Hybrid Model to Design Proactivity and Multi-Agent-Systems

Jean-Dany Vally and Rémy Courdier
MAS[2] IREMIA
Université de la Réunion
15 Avenue René Cassin - BP 7151, 97715 Saint-Denis messag cedex 9
La Réunion, FRANCE

Abstract: - In this paper, we present a new way to model agent and multi-agents systems (Mas). The modeling we introduce relies on the proposal of a MAS generic model, and a hybrid model of aptitudes supporting the specific agent's properties (autonomy, independence, adaptation). In addition to the formal description of the agent's particular features, the hybrid model offers intrinsic advantages to handle and interrogate the modeled systems. We notice that one of the major obstacles to the design of multi-agents systems lies in the incapacity of some models to express in a generic way the possibilities to exploit the system. This aspect occulted at the modeling time is in part compensated in the existing platforms by the use of static tools (mainly for observation) likely to satisfy the customer's needs. So, to obtain a unified method of multi-agents design we move away from this policy consisting to rely on specific tools to meet the user's needs. We consider that the system's exploitation is part and parcel of the user's needs, which must be coped with the modeling of the system. This allows, on the one hand to obtain a greater flexibility in the system utilization and, on the other hand to save the addition of specific mechanisms in the modeled systems. The hybrid model binding Conceptual Graphs (CGs) and Colored Petri Nets (CP-nets), intrinsically brings a solution to these expectations as well as a solution to the expression of the agent's specificities.

Key-Words: - Multi-agent Simulation, Evolution of Agents, Conceptual Graphs, Colored Petri Nets

1 Introduction

The multi-agent paradigm has been successfully applied to the development of simulation environments. We have proposed a generic simulation platform [15] (GEneric Architecture for Multi-Agent Simulation) designed to support specific considerations of non-linear complex systems as defined in [1]. Our aim is now to extend the scope of applications supported by Geamas by accepting applications that are either reactive or cognitive and to propose methodology principles to guide the process of the system design [5].

The originality and all the difficulty of our process lies in the research for a proposal of a methodology adapted to the complete MAS life cycle. During the design stage, this also implies the consideration of the static and dynamic aspects of entities, as the consideration of agents handling during the process. This approach is the result of the report that it is very difficult to understand what occurs in a MAS made up of hundreds of autonomous and self-adapting agents without integrating, since the design stage, some modeling tools adapted to their observation.

To establish our reflection foundations, we already have powerful formal elements used in object-oriented methodologies [17], in agent-oriented methodologies [6] [3] and in fields like SGBD [2] (for the querying of more complex systems).

In other fields such as linguistic, the semantic structures (Conceptual Graphs [20], …) express knowledge in a logically precise form, humanly understandable, and computationally tractable. The Conceptual Graphs (CGs) model is rich enough to include the main features of the object-oriented model [21] [7] and to encompass new directions in AI. Moreover it provides some well suited forms to support useful querying aspects. But it shows some lacks when we have interest in efficient representation of dynamic process. Colored Petri Nets (CPNs) [9] [11] is a graphical oriented language for design, specification, simulation and verification of dynamic systems. Such model completes in a fashionable way the lacks of the CGs model, and does not require to modify profoundly both models with artificial concepts.

Our work consists in providing formal elements useful for the modeling of multi-agents systems properties. Such elements can be derived from existing formal elements. We think that the conceptual graphs and colored petri nets, which have not been both evaluated for the modeling of multi-agents systems, offer a good base for the design of conceptual modeling tools that support the agent-oriented methodological process that we want to work out Table 1.

Objectives	Models & Languages	Advantages
Proposal for a method describing the different steps of an MAS life cycle	Agent Modeling Technique (AMT bases)	Capabilities to model bad understood or unpredictable complex systems
Expression of the Agent's specifities with the theoretical support CGP-net	CG & CP-net to model the agents' faculties (CGP-net)	Visual and formal modeling of a real world
Implementation support by extension of existing tools	Basic static & dynamic model (CG & CP-net)	Tools to simulate, handle and query a MAS

Table 1: Framework and position

The article is organized in four sections. Within the first one, we present and discuss methodologies and models concerning the agents and multi-agents systems. According to this description, we recall the main concepts of the agent paradigm and justify the choice of CGs and CPNs to express them formally. In the third part, we formally specify some of these characteristics in our generic model. And finally, in the last part we will discuss the advantages and contributions of our proposal. In conclusion, we present future working lines.

