International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, New York, USA
Volume 10 – No.7, March 2016 – www.ijais.org

Application of the Interaction-Oriented Approach in Mobile Systems

A. Sayouti
Team Architecture of Systems
LISER - Laboratory
ENSEM, Hassan II University
BP 8118, Oasis, Casablanca, Morocco

W. Bouab Bennani, N. Belghini
Team TIC & Team Mobile Systems
LSI Laboratory, ESTEM Research Center,
Casablanca, Morocco

ABSTRACT

Internet is not only an information highway, but also a mean to remotely control mechanical systems, such as mobile systems. The Internet limits the quantity of information which can be transmitted and introduces delays which can make the remote control difficult or impossible. The solutions, proposed to face the limitations of the communication channel, are founded on the autonomy and the intelligence based on multi-agents systems granted to the mobile system in order to interact with its environment and to collaborate with the remote user.

Interactions play an important role for the coordination and the cooperation of agents in a multi-agents system. In this context, the expression and the management of these interactions constitute significant stakes. In this paper, we propose an abstraction of these interactions where an interaction is regarded as a shared resource separated from the internal architecture of agents and easily reused by them. In a first part, we describe and compare the interaction-oriented approach to the classic one. Then a list of properties is given to show the benefit to consider an interaction, not only as a communication means, but also as an entity shared by all agents and not reserved to some of them. An illustration of our approach is given in a remote control application of two mobile systems using VRML.

Keywords

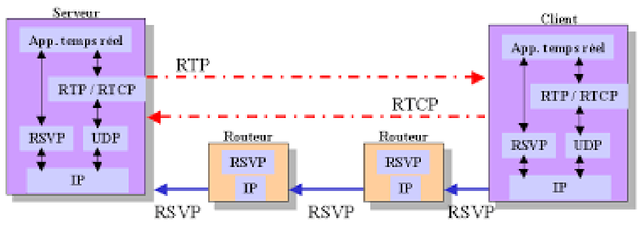
Mobile System, TIC, Multi-Agents Systems, Distributed Systems, Interactions, Web Application, Virtual Reality.

1. INTRODUCTION

The World Wide Web (WWW) is one of the major components of the Internet and has promoted dramatic change in our living and working styles. The WWW eliminates traditional communication barriers such as long-distance, time constraints, and sometimes, language differences (i.e. if a graphical communication system is used). Therefore, the WWW provides us with a new working environment where people living in different places can work together collaboratively and efficiently given proper design of the environment. These new technologies can be exploited not only for distributed communication but distributed command, control applications [1] and simulation as well.

The remote control extends the sensorimotor capacities and the possibilities of actions of a human being in a distant place. The Internet network (network without quality of service) limits the quantity of information that can be transmitted (bandwidth) and introduces delays which can make the remote control difficult or impossible.

Many studies tend to increase the QoS of the Internet network using different techniques. Include the definition of RTP and RTCP protocols, bandwidth reservation opportunities with RSVP or simplification of routing algorithms using DiffServ and MPLS for example.



Our work wanting to place in the Internet at large, we do not consider these techniques. The solution proposed, through this work, advocate removal of human operators from the feedback control loop, and equip the mobile systems with a high degree of local intelligence in order to handle the uncertainty in the real world and also the arbitrary network delay. The need that consists in wanting to assign to the mobile system the maximum of autonomy and intelligence brought us to examine in the detail the choice of remote system architecture [2].

The organization of a system - or its control architecture - determines its capacities to achieve autonomous tasks and to react to events. The control architecture of an autonomous mobile system must have both decision-making and reactive capabilities: situations must be anticipated and the adequate actions decided by the mobile system accordingly, tasks must be instantiated and refined at execution time according to the actual context, and the mobile system must react in a timely fashion to events.

To meet this global requirement, the control architecture should have the following properties [3]: Autonomy, Reactivity, Robustness and Extensibility.

We note an interesting link between the desirable properties of intelligent control architecture for autonomous mobile systems and the behavior of an agent:

- Autonomy: An agent has its own internal thread of execution, typically oriented to the achievement of a specific task, and it decides for itself what actions it should perform at what time.
- Situatedness: Agents perform their actions while situated in a particular environment. The environment may be a computational one (e.g., a Web site) or a physical one

(e.g., a manufacturing pipeline), and an agent can sense and effect some portions it.

