Toward a modelling of the UML filtering mechanism by the Petri networks

Boubker Sbihi
Equipe EIT
24, Résidence al Houda, Massira 1, Témara
Bousbihi@menara.ma

ABSTRACT

Our paper treats the theme of the modelling by the Petri networks of the UML filtering mechanism. Indeed, UML 2.0 is nowadays the most language used for the object modelling. It contains many diagrams that allow him to make a clear and detailed representation of the object model.

However, the UML class diagram is not very adapted in the case of the fusion of several classes' models in modelling multi-views. From where the idea to use the Petri networks formalism in order to facilitate and to automate the fusion in such a context.
We will associate in a first time a Petri networks for every UML diagram of classes relative to one point of view, before integrating the filtering mechanism because of it contribution in the modelling of multi-views systems. Finally, we adopted the concept of colored Petri networks to model the fusions.
KEYWORDS: Point of view, UML, Mechanism of filtering, Petri Networks.

1. Introduction

Since the third millennium, the users of the computer systems became more demanding for functionalities, diversity of the information and the conformity to the users needs. This is why these systems must be evolutionary to be able add new users witch having a large access to the data of all natures. The previous decade has been marked by a lot of works concerning the access to the heterogeneous data in the domain of the data bases and the integration's platforms such as CORBA, Java Beans and other. In the same decade, some other works were introduced, concerning the models and the steps for the object analysis and the conception such as the Unified Modelling Language witch is principal element of this research. The modelling of the complex systems [Lemoigne 1990] cannot be achieved by only one point of view because of the different needs of every user and its rights of access to the information. These different needs require the construction of several partial models corresponding to every actor of the system (the developers and the users). The interdependence of these models creates large problems of consistency and redundancy.

The use of the classic technology oriented object has given important progresses for the modelling of this type of system.

It also brought a big homogeneity in the formalisms representation, a big possibility to express by the mechanisms of encapsulation, of masking the information and the best

maintenance and reuse, etc. But, it suffers from the insufficiency to solve all problems put by the complex modelling systems.

The introduction of the point of view concept in the development oriented object of the complex systems permits during the process of development to consider different users and their rights of access to the system.

It is characterized by its development of only one divisible unique model, accessible according to several points of view instead of several interdependent under-models correspondents to the different points of views of the systems to modelling.

It allowed to solve the majority of the problems and brought several improvements in the modelling of the complex systems. The points of view approach guarantees the consistency of the data, the deletion of some redundancies, the enrichment of the multi-models approach, the management of the rights of access, the centralization of the knowledge and the possibility to evolve the points of views.

The Unified Modelling Language [UML 2.0] standardized by Object Management Group [OMG] is the result of the fusion of several oriented objects methods and became the reference of the oriented objects modelling. It covers the static and dynamic aspect of a system according to its different diagrams. For it, UML define ten diagrams to represent the different points of view of the modelling. Its goals goal is, to specify, to visualize, to construct and to document the computer systems especially the complex systems.

However, Only UML can not take in account the notion of points of view in its different diagrams except the class diagram witch proposes three type of a static visibility (Private, Public and Protected). This visibility cannot change dynamically and however can't not answer to the users needs of the complex systems. The UML suffer also from the non existence of a development method so it propose a group of diagrams covering the object modelling system contrary to any development method of software. This situation leads us to combine it so that supports the notion of points of view; witch is the main concept of the development of the complex systems, with different types of users. In order to integrate the notion of views and points of views in the object modelling several works of research have been realized by our team and these research allowed to us to create the VBOOM method (View Based Object Oriented Methodology) [Coulette and al 1996] and its adaptation toward the UML, U_VBOOM [Hair and al 2004] and finally the VBOOM++ method [Sbihi and al 2003] based on the filtering mechanism and UML.

In the VBOOM method and these variants, the fusion phase of the visual diagrams belonging to each actor of the system is semi automatic. In this research, we will use a new approach and will add the formalism of Petri networks [Badouel and al 1999] that will have for goal to automate and to facilitate the fusion steps. This automation will associate to each actor class diagram one Petri networks diagram and fusion Petri networks diagrams automatically and deducts the finally classes diagram.

Our work in a first time, consist achieved a relation between a UML classes diagram and a Petri's network. In this paper, we begin with presenting the UML filtering mechanism (section 2), then, we present the Petri networks (section 3), after that we propose our approach of relation between a Petri networks diagram and UML class diagram (section 4); finally, we present a realization by the CPN-TOOL software that represents an important several advantages (section 4). We finish this article by a conclusion that will include some perspectives.

2. The Filtering Mechanism

Since the fusion of the methods objects dominate (OMT, Booch and OOSE), and standardization by the OMG group in 1997, UML is the most used standard in the oriented object domain. UML gives a more formal definition and bring the methodological dimension which does not exist in the object approach

One of the main concepts of the object paradigm is the encapsulation. Indeed, the oriented object programming allows hiding the implementation of an object.

