a formalism that must be capable to express the internal state of the agents, their behaviors and the interactions between them. In this context, one can mention the Petri Net. The use of the RdP to modelize an SMA presents a major contribution. For example, a Colorful Petri Net can modelize the simultaneous communications of the agents with the help of the functions manipulating some colors. This has been justified in numerous works as [25]. So, the Objects Petri Net (RPO) [26] presents a power to modelize the dynamic aspects of the agents.

4. Limits of the Classic RdP

The classic Petri Nets as those of Object (RPO), Place Transition (PT) and Colorful present an insufficiency at the level of their expression when it is about the system of large size as the multi agents system. These systems are characterized by the interactivity of the elements that they compose. For the Colorful RdPs, the classes of color cannot express the state of the elements for example directly (Tokens) of the system or the relations between them. Otherwise, the RPO can describe the internal state of the tokens but not the relations between them in an efficient manner because it requires the places and the supplementary transitions that put in game the utilized methods. Indeed, an Objects Petri Net modelizes a multi agents system by a quite elevated number of places and transitions by the invocation of a set of methods that describes essentially the behaviors of the agents around their environments.

The multi agents approach can be considered as an evolution of the object-oriented paradigm. From a conceptual viewpoint, an object is merely a data structure which is associated with the functions (cf. Figure 1) [27]. The agents are autonomous entities whose behavior does not depend on an outside expression, contrary to the objects.

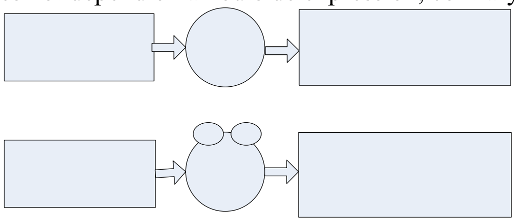


Fig. 1. Difference between object and agent [27].

The works already achieved are around the modelling of the SMA by an RdP respecting the load notebook. It is often needed to make a coupling between two types of Petri Net to satisfy a possibly determinist aspect in the system specification as the interaction and the communication between the different entities that compose it. Therefore, our idea consists in benefiting from the properties and features of agents and integrating them in a classic Petri Net. Thereafter, we propose our approach that consists in defining a new model of Petri Net called Agents Petri Nets. First of all, the general definition will be given and then every property with explanatory illustrations will be retailed.

5. Proposed Formalism: Agents Petri Nets

An agent is defined as an autonomous entity capable of communicating with other agents to discern at least its environment partially and the objects that are situated there, and to have correct or erroneous representations about the behaviors of a part or the set of the other agents of the environment [28]. So, contrary to the objects, an agent possesses an autonomous behavior. It is capable of taking some decisions and establishing plans of actions to accomplish complex activities. All agents do not have this degree of autonomy [7].

5.1. Definition 1: Agents Petri Nets

An Agents Petri Net is defined as being an oriented bipartisan graph that possesses two types of knots (place and transitions).

The bows are ties between these knots that indicate the conditions of activation of a transition. Every transition carries the functions that manipulate the internal state and the behavior of an Agent (Token) in its environment. The distribution of the tokens in the places in a given instant is called marking of the Agents Petri Nets.

A marking gives the state of the system that depends on the interaction between the entities that compose it. The change in internal of the state or the behavior of every Agent, in the first place or the whole system, in the second place, is assured by funtions.

In a formal manner, one calls Agents Petri Net the 9-uplet:

$$Q = \langle P, T, A, Meadow, Post, Pr_j, F, Ft, Env_k \rangle \quad \textbf{(1)}$$

where:

- P is a whole places finished but not empty,
- T is a whole finished not empty of transitions,
- A a whole finished not empty of agents,
- Meadow: $PxT \longrightarrow N$ an application of impact before,
- Post: $PxT \longrightarrow N$ a rear impact application corresponds to the bows,
- Pr_j: meadow condition of clearing,
- F (A_i, A_j): function relation agent that presents the condition of clearing,
- F_t: function agent that uses three variables:
- F_t (t_k) = <Per, value, Inter>,
- Env_j: Environment of work that describes system multi agents.