2 Agent methodologies and models

After the fashion of object-oriented methodologies [17], some notions repeat: organizations, groups and roles. In the different methodologies these notions are more or less formalized with more or less close semantics.

In the ***Agent-Oriented Analysis and Design*** methodology [3], three models are defined for analyzing an agent system: the Agent, Organizational and Cooperation model. In this methodology, the modeling of specific agent features relies on extensions of object models.

The method ***Agent Modeling Technique for Systems of BDI agents*** [13] defines two main levels: the external viewpoint for the decomposition of the system into agents and their interactions and the internal viewpoint for the Modeling of each BDI agent class with three models (***belief model***, ***goal model*** and ***plan model***).

One of the first methodologies to appear is the ***Cassiopeia*** method [6]. The Cassiopeia method is a way to address a type of problem-solving where collective behaviors are put into operation through a set of agents. The main concepts in Cassiopeia are those of role, agent, dependency and group. An agent is viewed through three levels: Individual, Relational and Organizational role.

The organizational point of view is also present in Aalaadin [8]. The main model for Aalaadin is the ***agent-group-role model***: the agent can handle roles within a group, roles are functionalities or services of an agent and the group structure associated with the roles enables to express several organizational types. It is assumed that an agent is an active, communicating entity which plays roles within a group (roles are identified within a group).

Aalaadin or Cassiopeia describes a first control of collective behaviors by the use of graphs. We undertake to go still further in this way by using graphs formalisms adapted to the modeling of the agent paradigm specificities.

Now we have studied the main features of multi-agents systems, we can concentrate on the ways we have chosen to formalize the more relevant features of the agent paradigm.

3 Merging features of CG and CPN

Actually, agents unlike objects are evolutive entities with their own motivations for acting in the world. An agent evolves with a more or less improved cognition.

This cognition can be modeled with different faculties at different improvement levels. Perception, memory, learning, reasoning, understanding and action are different aspects of the same process of cognition.

These aspects were very studied in the field of linguistic. Thus, we can draw nearer to a model that is very used in this field such as that of conceptual graphs. A conceptual graph represents a mnemic structure generated by the process of perception. It describes a way for assembling percepts.

3.1 The conceptual graphs

Conceptual graphs are a system of logic based on the existential graphs of Charles Sanders Peirce and the semantic networks of IA.

Many popular diagrams can be viewed as special cases of conceptual graphs: types hierarchies, entity-relationship diagrams, dataflow diagrams, state transition diagrams and petri nets [20]. Conceptual graphs embed these notations in a general framework of logic.

No extensions of the theory or the notation are needed to use conceptual graphs as a design language for object-oriented systems [21] [7].

Some tools have been developed to support a precision information retrieval like WebKb [14] which retrieves informations on Web-accessible databases, Notio [19] a Java API for constructing conceptual graphs tools, …

3.2 Coloured Petri Nets

When including a sub-subsection you must use, for its heading, small letters, 11pt, left justified, bold, Times New Roman as here.

Coloured Petri Nets [10] is a modeling language developed for systems in which communication, synchronization and resource sharing play an important role. CP-nets combine the strengths of ordinary Petri nets with the strengths of a high-level programming language.

CP-nets have computer tools supporting their drawing, simulation and formal analysis. Moses [18], Renew [16], … are high-level Petri net simulators that provides a flexible modeling approach. Existing tools for the definition and use of CG and CPN are available and extensible, we can rely on these tools to develop a simulator for Multi-Agents Systems.

3.3 The proposal of an hybrid model

The merging of the two models is done informally in the following way. We directly associate the values of tokens with conceptual graphs. These values will belong to the ***Conceptual Graph*** type. Arcs expressions, possibly referenced by variables, will contain conceptual graphs used to identify valid tokens. These conceptual graphs could be bound to variables to represent them in other arcs expressions. The possible operations inside the arcs expressions are combinations of the canonical formation rules of the CGs. And finally, the guard functions are expressions where a particular relation type called actor could be used.