It allows him to reach these attributes only by some methods named interface used for this operation. It is possible to define some levels of encapsulation, in order to define the class type having access to the attributes. There are three levels of visibility that are: private, public, protected. For the attributes, UML defines three properties that are: Changeable, Frozen, Read-Only. The UML visibility is a static one. The possibility to modify this visibility dynamically during time is not possible but our filtering mechanism can do it. In the same way, an attribute to whom we wants to give the privilege of reading for some points of views, the modification for others and to mask its value for the remainder, cannot be a visibility nor public, nor private, nor protected. what proves the insufficiency of the UML.

Because of the inability of UML to managing the dynamic aspect of the visibility, that means supporting the dynamic change of the points of views, a filtering mechanism has been conceived to permit the filtering of the primitive according to the one point of view wanted.

The filtering mechanism based on the UML visibility was the object of several researches [Sbihi and al 2003]. This mechanism is characterized by its support to the dynamic aspect (possibility to allocate and not allocate the visibility of one point of view on a primitive. Indeed, the filtering mechanism defines a right of access to each primitive for each of point of view. These rights of access can be modified dynamically during time. Some new visibilities in more of those of the UML were defined in the following figure (Figure 1):

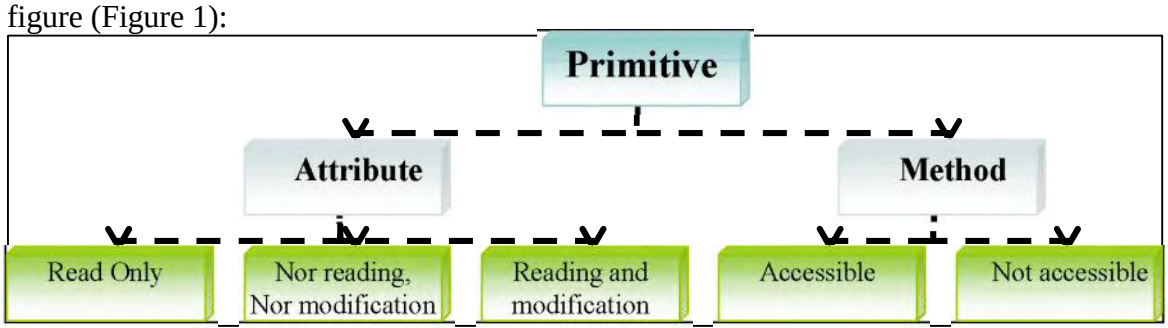

Figure 1: Types of rights of access for a primitive in the filtering mechanism

In a class diagram, we distinguish two types of primitive:

- Primitive divisible public: having "3" as value of visibility for the attributes and "5" for the methods, these values are applied for all points of view.
- Primitive multi-points of view whose visibilities depend on the points of view.

The same distinction is applied on the classes. So, a class containing at least one primitive multi-points of view is called multi-points of view class. On the other hand a class of which all primitives are divisible public is called public divisible class. So, the

filtering mechanism permits to conceal the different multi-points classes of view and to make them available only through preset methods of filtering taking the points of view like arguments [Sbihi 2005].

The filtering mechanism is implemented by the instantiation of a filtering class of which all multi-points of views classes inherit. This class possesses some methods with the public UML visibility taking for parameter the different points of view.

To distinguish the filtering class of the other classes, we add to it the UML stereotype << Filtering >>. To understand more in detail the filtering mechanism, we take for example the class course witch can be seen according to two different points of view, the first point of view is the student and the second is the teacher. The rights of access on the information and services of these actors are different in the class Course.

In the same way these actors don't have the same needs and the same responsibilities. Certainly, we can always realize some mechanisms to make controls of the access to a class; but such a solution is difficult and increases the complexity of the class extensively. It is the reason for which these controls will be made in another class and the access to theses primitives will made by the relation of inheritance. This technique allows us to benefit from the reutilisability of the code. The class course can be illustrated in following figure (Figure 2):

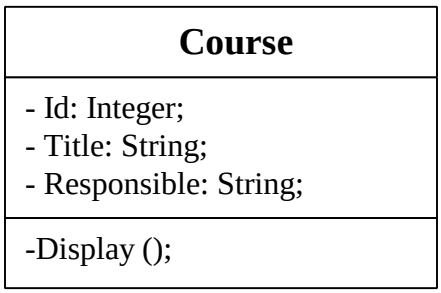

Figure 2: The class" Course"

In the filtering mechanism, we will give to each primitive multi-points of view, the private value of visibility UML. This affectation aims to forbid all type of direct access to the attributes and to the multi-points methods. On the other hand the access to the primitive will be made according to some methods predefined and that possess the points of view like parameters.