5.2. Definition 2: Constraints of an Agents Petri Net

A constraint on an agent is defined as: Cont (Ai, K, j)
Cont (Ai, k, j) is defined as being a meadow condition of clearing of a T transition descended of a P place.

In a formal manner, one defines the constraint on an agent exit of a P place as

$$\forall k \subset I, j \in J, \exists Cont(A_i, K, j) = b$$

Where:

- I: set of tokens of a place,
- J: set of places of a Petri Net,
- K: under I set of,
- j: number of place belonging to the network,
- Ai: Agent of indication (number) i,
- b: Boolean (0 or 1).

5.3. Definition 3: Function Meadow condition

Either Cont (Ai, K, j) =b
Either nk: number of elements of coins - together K. For a number nk of agents that enters into an environment:
cont(A1 , k , j) and cont(A2, k , j) ..and cont(Ank, K, j)=b
That gives:

$$\prod_{i=1}^{i=nk} cont(A_i, K, j) = b$$

The function meadow thus Prj condition descended of a P place of indication j is defined as:

$$Prj = \prod_{i=1}^{i=nk} cont(A_i, K, j) = b$$

By hypothesis, an agent Ai debit to be hired only Approx. in only one environment One defines Card (Env(Ai thus)) =1.

Interpretation of the possible values of Prj

The Boolean value sent back by Prj gives the starting point at an action (transition). An engagement of agent in an environment very determined Env will be preceded by controls it makes by this function. Under the hypothesis of uniqueness described above, the coin - together k of the agents has an equitable environment cardinality that is equal to 1or 0.

- If Prj = 0 then the condition of clearing is not valid and in this case at least exists an agent that did not respect the principle of uniqueness, of course it is already engaged in another environment.
- If Prj = 1 then the condition of clearing is valid and in this case one guarantees that all agents in question respected the principle of uniqueness.

Illustration (cf. Fig. 2) It is supposed that:

- Workshop1 contains the Machine M1 and M2,
- Workshop2 contains the Machine M3, M4 and M5:

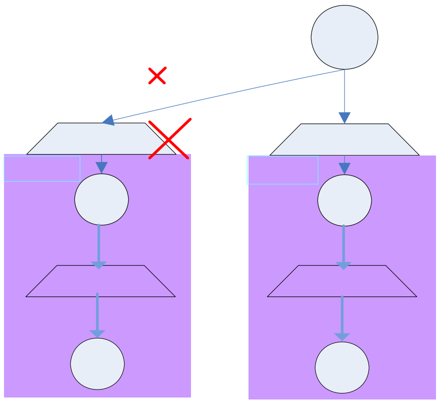


Fig. 2. Illustration of the Function Meadow condition

- Case 1: the two machines M1and M2 belong to the same workshop (Environment): Workshop1. In this case their use is permitted: meadow Pr0=1 condition,

- Case 2: the Machine M1, M3, M4 and M5, cannot belong to the same workshop (Environment: Workshop2) because the Machine M1 is already engaged in another environment. In this case one cannot clear the transition T1.

5.4. Definition 4: Function of adherence (relative to an agent)

This function gives birth to a relation between an agent and its environment. The engagement of an Agent Ai in an Envj environment describes a criteria of adherence in the first place then the number of time that this agent has been engaged in Envj.

It offers more mechanisms of explanations and minimizes the difficulties with the tasks that require knowledge of the world (Env) that cannot be gotten only by the memorization or the reasoning and not by the perception. One uses the definition of Ferber [2] that showed that a cognitive agent has the capacity to reason on representations of the world, to memorize some situations, to analyze them, to foresee some reactions possible for any action, to draw the conducts of the future events and therefore to plan its own behavior.

In a formal manner, we defined the function of adherence of an agent Ai, in an Envj environment noted Apai by:

Where

- b: constraint =Prj (b=0 or 1): the engagement of Ai in Envj,
- d: degree of adherence: whole gives the number of time that the agent Ai has been engaged in Envj.

Interpretation of the adherence function

At any time, this function gives a description of relation between the agent and a very determined environment. It guarantees the updating of the basis of an agent's knowledge. An agent's reaction depends on its environment. The evolution of Agents Petri Net depends on the system to study what implies, implicitly, that every agent looks for the criteria of expertise of another. That's why, it must interpret the value of **d**.