Proposal 1: A CGP-net is a tuple **CGP**=(**CG**,**P**,**T**,**A**,**N**,**G**,**E**,**I**) where:

(i) **CG** is the ***Conceptual Graph*** type,

(ii) **P** is a finite set of ***places***,

(iii) **T** is a finite set of ***transitions***,

(iv) **A** is a finite set of ***arcs*** such that: $\mathbf{P} \cap \mathbf{T} = \mathbf{P} \cap \mathbf{A} = \mathbf{T} \cap \mathbf{A} = \emptyset$,

(v) **N** is a ***node*** function and is defined from **A** into $\mathbf{P} \times \mathbf{T} \cup \mathbf{T} \times \mathbf{P}$

(vi) **G** is a ***guard*** function defined from **T** into expressions such that: $\forall$ **t** $\in$ **T** [**Type**(**G**(**t**))=**B** & **Type**(**Var**(**G**(**t**))=**Conceptual Graph**],

(vii) **E** is an ***arc expression*** function defined from **A** into expressions such that: $\forall$ **a** $\in$ **A** [**Type**(**E**(**a**))=**Conceptual Graph** & **Type**(**Var**(**E**(**a**))=**Conceptual Graph**],

(viii) **I** is an ***initialization*** function.

.

(ii)+(iii)+(iv) The ***places***, ***transitions*** and ***arcs*** are described by three finite sets that are pairwise disjointed.

(v) The ***node*** function maps each arc to a couple in which the first element is a source node and the second is the destination node. The two nodes must be of different types (i.e., one of them a place and the other a transition).

(vi) The ***guard*** function **G**, maps each transition **t** into a boolean expression where all variables belong to the ***Conceptual Graph*** type.

(vii) The ***arc expression*** function **E** maps each arc to an expression belongs to the ***Conceptual Graph*** type.

The CGP-Net's evolution does not differ from the classic CPNets evolution (***binding***, ***step***, ***occurrence***) [10]. However, we must notice that in classical CP-Nets ***arc expressions*** are identified values or variables but moreover in our model they are structured knowledge queries. It allows on the one hand to do an abstraction (and thus simplification) of several bindings we have interest in and, on the other hand to increase the expression since the CGP-Net itself can be referenced as a concept in ***arc expressions***.

This simple improvement enables us to model the main Mas features in a generic way, indeed it completes the work done with ***Types*** (***Is-a*** relation) by invoking relations between ***Instances*** and ***Types***. ***Is instance of*** is no more the only ***Type***'s relation to manage instances, now with a CGP-Net modeling a ***Type***, we can model ***Instance mutations***. The tokens representing instances are explicit in the ***Type*** definition (a CP-Net model) and they are transmitted between ***Types*** by transitions. Later we call these completions of ***Types***: ***Dynamic Forms***. In this article we do not further develop in this way, we just present the modeling of proactivity for an agent instance. We observe that in this way any strategy could be used to model the instance managing. The CP-net will allow the basic manipulations and the CG the knowledge queries.

We consider that performatives as defined in KQML [4] are part of the message (represented by a token), and we concentrate on the entity interpretation and management (arc expressions and transitions). It enables to deal with heterogeneous entities, some will parse performatives or sender and others will be only sensitive to the contents or even parts of the contents. In this way, knowledge processing depends on the perceived signals as well as the agent's active modes.

4 A new activity design model

We will rely on an intuitive definition, now traditional, of agent before proposing a more practical definition.

Definition 1: An agent is an encapsulated computer system, situated in some environment and able to act in a flexible and autonomous way in order to meet its design objectives [12].

Proposal 2: An agent ***a*** is a polymorphic entity which is part of a whole ***W*** and its lacks ***L*** generate the activity Fig. 1.