The multi-points of view primitives of the class course can be gathered in a table witch name is the dictionary of the primitive multi-points of view. This dictionary illustrates the points of view and their rights of access on the multi-points of view diagram global that appear in the multi-points of view attributes and methods. For the case of the class Course, we can model it according to the following dictionary (Table 1):

Name of the class	N° et Name of primitive		N° of point of view	
			Student	Teacher
Course	1	Id	2	3
	2	Title	2	3
	3	Responsible	2	3
	4	Display() ;	4	5

Table 1: Excerpt of the primitive multi-points of view dictionary for the Course class

Only the multi-points of view primitive that must have a visibility private UML are regrouped in this dictionary. This filtering can be applied for all UML diagram or a fusion of a some UML diagrams, what integrates the points of view notion in a UML diagram of class.

3. Formalism of the Petri's network

The Petri networks formalism is a tool permitting the study of the dynamic and discreet systems. It is about a mathematical representation permitting the modelling of a system.

The analysis of the Petri's networks can reveal the important features of the system concerning its structure and its dynamic behaviour. The resulted of this analysis are used to value the system and to permit the modification.

The main concepts of a Petri networks are the events that are the actions playing in the system and the conditions that are the predicates describing a state of the system. A Petri networks (RdP in French) is a bipartisan graph made of two types of summits: the places and the transitions.

The oriented arcs join some places to some transitions or some transitions in some places. Every place can contain one or several tokens, represented by points. These tokens permit to model the dynamics of the system.

The marking of an RdP is a vector with positive or null whole component, and whose dimension is equal to the number of places. The last component of this vector represents the number of tokens that appears in the last place of the RdP.

In a more formal manner, a RdP is 5-uple PN=(P, T, A, W, M_0) where:

- $P=\{p_1, p_2, \ldots, p_n\}$ is a finished group of places,
- $T=\{t_1, t_2, \ldots, t_q\}\}$ is a finished group of transitions,
- $A \subset (P\times T) \cup (T\times P)$ is a finished group of arcs,
- $W : A \rightarrow \{1, 2, \ldots\}$ is the function weight attached to the arcs,
- $M_0 : P \rightarrow \{0, 1, 2,\ldots\}$ is initial marking.

. The works of research concerning the Petri networks can be classified in three categories:

- The theoretical study of the different types of Petri networks such as the temporized RdPs, the stochastic RdPs, the colourful RdPs,…
- The use of the RdPs in the conception and the management of the dynamic systems as the systems of production, the problems of organization, synchronization and cooperation between units working in parallel.
- The use of the RdPs with other abstract methods, we can mention in this case the achieved works in [Benzina and al 1997] concerning the representation of the diagrams (Structured Analysis - Real Time) with the help of the Petri networks.

4. The modelling of a filtering class by the Petri networks

UML represents a big advantage for the developers because it allows them to model their system by a universal language while benefiting from the force and the large possibilities that offers the object paradigm. On the other hand, its association with the filtering mechanism made it again more effective and able to model the most complex systems. Indeed, in a modelling of the complex systems, the evolution of these systems and their continual need to evolve impose to find a means to merge the diagrams of

classes in a systematic way and to distribute the work of modelling on several steps. The most adequate way to achieve this fusion is to convert the diagrams of classes in a mathematical formalism that assures a simple and automatic fusion. It is the case of the Petri networks that permit to study complex dynamic systems. The first approach that we followed to achieve this modelling consisted to take only one class in a first time and model it by a Petri network.

In this sense, we decided to work on the presented class course, in the previous paragraph and that offers a favourable environment to experiment our approach and to test its validity. The modelling of this class by a Petri networks can be made as the following manner:

Objects of the diagram of classes	**Representation with the Petri networks**
A point of view	A place
A primitive	
The access of a viewpoint to a primitive	The clearing of a transition
The access to a primitive without access rights	The weights of the bows

Tableau 2 : Rules of relation between the diagram of classes and its corresponding RdP

There for, we have an initial marking for each point of view. This marking permits to model the right of access of each of the points of view relatively with the primitives. On the other hand, we allow the administrator of the system to change the rights of the different points of view dynamically, and this with acting on the matrix presented in the previous paragraph. The intuitive modelling will take the different values of the matrix like weight of the arcs between the points of view and the primitive multi-points of view. This modelling is illustrated in the following figure (Figure3):