Illustration: (cf. Fig. 3) Let's take the example of the Figure 2 with some modifications:

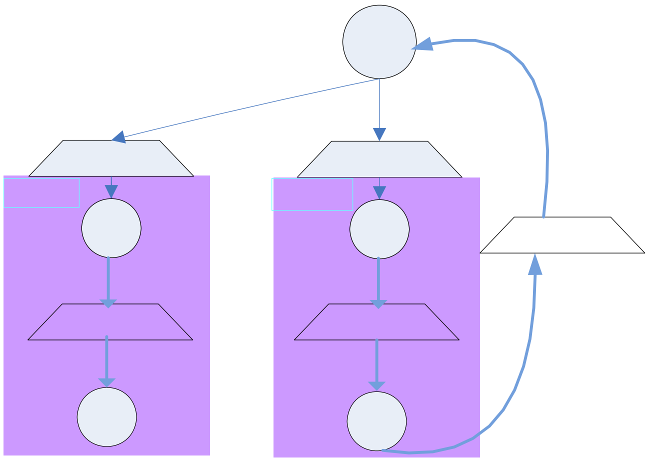


Fig. 3. Illustration of the adherence Function

<M1 , M3 , M4 , M5>

– Before the clearing of T0 and T1, agent M1 admits respectively as degree of adherence 0 for the Workshop1 environment and Workshop2:

  ApM1=Apa(M1, Workshop1, 1, 0)

  ApM1=Apa(M1, Workshop2, 1, 0)

$Pr_0=1$ T_1

– After the clearing of T0, agent M1 will have as degree of adherence 1 for the Workshop1 environment and 0 for Workshop2:

Wokshop2 Woksh

$P_{2'}$ M8

  ApM1=Apa(M1, Workshop1, 1, 1)

  ApM1=Apa(M1, Workshop2, 1, 0)

– Agent M1 leaves his Workshop1 environment: clearing of the T2 transition. It will allow him to be free.

T_{11}

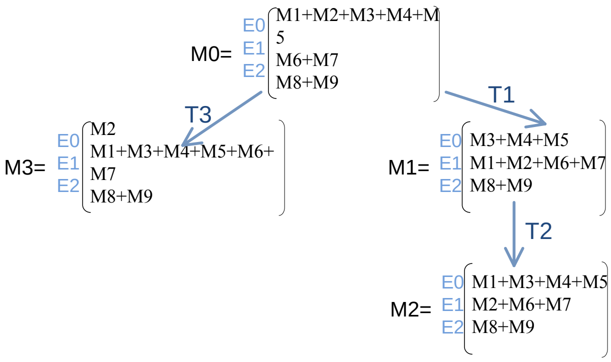
– After the shooting of T1, agent M1 will have respectively as degree of adherence 1 for the Workshop1 environment and Workshop2:

  ApM1=Apa(M1, Workshop1, 1, 1)

  ApM1=Apa(M1, Workshop2, 1, 1)

The function of adherence of every agent allows us to deduce the Marking of a RPA.

P_{22} M9



5.5. Marking of a RdP of Agents

A marking of an Agents Petri Net R is a family indexed by P. It is a Vector column whose composing is the number of marks in the place Pi at a given instant. A Petri Net marked of agents is a couple <R, M0, E0> in which R is an Agents Petri Net, M0 is a marking of R named initial marking and E0 is called Environment initial. A p place is a meadow condition of a t transition if a bow oriented of p toward t exists. Symmetrically, p will be a post condition of t if a bow joining t to p exists. The marking of a Petri Net evolves to every activation of a transition. Such an event is governed by rules of clearing: a transition can only be activated if the marking of the set of the places meadow condition allows it. With the activation of a transition, there is a consumption of the number of marks adequate in the places meadow condition and production of marks in the places post condition.

If the marking Mi is open to leave from the M0 marking after clearing the sequence of S transition = (S1, S2,… ..Si), then Mi=M0+CS.

Where S is the vector of n-size in which every SJ represents the number of time, the transition tj is cleared in the S sequence. C is the adjacency matrix of size n x n.