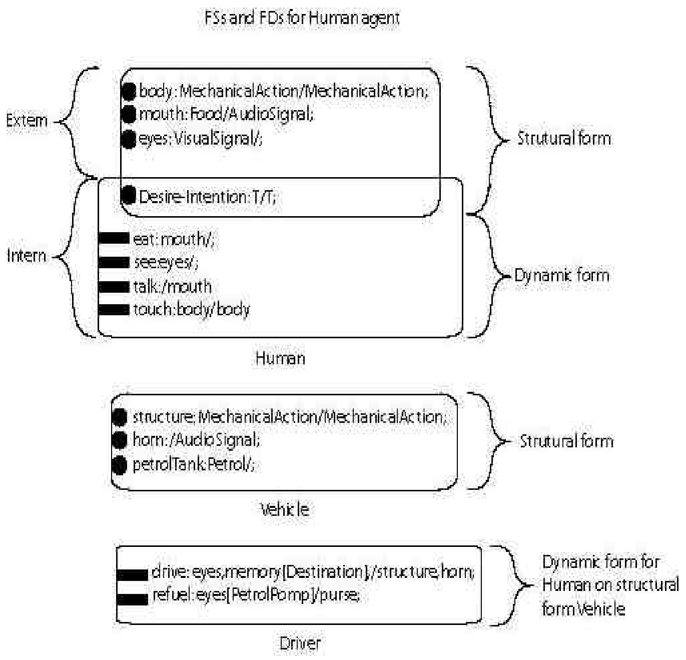

FSs and FDs for Human agent
Extern
Intern
body: MechanicalAction/MechanicalAction;
mouth: Food/AudioSignal;
eyes: VisualSignal/;
Desire-Intention: T/T;
eat: mouth/;
see:eyes/;
talk:/mouth
touch:body/body
Strutural form
Dynamic form
Human
structure: MechanicalAction/MechanicalAction;
horn: /AudioSignal;
petrolTank:Petrol/;
Strutural form
Vehicle
drive: eyes,memory[Destination]/structure,horn;
refuel: eyes[PetrolPomp]/purse;
Dynamic form for Human on structural form Vehicle
Driver

Figure 1: Agent forms

The forms ***F*** of an agent ***a*** are ***structural forms*** or ***dynamic forms***.

The ***structural forms*** **SF**s define different interaction supports for the agent. An **sf** can represent an agent's attribute, so it is indissociable (Example: The physical representation of the agent in the environment) or then, an existential agent's characteristic and so it is separable (Example: An environmental element acquired by the agent and by which it could interact with other entities). **SF**s are used to model mutations (a butterfly agent is defined by several **SF**s, from egg to adult) some properties will evolve and others regress. Links between **SF**s exist if there is a least one **df** which connects them. For example a *Driver* is a **df** between the *Human* **sf** and the *Vehicle* **sf,** Fig. 1. The agent's **SF**s are defined by sets of input and output places for receipt, action, and memorization Fig 1.

The different agent's **DF**s are defined with specific ontological supports (Signals' types hierarchies, …) enabling it to interact with other parts of the environment. A **df** denotes an agent interaction and interpretation mode. There are two main **DF**s types, they square to the logical modalities *necessary* and *possibly*. Before handling them in the next part, just notice that some of the main agent's characteristics are brought by **SF**s.

Independence and relations with the system (W)

The agent's independence with the other parts of the system rises directly. This is due to the fact that the interactions are done by knowledge transfer between **SF**s without direct calls to the behaviors. In counterpart the system, as a whole, must ensure the knowledge transfer between its various parts. **SF**s and **DF**s enable to model these two interaction types without deteriorating the agent's independence.

The agent's interactions as a part of the system are of two types [22]:

The ***vertical interactions*** describing dependences between the system's properties and the part's properties (and vice-versa).

The ***horizontal interactions*** describing constraints among parts, which characterize the system's integrity.

Moreover, the vertical interactions (which induce the representation of the system as a whole) offer the possibility to ``objectify" or ``agentify" the signals emitted by the agents, Fig. 2.

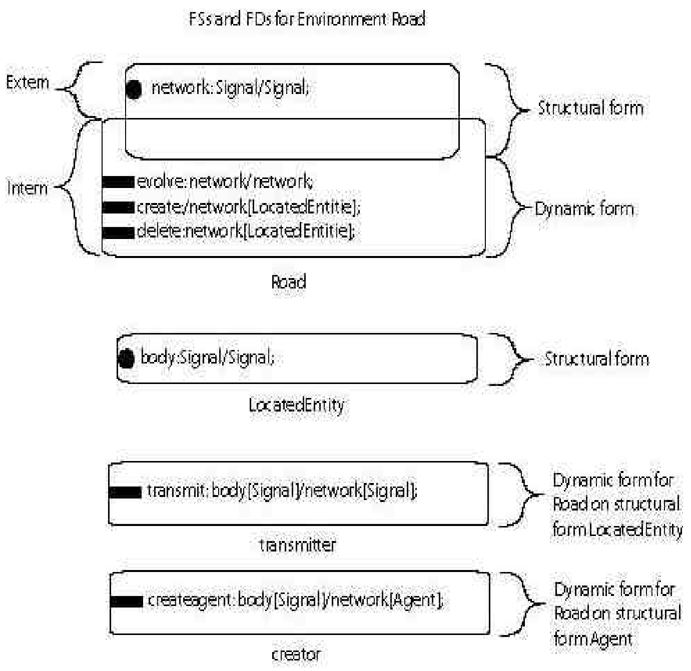

FSs and FDs for Environment Road
Extern
Intern
network: Signal/Signal;
evolve: network/network;
create:/network[LocatedEntitie];
delete:network[LocatedEntitie];
Structural form
Dynamic form
Road
body:Signal/Signal;
Structural form
LocatedEntity
transmit: body[Signal]/network[Signal];
Dynamic form for Road on structural form LocatedEntity
transmitter
createagent: body[Signal]/network[Agent];
Dynamic form for Road on structural form Agent
creator