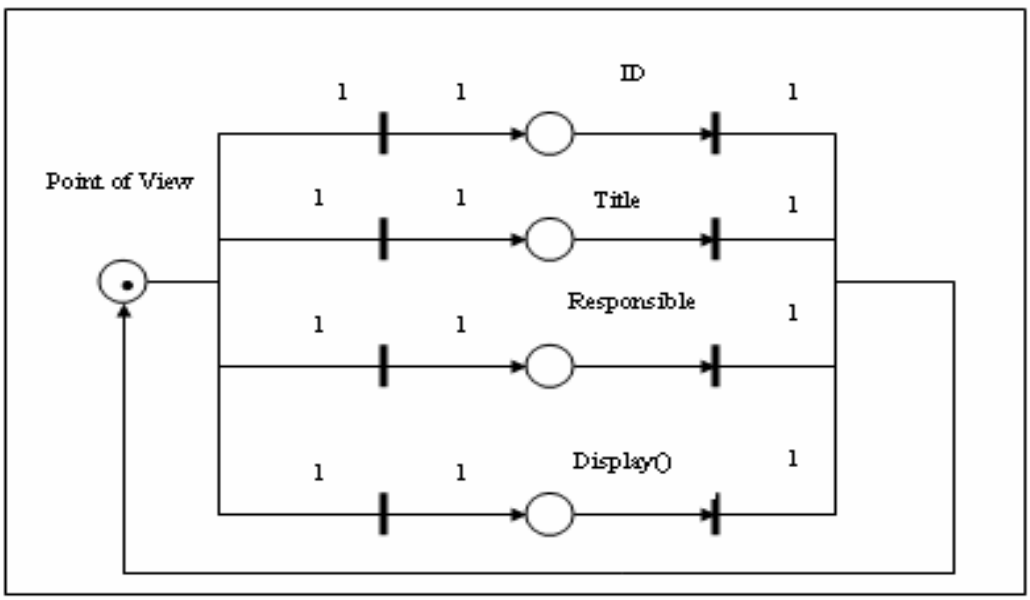

Figure 3: The Petri networks of the class" Course" of an intuitive modelling

This modelling concerns only one point of view that reaches all primitives.

As illustrate on the diagram, all weights of the arcs have as value 1. For the case of two points of view, for example a student and a professor, but the student don't have right of access to the attribute mark. The proposed solution is translated in the following Petri networks (figure 4):

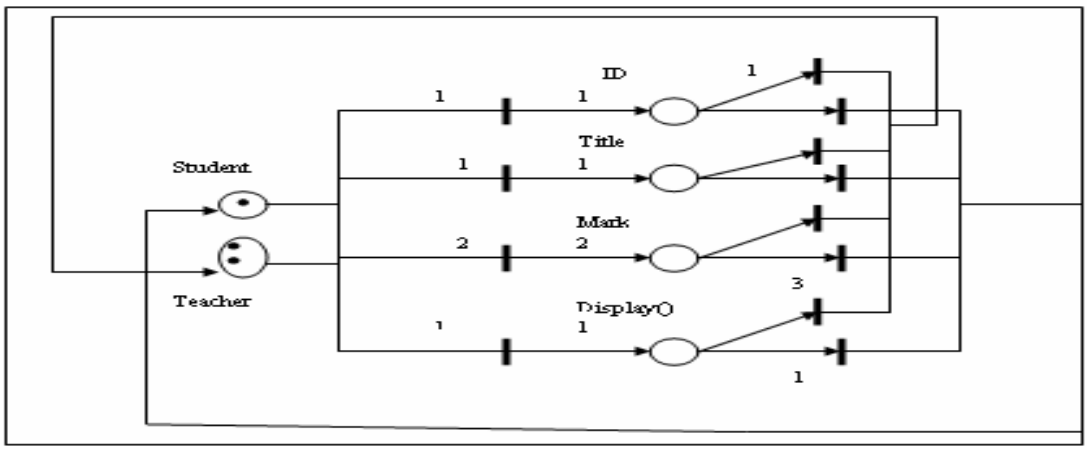

Figure 4: The Petri networks of the class" Course"

Through this network, we see that the point of view student cannot reach the Mark place because it doesn't have the necessary marks to cross the transition. In the other side, we can say that the primitive Mark will never fill the place of the point of view student because the weight of the arc to cross the corresponding transition is superior to the marking of the place Mark.

However, we can not manage through this modelling the respect of the initial marking of the points of views, because, the teacher can activate the ID place very well, and to deactivate this place, it is the student who receives the token, and can activate the mark place whereas he doesn't have the right of it. So that, we noted that this modelling had the disadvantage to be not able to indicate in a precise way the origin of the marking. Indeed, In one instant, we don't know, in a precise way, the origin of the activation of a place (primitive). Of this fact, we can't give its initial marking to one point of view allowing him to reach others primitives.

Therefore the optimal solution would consist in using the possibilities that the colorful Petri networks offer to exceed all confusions generated by the modelling with the Petri networks. Indeed, in this new approach, we are going to give to each point of view a specific color and model the different rights of access available to know reading writing and modifying by the number of the tokens for every color. While adopting this approach, we will try in the following paragraph to explain it more in detail while using the CPN Tools.