In the Agents Petri Net the clearing of a transition implies the evolution of the system. A sequence of clearing gives a casual manner a history of relation between two or several agents. The intelligence of such an agent is based therefore on its capacity to interpret this sequence. Indeed to next transition the agent must put in consideration the visited transitions that is to say the actions already realized. Therefore, the marking of the Agents Peri Nets presents a dynamic description of the agents in an SMA and a help of choice for the agents; i.e., the choice of criteria of expertise of the other agents.

The E and T wholes are finished and discreet; one can represent the applications Meadow therefore and Post under matrix shape.

Illustration

Matrix Meadow

Matrix Post

Adjacency Matrix

	T1
E0	M1+M2
E1	0
E2	0

Pré=

5.6. Definition 5: Function of adherence (relative to an environment)

The creation of the function of Apai adherence of an agent Ai in an Envi environment allows us to find a function of Apej adherence inversely. This new function describes the set of the agents that belong to the same environment j with certain degree of adherence di.

One defines the function of adherence related to an Envj environment as:
Where

- nk: number of agents of the environment,
- Ai: Agent of indication i,
- di: degree of adherence of agent of i indication.

Post=

	T1
E0	0
E1	M1+M2
E2	0

Thus, one can simplify this function and get:

$$Ape_j = Ape(Env_j, \bigcup_{i=1}^{i=nk}(d_i))$$

Illustration:
From the RPA of Fig. 2 one can deduce the degree of adherence of every environment:

ApeWorkshop1 =Ape (Workshop1, $\bigcup_{i=1}^{i=5}(d_i)$)

C=

	T1
E0	-(M1+M2)
E1	M1+M2
E2	0

ApeWorkshop2 =Ape (Workshop2, $\bigcup_{i=1}^{i=5}(d_i)$

With 5 is the number of machinery used.

One can deduce the following adherence matrix:

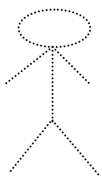
	M1	M2	M
Workshop 1	1	1	0
Workshop 2	1	0	1

5.7. Definition 6: Agent Moderator

An agent is said to be moderator if it is important by contribution to another. The important term indicates that the moderator dominates at the time of a communication, or it possesses a hierarchical degree (dh) less elevated (dh=2 dominates dh=3).

An agent is said to be total moderator if it is important by contribution to all agents of its environment. Thus, it possesses a hierarchical degree dh that is equal 1.

Illustration : <M1, M2, M3, M4, M5>



Agent total Moderator

Agent Moderator

Agent Moderator % à M4 and M5

5.8. Definition 7: Function of relation agent of order 2

We defined a relation of order 2 as a function admitting two entries E1 and E2 and in exit only one value Boolean S. The entry E1 is imperatively moderator. This function presents a meadow condition of clearing of a transition.

Thus, we defined this relation by the function F (E1, E2) =S.

Let's have two agents Ai and Aj in the same way environment Env.

$\forall A_i \in A, \forall A_j \in A \exists\ a\ Function\ F$ as F(Ai, Aj)=S,

Where

– A: set of the agents of the environment,

– Ai: is an agent moderator,
– Aj: is not an agent moderator,
– S: Boolean value sending back 1 or 0.

Thus, one can generalize this function to get a function of relation agent of n order: set of n agents a function F as F (Ai, Ai,… An) =S

Interpretation of the possible values of S:

- If S = 0 then no relation exists between the two agents that communicate themselves. In this case, the agent non moderator cannot enter in relation with the agent moderator voluntarily, or forced by the agent moderator of total order, or because it is already occupied.
- If S=1 then one establishes a relation between the two agents concerned. In this case, the agent moderator asks for the establishment of a communication with another agent that is called non-moderator and which accepts this demand.

Illustration :(cf. Fig. 4)

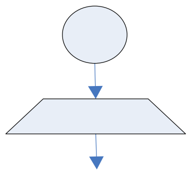


Fig. 4. Function of relation agent of order 2

5.9. Definition of the function Ft

The function agent describes the relation between two agents that communicate with each other, the data interchange and the behavior of each of them. It modifies the values descended of an agent directly. These define the capacity to discern and to react to the modifications occurred in its environment. Generally, it is written as follows:

$$F_t\ (t_k) = <Per, Inter, Value>.$$

Interpretation of the possible values of Function Ft

– Initially, Ft (tk) = <0, Φ, 0> it implies that there is not any interaction between the agents. If the value of *Per* = 0 then directly we have *Inter* = 0. Never can we have *Per* = 0 nor and Inter = 1. *Value* =Φ, in this case no action is triggered and in guard the previous situation of the agents,