- Proactivity: In order to accomplish its design objectives in a dynamic and unpredictable environment the agent may need to act to ensure that its set goals are achieved and that new goals are opportunistically pursued whenever appropriate.
- Decentralization: there is no designated controlling agent (or the system is effectively reduced to a monolithic system).

A multi-agent system (MAS) is a system composed of multiple interacting intelligent agents [4]. Multi-agent systems can be used to solve problems which are difficult or impossible for an individual agent or monolithic system to solve.

Interactions between agents within a MAS are largely recognized like one of the essential mechanisms to ensure collaboration by the distribution of tasks, the resource sharing, the coordination of actions and the resolution of conflicts. It is thus interesting to be able to manage these interactions through all stages of MAS design and execution [5]. In the design phase, the definition of the communication semantic is carried out on a high level of abstraction. However, in the implementation, this semantic is described in the code of the agents. This coupling between the interactions and the code of the agents makes the interactions lose their statute of conceptual entities and significantly complicates the maintenance, the openness and the re use of MAS components. Hence, we proposed an approach based on software interactions offering an abstraction and a better management of the interactions between agents in relation to software components domain and particularly the work presented in [6] on the dynamic adaptation of the component based applications.

This paper is presented as follows: in section 2, we describe the usual agent's model and we show its limits in term of interaction management. In section 3, we present the interaction oriented approach. Then we will present a remote control simulation of two mobile systems using VRML as an illustrative example. Finally, some conclusions are presented in section 5.

2. THE CLASSIC APPROACH

Although many models of agents were proposed [7], an agent is mainly made up of:

- Roles: what an agent must do.

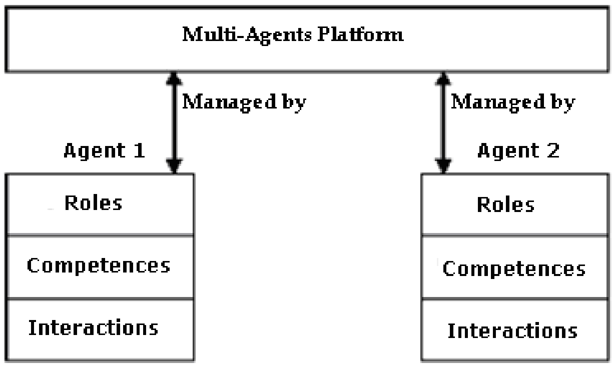


Fig. 1. Agent's Internal Architecture

- Competences: they can be internal relating to the way in which the agent ensures its roles, or external (social) concerning its relationship with others to perform its task.
- Interactions: they allow the agent to communicate with its environment and/or with the other agents.

The internal architecture of an agent is illustrated in Figure 1. One of the principal properties of the agent in a multi-agents system is its capacity to be in interaction with the others. These interactions are generally defined like any form of action carried out within a society of agents which modifies the behavior of another agent. They give to the agents the possibility of taking part in the satisfaction of a global goal.

This participation allows the system to evolve to one of its objectives and to have an intelligent behavior. These interactions are mainly based on the communication. A communication can be defined as a local action of an agent towards other agents. The questions covered by a communication model can be summarized by the following interrogation: who communicates what, to which, when, why and how? Several works were interested to answer this interrogation. These works can be classified in two groups: works on the inter-agents communication languages [8][9][10] and works on the interaction protocols [11][12][13].

The most widespread interactions languages are KQML and FIPA-ACL standardized by the FIPA (Foundation for Intelligent Physical Agents). KQML is based on the work of Searle on the Speech Acts. In KQML, the messages are independent of the transport mechanism (TCP/IP, SMTP, etc.) and are conceptually composed of three layers as shown in Figure 2.

The content layer is generally composed of an expression in a knowledge representation language like KIF, Interlingua, PROLOG, etc. The message layer adds necessary information to the interpretation of the content: the language, the ontology and the speech act associated to the message. The communication layer contains low level information relating to the identification of the sender and the receivers as well as transport layer parameters.