5. Realization with CPN Tools

CPN Tools permits to publish, to simulate and to verify some colorful Petri networks hierarchical of large size. Distributed and maintained by the CPN Group, CPN Tools replaces Design CPN that has been used by more of seven hundred fifty organizations (of which two hundred industrial) in fifty countries. It uses the language CPN ML, variant of the Standard Meta Language that is used by 1250 users in 77 pays. A comparison of the tools supporting the Petri networks is illustrated in the following figure:

	Modeless	Analysis	Auto completion	Undo/Redo	Cut/Copy/Paste	Graphical Editor	Continuous Simulation	Step by Step Simulation	Feedback	Animation
CPNTool	✓	✓	✓	✓	✓	✓	✓	✓	✓	
Viptool		✓		✓	✓	✓	✓	✓	✓	
PIPE		✓				✓		✓	✓	
Visual Object Net++						✓	✓	✓	✓	
JARP		✓		✓	✓	✓		✓	✓	
Renew			✓	✓	✓	✓	✓	✓	✓	
STNPlay		✓				✓	✓			
Petshop [1999]	✓	✓	✓	✓	✓	✓	✓	✓	✓	
PACE 5.0+		✓	?	?	?	✓	✓		✓	✓

Table 3: The comparison of several tools supporting the Petri's network

According to a previous table, the CPN-tools present a important number of advantage and it is the reason of our choice. In our case, it will serve to simulate the Petri networks and will permit to validate our approach and so that it can spread this network in order to representing all multi-points of view classes of the system. Thus, the administrator of the system will be able to use it to verify the respect of the rights of access of the different points of view. For modelling the colorful Petri networks equivalent to the following matrix (Table 4):

Primitive	**Right of Access**	
	Student	**Teacher**
ID	2	2
TITLE	2	3
REMARK	1	3

Tableau 4: Visibility of the attributes of the class" course"

The colorful Petri networks correspondent is illustrated in the following figure (figure 5):

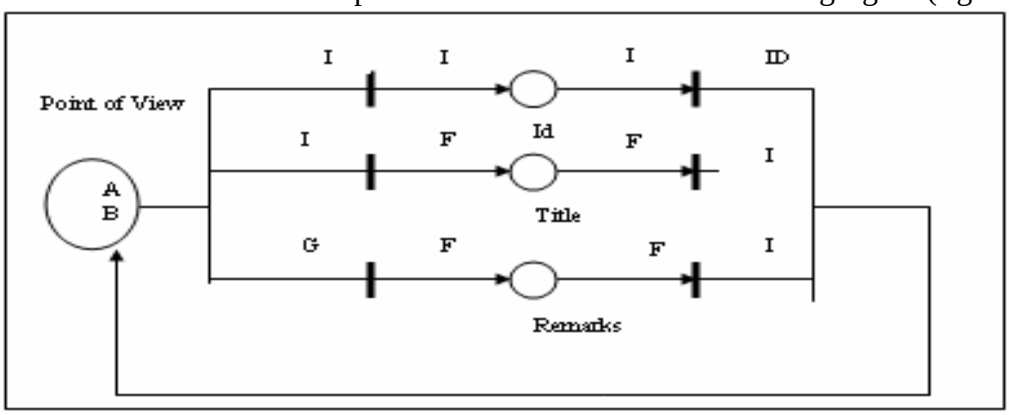

Figure 5: The colourful Petri networks of the class course

With:

- The tokens "a" represents the point of view student. There is initially one token a

- The tokens "b" represents the point of view Teacher. There is initially one token b
- F, ID,G are as:
  - F(a)=a, F(b)=2b,;
  - G(a)=2a,G(b)=b;
  - ID(a) = a,ID(b)=2b;

We starts with drawing the network, we will need an initial place, one place for each primitive and some transitions joining the different places like illustrate in the following figure (Figure 6):