Figure 2: Environment forms

The dynamic forms describing the whole, or the parts, are represented by a model adapted to the specificities of the agent's paradigm. These specificities intrinsically bring flexibility for the adaptation and evolution of the system. And one of the most significant characteristics that we develop in this article is the proactivity.

5 CGP-nets and proactivity

One relevant feature of the agent paradigm is the **proactivity** concept. CP-Nets brings intrinsically a generic form to model proactivity. The agent's lacks are simply mixed to this model to reduce the intrinsic random phenomenon.

Definition 2: Agents are proactive entities, they do not simply react to their environment, and are able to exhibit goals-directed behaviors to initiate actions. [23]

Proposal 3: A dynamic form ***f*** of an agent ***a*** is an interaction and interpretation mode, it is defined by: Processing ***reactions*** (access, storage, combination) of knowledge generated by the system. These knowledge's processings enable the agent to efficiently react according to the system. Or answers ***actions*** to the absence of information generated by the system. These answers enable the agent to effectively act according to its ***lacks***.

The model used to represent the dynamic forms is the CGP-nets. It brings a smart representation for modeling of **DF**s. The structural components are modelled with places (sources, destinations) and the dynamic components (capacities) with transitions. The ***reactions*** indicate the transitions having classical arcs of the CP-nets as sources. The ***actions*** indicate the transitions having ***proactive arcs*** as sources (functionally corresponding to the ***inhibitor arcs*** but bringing a different semantics to the transition which generate the action). ***Lacks*** correspond to the ***proactive arcs*** expressions.

Example: Initiation of an interaction between an agent and a taxitor due to the lack of food Table 2 & Fig.3.

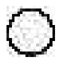
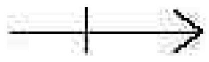
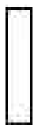
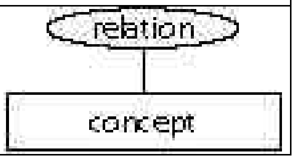
CP-net	Place	Inhibitor arc	Transition	Inscription
Specialization CGP-net	Structural component	Proactivity arc	Action/reaction	Event/Lack
Notation				

Table 2: Specialization of CP-nets concepts

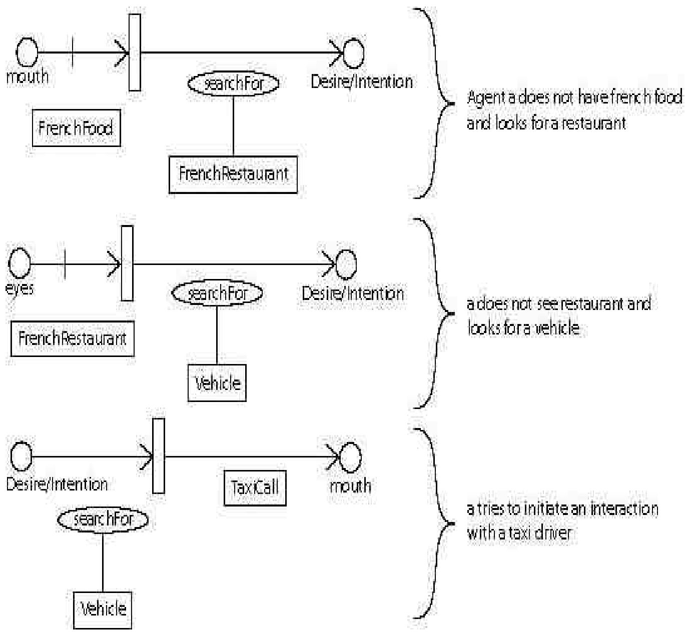

mouth
searchFor
Desire/Intention
FrenchFood
FrenchRestaurant
Agent a does not have french food and looks for a restaurant
eyes
searchFor
Desire/Intention
FrenchRestaurant
Vehicle
a does not see restaurant and looks for a vehicle
Desire/Intention
TaxiCall
mouth
searchFor
Vehicle
a tries to initiate an interaction with a taxi driver

Figure 3: Agent proactivity: behavior activation in answer to the lack of signals

Lacks are modelled with CGs, this allows to model generic to very specialized lacks. When mixed to CP-nets they suppress the need to define several arcs inscriptions to recognize different tokens. The equality between an arc inscription and a token, is no longer simply identity. It is a constructed identity.