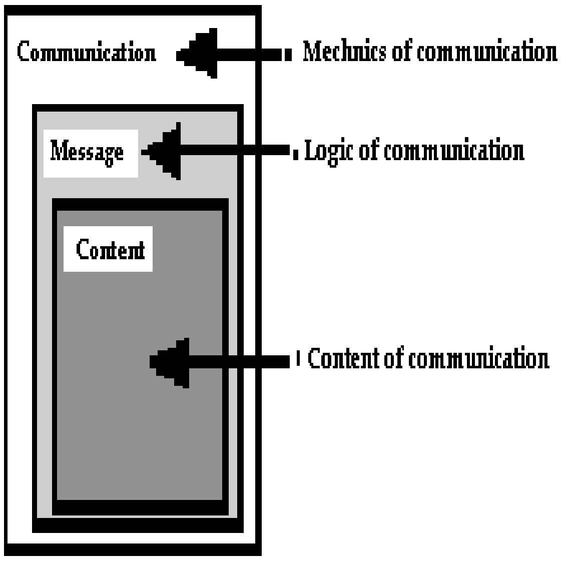


Fig. 2. The three encapsulation levels of the KQML language

Like KQML, FIPA-ACL is based on the speech acts theory. The semantics of FIPA-ACL is defined by a formal language called SL (Semantic Language). However, KQML and FIPA-ACL describe only the message structure, whereas the definition of complex interactions requires the establishment of elaborate protocols. The purpose of several works was to specify interaction protocols which varied according to the context: coordination of agents' actions, negotiation, auctions, etc. Thereafter, theses protocols were standardized by FIPA. Other works were focused on the formalization of protocols like, COOL which propose to model a conversation using finite state automata. The same principle was employed in AgenTalk which add the possibility to create easily new protocols by the specialization of existing ones.

3. THE INTERACTION-ORIENTED APPROACH

One idea which has been proved reliable in the software components field is the separation between the interactions and the components [14]. This manner to define the interactions opened the way to new possibilities of abstraction and expression of the interactions. Thus, an interaction is not specific any more to only one component but can be reified into an entity as well as the other components of the system. This entity is viewed as a shared resource that the components can consult, use or instanciate. This approach separates the interaction-related behaviors and functionalities from the algorithmic parts of the agents (see Fig. 3). From now on, interactions will be managed by an entity called "Interaction Manager".

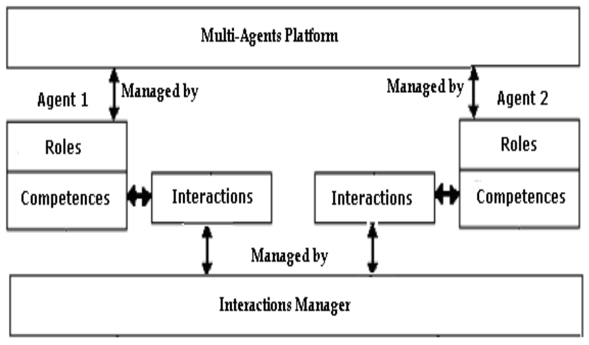


Fig. 3. Agent's New Architecture

Interactions are the basic elements of our approach with the following properties [15] (see Fig. 4):

- The interactions are defined in a formal and declarative way independently of the communicating agents. This formal description will be represented as classes in the applicative language called "interaction patterns".
- An interaction pattern is an abstraction of the interaction concept. It is defined on the agent class level and can be instanciated. It is the counterpart to classes in object oriented languages whereas interactions instances are the counterpart to objects. In other words, the interactions are objects created by instanciating the interaction patterns.
- An interaction pattern is implementation-independent and an interaction instance can connect heterogeneous agents across different platforms. For instance, an interaction can connect a JADE agent with a MadKit agent.
- The instanciation of an interaction pattern is independent of the instanciation of the interacting agents' classes. An interaction can be dynamically created between agents that need to interact and, it is destroyed when they don't need to interact any more.