– In the course of the clearing of the transition tk, there will be change of values between the agents. In this case *Per* takes the value 1, *Inter* takes the value 0

M1
M2 M

F(M1, M

and *Value* defined the action or the task to achieve. The relation of order already defined gives the sense of transfer of the information. So:

Ft (tk) = <1, *Value*, 0>

– After the clearing of the transition tk, *Inter* takes the value 1; it indicates that the action has been achieved with success. So:

Ft (tk) = <1, *Value*, 1>

Illustration: (cf. Fig. 5)

– Initial state of treatment: the two Machine M1 and M2 are waiting: *Per*=0, *Value* = Φ and *Inter*=0

– The M2 machine wants to pass the P4 piece to be treated by M1: *Per* =1 *Value* =M2.Passe[P4] *Inter*=0

– The M1 machine accepts the demand for M2: *Per* =1 V*alue* =M2.Passe [P4] *Inter* = 1.

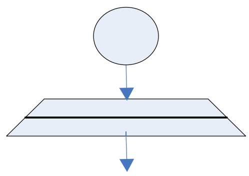


Fig. 5. Function agent of order

Interpretation of the possible values of Per:

Per is a Boolean value that puts in relation two agents Ai and Aj through a transition T. If Per = 0, then the agent Ai did not send information to agent Aj,

So *Per* = 1, then the agent Ai send information to agent Aj.

Interpretation of the possible values of Inter:

Inter is a Boolean value that validates the transmission or the exchange data between two agents Ai and Aj through a. transition T.

– If Inter=0, then agent Aj did not receive the information of agent Aj or it refused it,

– If Inter = 1, then the agent Ai received information sent by agent Aj. Indeed the task has been achieved. Then it is a value of validation.

Interpretation of the possible values of value:

Value is an action achieved by the two agents Ai and Aj. This action represents the transference of data from the agent moderator Ai toward the other agent Aj. It is a data structure going from the simplest (whole, real, character, Boolean) to the most complex

M1
M2 M3
F(M2 M1)
F_t=<1 M2 Pass

(table, matrix, object, class of object, composing, Agent). Value modifies the behaviour of an agent.

6. From the Multi agents Systems Toward the Agents Petri Nets

Under the shape of a Table a way of correspondence between the two approaches according to very determined features will be given.

Characteristic	**Multi Agents Systems**	**Agents Petri Nets**
Name	Agent State of the system Set of rules Set of relations of actions d'actions	Token Place Meadow condition (Pr_j) Transition
Class	Agent Administrator Agent Reactive Agent Cognitive Agents Hydride	Agent Total Moderator Agent not Moderator Agent Moderator Agent Total Moderator
Autonomy	Interaction between agents	Function relation of order 2 or n
Reactivity	Agent - Agents	Function Agent F_t=<*Per, Valeur, Inter*>
Heterogeneity	Agent – Environment	Related to adherence Function an agent:
Sociability	Environment – Agents	Related to adherence function an agent (Apa_i) and her environment : $Ape_j = Ape\,(Env_j, \bigcup_{i=1}^{i=nk} (d_i\,)$
Intelligence	Comportment, capacity of interaction	Exploitation of the values possible of *Per*, *Inter* and *Value*

7. Conclusion

In this paper we defined the setting of our work while expressing our position in relation to the works that treated the formal methods for the modelling of the system multi agents. We proposed a formalism that combines the Petri Net and the SMA. This combination gives birth to a new formalism called "Agents Petri Nets". We gave all definitions in relation to this type of network. It takes advantage of features of the agents and systems multi agents. Indeed, each token of a place represents an agent and the transition is endowed with a set of functions that describes particularly the condition of its clearing and the relations between the agents. The major contribution of an Agents Petri Net by contribution to the others is its power of expression, modelling of the interactions between the agents, the remarkable reduction of size of network and the gain at the level of modelling time. The definition of this model helps us to modelize in the internal state efficiently and the dynamic behavior of an agent in an SMA. We will study the modelling of other real cases of different domains more deeply in our future works.