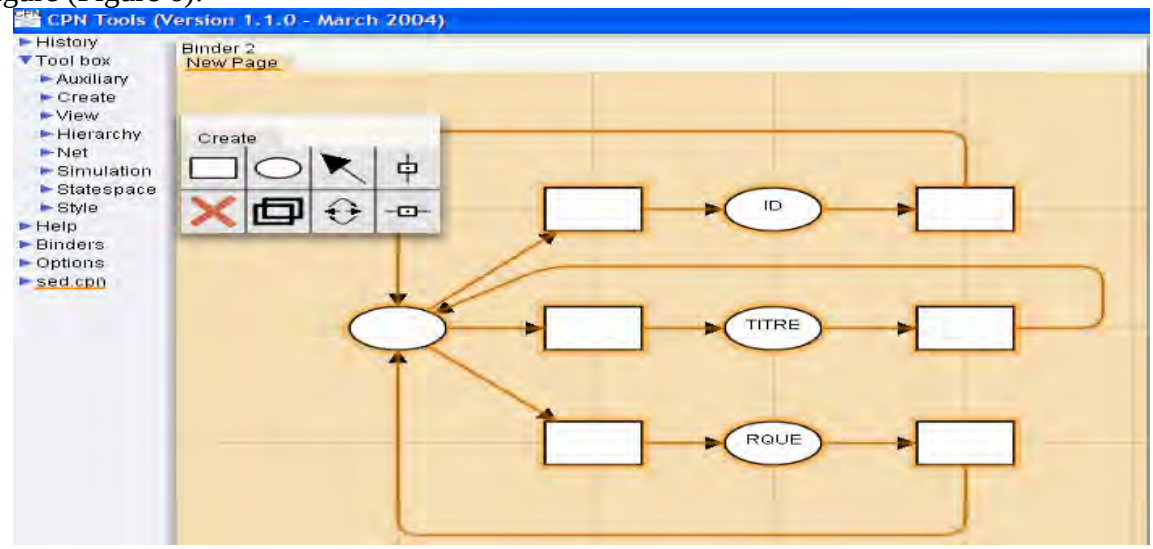

Figure 6 : Diagram of the colorful Petri networks under CPN Tools

It is necessary to give a color to every point of view: "a" a color for the teacher and the "b" color for the student for example. The declaration of these two colors made by the creation of a simple type, "pt_vue" for example uniting them:

color pt_vue =with a | b;

The Declaration of the colors is presented in the following figure (Figure 7):

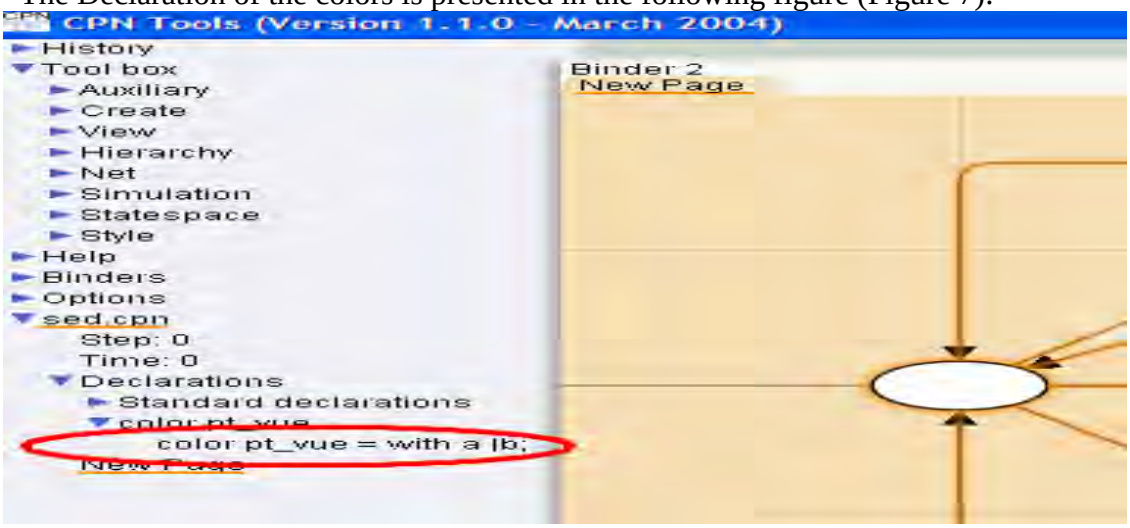

Figure 7 : The Declaration of the colors

The type "pt_vue" must be given to all the places as illustrate in the following figure (Figure 8):