6 Interrogation and handling of the system

The various operations we wish to apply on multi-agents systems are described in dynamic structures. These dynamic structures can be defined for the environment or a particular agent depending on the concepts concerned with the query/handling.

For example we can file all foods consumed by an agent in a particular database Fig. 4. Thus, the token containing the Daniel reference will contain also all foods which this one will have consumed.

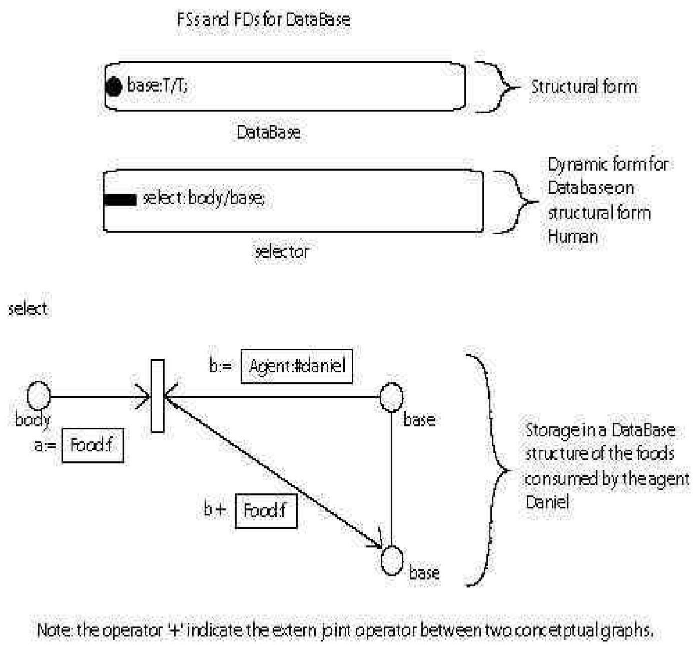

FSs and FDs for DataBase
base:T/T;
Structural form
DataBase
select: body/base;
Dynamic form for Database on structural form Human
selector
select
b:=
Agent:#daniel
body
a:=
Food:f
base
b+
Food:f
base
Storage in a DataBase structure of the foods consumed by the agent Daniel
Note: the operator '+' indicate the extern joint operator between two conceptual graphs.

Figure 4: Request on the agent Daniel

CGs tools make it possible to extract and compose new knowledge for the system whereas CP-nets tools enable to evolve and backup various states of a system. A prototype relying on Notio and Renew is currently implemented.

Some people may think that CG or CP-net are too basic and heavy models for representing Mas system but we do not forget that for now people are trying to define specific relations or structures between types and instances (for representing in an efficient way evolutive entities). Indeed static relation are already well defined between Types, and investigate other relations (like modalities) are not sufficient to determine useful consequences for instances, we have to manage the instances in some less abstracted ways. So a robust instance model with simple modifications can lead us to generic well defined relations. And moreover, in this instance the model intrinsically support simulation and querying aspects which are useful for dealing with a simulated Mas. This model has to be considered as a part of an agent and Mas ontology, like an advanced interaction diagram. It is actually a formal way to interact with an agent system.

7 Conclusion and perspectives

In this paper, we have introduced a new way of modeling agent and multi-agents systems. This model allows to decompose the various facets of an agent and a system in structural forms and dynamic forms, supporting the two interactions types between these two entities. The use of the conceptual graphs and coloured petri nets for the representation of SFs and DFs brings several advantages to express adaptation, learning and interrogation in a multi-agents system. Moreover one of the main agent's characteristic: the proactivity, is modelled in a simple and intuitive way. Finally, the prospects of this work are interesting for simulation thanks to multi-agents systems since it lies on a reliable basis for which a great deal of work have been already done concerning the formulation of requests. This opening with regard to the possibility for expressing requests is very significant in the field of multi-agent simulation (it is an aspect which we propose to develop in our future work while basing it on the existing studies binding Conceptual Graphs and Databases).