References

1. Petri, C.A.: Communication with Automats, thesis of doctorat at the Boon University, 1962.
2. Fougères, J.: A cognitive architecture of communicating agents in complex information systems, M3M Laboratory, the Belfort-Montbéliard University of Technologies, 2002.
3. Ferber, J.: The multi agents system: towards a collective intelligence, InterEditions, Paris, 1995.
4. Tagne, E., Viho, C, Tonye, E. and Akono, A.: Modelling of architecture of system multi agets with Petri Net, IEEE SITIS, 2005.
5. Kefi, M., Korbaa, O., Ghedira, K., Yim, P.: Formalization of muli agents model with Petri Net: application to problem of storage of centenaries, 6th French speaker Conference of Modeling and Simulation-MOSIM'06, Rabat, April 2006.
6. Caspi, P.: Elements for the choice of methods of development of critical software systems, Research report N° TR-2005-17, November 2005.
7. Quesnel, G.: Formal and operational approach of multi modelling and simulation of complexe systems: contributions for modelling of multi agents systems, thesis of doctorat, December 2006.
8. Lamy, P., C.Blaise, J.: Use of formal methods, J'automatise N°42, September-October 2005.
9. Idani? A .: B/UML: Setting in relation of B Specification and UML description for help of external validation of formal development in B, the Grenoble 1University, thesis of doctorat, November, 2006.
10. Gervais, F.: EB4: Towards a combined method of formal specification of information systems, thesis of doctorat, the Sherbrooke University (Québec), Canada, June 2004.
11. Spivey, J. M.: The Z Notation, a Reference Manual. Prentice Hall, 2nd edition, 1992.
12. Smith, G.: The Object-Z Specification Language, Kluwer Academic Publishers, 2000.
13. J.R. Abrial, The B-Book: Assigning programs to meanings, the Cambridge Press University, 1996.
14. Jones, C.B.: Systematic Software Development using VDM, Prentice Hall, 1990.
15. Bolognesi, T. and Brinksma, E.: Introduction to the ISO specification language LOTOS, Computer Networks and ISDN Systems, 1987.
16. Hoare,C.A.R.: Communicating Sequential Processes, Prentice-Hall, 1985.
17. Milner, R.: Communication and Concurrency, Prentice-Hall, 1989.
18. Yoo, M-J.: A componential for modeling of cooperative agents and its validation, Thesis of doctorat, the Paris 6University, 1999.

19. Ferber, J.: A Meta-Model for the Analysis and Design of Organizations in Multi agent Systems, Proceeding of the 3rd International Conference on Multi-Agents Systems (ICMAS'98), IEEE CS Press, June 1998.
20. Seghrouchni, A.: Protocol Engineering for Multi-Agents Interacion, 9th European Workshop on Modelling Autonomous Agents in Multi-Agents
21. World(MAAMAW'99), 1999.
22. Ferber, J. and Gutknecht O.: For an operational semantic of multi agents system, Acts of 8eJFIADSMA, Speak french days of distributed artificial intelligence and multi-agents sytem, 2000.
23. Gruer, P., Hilaire, V., Koukam, A. and Cetnarowicz, K.: A formal framework for multi-agent systems analysis and design, Expert Systems with Applications, 23(4):349–355.2002.
24. Duboz, R., Ramat, E., and Quesnel, G.: Multi agents Systems and theory of monetization and simulation, Acts of French speaker twelve days about Multi-Agents Systems (JFSMA), Paris, 2004.
25. Suna, A.: CLAIM and SyMPA: a Environment for programming of intelligent and mobiles agents, thesis of doctorat , the Paris 6 University, December 2005.
26. Seghrouchni, A., Haddad, S. and Mazouzi, H.: Distributed Observation and analyses interactions in a multi agents system, Speak French days, IAD and SMA, Nancy, 1998.
27. Sibertin, B.: High-Level Petri Nets With Data Structure: Petri Net and Applications, Finland, 1985.
28. Briot, J.P.: Introduction to agents, DEA IARFA, the Paris 6 University - CNRS, 2002.
29. Brams G.W.: Petri Net: Theory and Practical, Volumes I et II, MASSON, Paris, 1982.