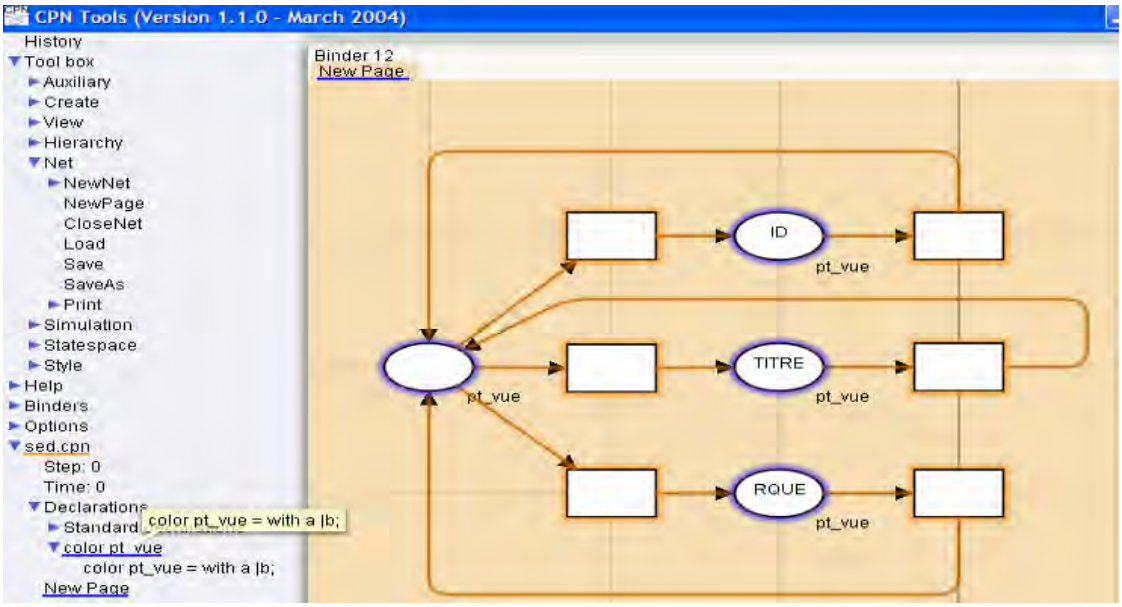

Figure 8 : The assignment of the colors to the places

Of the guards to the transitions to not permit the crossing only for the token witch color are "a" or "b". It requires the declaration of a variable x of type pt_vue that will contain the color of the token in question: x var: pt_vue;

The writing of the guards is therefore: [x=a or else x=b] and is presented in the following figure (Figure 9):

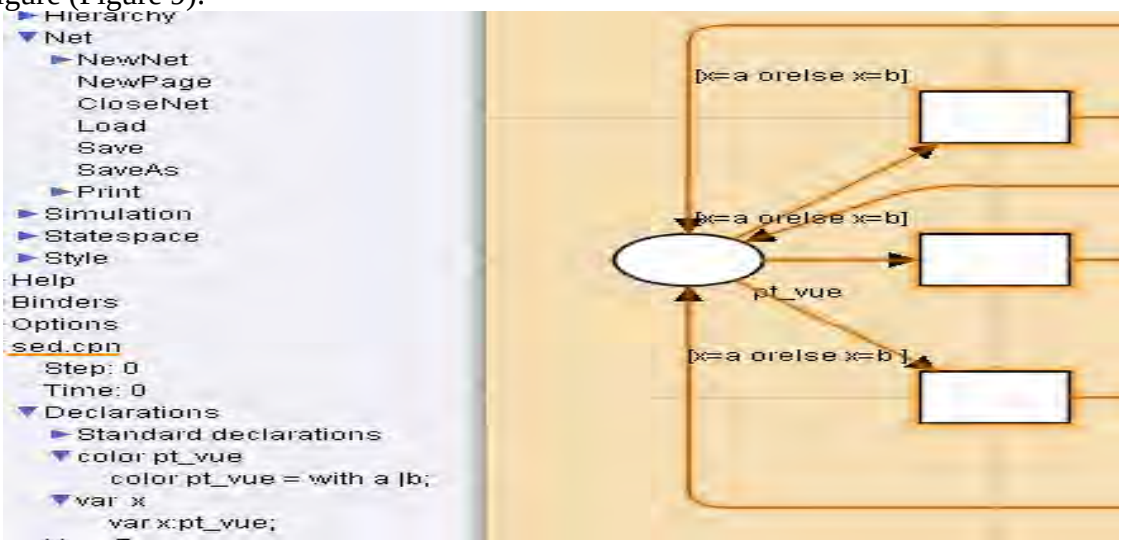

Figure 9 : The enrolment of the guards to the different transitions

It is necessary to think again about to give an initial marking to initial place: 1`a++1`b

To finish, it is necessary to declare the functions to assign to every bow. So the f function whose expression is:

f(a)=a,et f(b)=2b is declared like CPN ML by :

fun f(x)=case of a => 1`a | b=>2`b;

Conclusion

In this article, we tried in a first time to demonstrate the importance of our research while insisting on the place that occupies the paradigm object in the world of the computer development, and this in order to show the necessity of methods that permits to merge the different objects models. For it, we studied in a first time the norm UML and its contribution to the object conception before presenting the filtering mechanism and its role in the development evolutionary system. Then, we adopted the approach that we followed to find a formalism that reflects clearly the constraints of the system and its evolutionary aspect. We used for it the Petri networks while giving the reasons of this choice. Finally, we simulated by the CPN-tools tool the progress of the system in the Petri networks. In perspective of this project, we propose the development of a software layer that acts as interface between CPN-tools and a tool of UML conception such as Rational Rose,ArgoUML,...

The passage between the class diagram and the Petri networks will be made in an automatic manner. The example treated in this paper is a system having two points of view, however we can generalize easily to a important number of points of view while taking as a basis our analogy. We think that this work can be enriched if we can go down on a less level than the points of view such as the views.