References:

[1] T.R.J Bossomaier and D.G Green, Complex systems, *Cambridge University Press*,1996.

[2] Mokrane Bouzeghoub and Elisabeth Métais, Semantic Modeling of Object Oriented Databases, *17th International Conference on Very Large Data Bases, September 3-6, 1991, Barcelona, Catalonia, Spain, Proceedings*, , 1991, pp. 3-14.

[3] B. Burmeister, Models and methodology for agent-oriented analysis and design, *Working Notes of the KI'96 Worshop on Agent-Oriented Programming and Distributed Systems*, 1996.

[4] Philip R. Cohen and Hector J. Levesque, Communicative Actions for Artificial Agents, *Proceedings of the First International Conference on Multi--Agent Systems*, 1995, pp. 65--72.

[5] R. Courdier and P. Marcenac, Un processus de développement en spirale pour la simulation multi-agents, *L'Objet*, Vol.4, No.1, 1998, pp. 73--86.

[6] A. Collinot and L. Ploix and A. Drogoul, Application de la méthode Cassiopée à l'organisation d'une équipe de robots, *LAFORIA/IBP/CNRS, Paris VI*, 1996.

[7] G. Ellis, Object-Oriented Conceptual Graphs, *Proceedings of the Third International Conference on Conceptual Structures, (ICCS'95)*, 1995, pp. 144--157.

[8] J. Ferber and O. Gutknecht, A Meta-Model for the Analysis and Design of Organizations in Multi-Agents Systems, *Proc. of the International Conference of Multi-Agents Systems {ICMAS'98}*, Vol.X, No.X, 1998, pp. 128--135.

[9] Kurt Jensen, Coloured Petri Nets -- Basic Concepts, Analysis Methods and Practical Use, Vol. 1: Basic Concepts., *EATCS Monographs in Theoretical Computer Science*, 1992, pp. 1--234.

[10] Kurt Jensen, An introduction to the theoretical aspects of coloured Petri nets., *Lecture Notes in*

Computer Science; A Decade of Concurrency, Vol. 803, 1993, pp. 230--272.

[11] Kurt Jensen, An Introduction to the Practical Use of Coloured Petri Nets., *Lecture Notes in Computer Science: Lectures on Petri Nets II: Applications*, Vol. 1492, 1998.

[12] Nicholas R. Jennings, On agent-based software engineering, *Artificial Intelligence*, Vol.177, No.2, 2000, pp. 277--296.

[13] D. Kinny and M. Georgeff and A. Rao, A methodology and modelling technique for systems of BDI agents, *Agents Breaking Away: Proccedings of MAMAW'96*, Vol.1083,1996.

[14] P. Martin and P. Eklund, Embedding Knowledge in Web Documents, *Griffith University School of Information Technologie*, 1998.

[15] P. Marcenac and S. Giroux, GEAMAS: A Generic Architecture for Agent-Oriented Simulations of Complex Processes, *International Journal of Applied Intelligence*, 1998.

[16] O. Kummer, F. Wienberg, M. Duvigneau, Renew - User Guide, *Technical report University of Hamburg*, 2000.

[17] J. Rumbaugh and M. Blaha and W. Premerlani and F. Eddy and W. Lorensen, Object-Oriented Modeling and Design, *Prentice Hall*, 1991.

[18] R. Esser, J.W. Janneck and M. Naedele, Applying an Object-Oriented Petri Net Language to Heterogeneous Systems Design, *In Proceedings of Petri Nets in System Engineering*, 1997.

[19] F. Southey and J.G. Linders, Notio - A Java API for developing CG tools, *International Conference on Conceptual Structures (ICCS'99)*, 1999.

[20] J.F. Sowa, Conceptual Structures - Information Processing in Mind and Machine, *Addison Wesley*, 1984.

[21] J.F Sowa, Logical foundations for representing object-oriented systems, *Journal of Experimental and Theoretical Artificial Intelligence (JETAI)*, 1993, pp. 237--261.

[22] A. Varzi, Parts, Wholes, and Part-Whole Relations: The Prospects of Mereotopology, *Data and Knowledge Engineering 20*, 1996, pp. 259--86.

[23] M. J. Wooldridge and N. R. Jennings, Intelligent agents: Theory and practice, *Knowledge Engineering Review*, 1995.