We also think about to merge the RdPs associated to the flexible classes of a same system to succeed to the global RdP of the system. The choice of the Petri networks like tool of modelling is a first choice because the initial goal is to succeed to a formal definition of the class multi-point of view, however we consider in our future works to use other formalisms as the Statechartses, the formal languages, and to compare them.

Bibliographie

[Badouel and al 1999] : Badouel.E and Darondeau.P,(1999), Theory of regions. in Lectures on Petri Nets I: Basic Models, pages 529-586, Springer, Lecture Notes in Computer Science, Volume 1491.

[Benzina and al 1997]: Benzina.A and Paludetto.M,(1997), Une méthodologie fonctionnelle SA-RT et réseaux de Petri″, Concepts et Outils pour les systèmes de Production. Cépaduès Editions – Collection Automatique et Production – Coordination J.-C. Hennet, Toulouse.

[Booch and al 2003] : Booch.G, Rumbaugh.J, Jacobson.I, (2003),The UML reference manual, ISBN 0-201-30998-X Addison-Wesley

[Carré and al 1991] : Carré.B And Geib. J.M., (1991), The Point of View notion for Multiple Inheritance. In Proceedings of the ECOOP/OOPSLA.

[Coulette and al 1996] : Coulette.B, Kriouile.A., and Marcaillou, S., (1996) L'approche par points de vue dans le développement orienté objet de systèmes complexes. L'Objet vol. 2, nr. 4, pp. 13-20.

CPN-TOOL : http://www.daimi.au.dk/~kjensen/

[Dano, and al 1997] : Dano.B, Briand.H and Barbier.F, (1997), An Approach Based on the Concept of Use Cases to Produce Dynamic Object-Oriented Specifications. In Proceedings of the Third IEEE International Symposium on Requirements Engineering.

[Ettalbi and al 2002]: Ettalbi.A,Kriouile.A and Sbihi.B,(2002), Petri networks for the modeling of class multi-views; Act of the conference CIMASI 2002 : international Conference on the applied mathematics and the engineer's sciences, Casablanca 2002.

[**Finkelstein and al 1993] :** Finkelstein.A, Gabbay.D, Hunter.A, Kramer. J and Nuseibeh.B., (1993), Inconsistency Handling in Multi-Perspective Specifications. In Proceedings of the ESEC'93, Garmish-Paternkirchen (D), pp. 84-99.

[Hair and al 2004] : Hair.A, Sbihi.B et Belangour.A; (2004),U_VBOOM : une méthode d'analyse et conception unifiée orientée objet basée sur le concept de point de vue ; revue : RIST(Revue d'Information scientifique et technique), volume 14,no 1,2004.

[Kaplan and al 1995] : Kaplan.H, Harrison.W., Katz, A., and Kruskal,.V., (1995) Subject-oriented composition rules. OOPSLA'95, Austin, TX, pp. 235-250.

[Lacazeand al 2003] : Lacaze.X,Navarre.D and Palanque.P, http://liihs.irit.fr/lacaze/papiers/AWPN2003

[Leblanc 2004] : Leblanc.P, Implementation of the UML Testing Profile and Production of Executable Test Cases, (janvier 2004), White Paper Telelogic, www.telelogic.com.

[Meyer 1995] : Meyer.B., (1995), Object success - A managers's guide. Prentice Hall - The Object-Oriented Series.

[Mili and al 1999] : Mili.H, Dargham.J, Mili, A., Cherkaoui.O and Godin.R, (1999), View programming of OO applications. TOOLS, USA.

[OMG] : www.omg.org

[Proth and al 1995]: Proth.J and Xie.X,(1995), Les réseaux de Pétri pour la conception et la gestion des systèmes de production, Masson, 1995.

[Roques et al 2004] : Roques P., Vallée F,(2004), UML 2 en action : De l'analyse des besoins à la conception J2EE, Eyrolles Editions.

[Sbihi and al 2003] Sbihi.B, Kriouile.A, Ettalbi.A, Coulette. B. " Toward a generation of code multi-targets for the VBOOM method : Approche by filtring, " The International Conference on Software Engineering Research and Practice (SERP'03), Las vegas, USA 2003.

[Sbihi 2005] : Sbihi.B; La modélisation par les réseaux de pétri du mécanisme de filtrage UML; WONTIC'05, Workshop sur les Technologies de l'Information et de la communication, Kénitra, Maroc, 24-25 juin 2005,

[Sheng Uei Guan and al 2004]: Sheng Uei Guan, Wei Liu: Modeling Interactive Memex-Like Applications Based On Self-Modifiable Petri Nets. International Journal of Information Technology and Decision Making 3(3): 395-417 (2004).

[UML 2.0] : Unified Modeling Language 2.0, http://www.OMG.org/uml

[U2 Partners] : U2 Partners, UML, version 2.0, http://www.U2-Partners.org