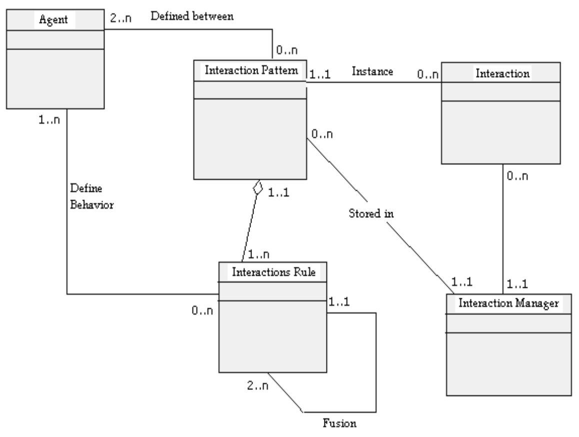


Fig. 4. Class Diagram of the Interaction-oriented Approach

3.1 MAS Architecture

Our approach proposes a new MAS architecture (see Fig. 5) and a new development process. Indeed, in the conventional MAS, the agent is developed in the same time as its interactions. So, the developer has to foresee in advance all the possible interactions of the agent. Also, any change of the interactions implies the update of the whole code of the agents, what is not an easy task. However, in our approach, the development process is seen as follows [16]:

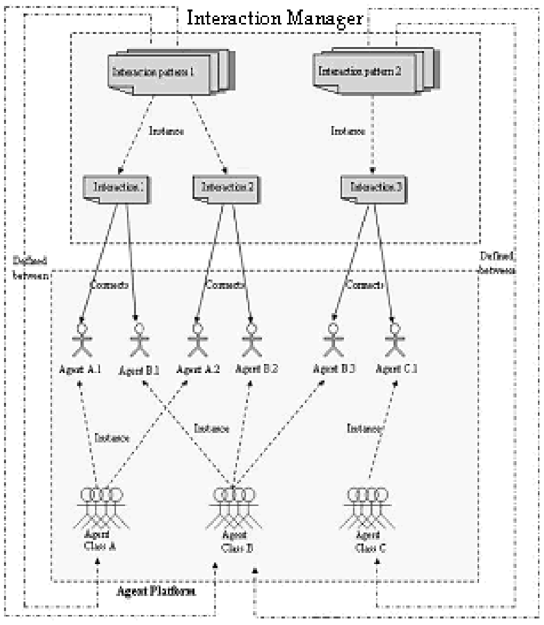


Fig. 5. MAS architecture

- In the first step of the development we are interested in the construction of the agents' classes and the description of their skills without worrying about interactions between them. We describe only what the agent can do without worrying about "with whom", "when" and "how" it communicates.

- In the second step we define the interaction patterns which allow the agents' classes to interact. These interaction patterns will be then registered in the interaction manager.
- At the deployment time, we instanciate the interaction patterns to create interactions between agents.

3.2 Interactions-oriented approach Advantages

The advantages of the interaction-oriented approach rise from two main ideas. The first is the separation of concerns which dissociates different aspects to take into account their specificities and to articulate them without confusing them. The second is the interaction abstraction. Thus, the advantages of our approach can be summarized by the following points:

- Quality of the design: as they are considered as first-class entities, the interactions may be designed, proved to be correct and validated, adapted and re-used without worrying about agents. The agents are relieved of coordination tasks; the design of an agent can focus on its computational functions, and it can be adapted, maintained or re-used without impact on the protocols it should use.
- Dynamic adaptability: in certain cases, it is essential that the application could be able to adapt itself dynamically to various contexts of execution (addition and modification of interactions at runtime). In that case, the developer does not any more need to define new interactions but, it creates them (by instanciation) from interaction patterns.
- Traceability between the design and the implementation: several works concentrated on the representation of the interactions at a conceptual level (Petri Nets, A-UML, etc), but in all these works, the link between the representation of the interactions during the specification and their representation during the execution is broken. However, the approach we have just introduced allows finding this traceability between the design and the implementation thanks to the reification of the interactions in separate entities.
- Refinement of the interactions: expressing the interactions as first-class entities favours the concept of inheritance between these entities. This concept of inheritance opens the way to the refinement of the interactions and to the construction of complex interactions from elementary ones.
- Manipulating interactions: from now on, it is possible to express interactions in which certain participants (or all the participants) are interactions. Thanks to such interactions, it is possible to define, in a simple way and with a strong expressive power, properties between interactions, such as the mutual exclusion (two interactions are not active at the same time in the system, etc.). These interactions can be considered as tools of manipulation of interactions.

4. APPLICATION OF THE ORIENTED - INTERACTION APPROACH

To illustrate and validate the interaction-oriented approach we developed an application of road traffic simulation as virtual environment for training. The user, through his Web navigator (see Fig. 6), addresses a request to a Web server (step1) and downloads on his work station an application developed in VRML (step2).

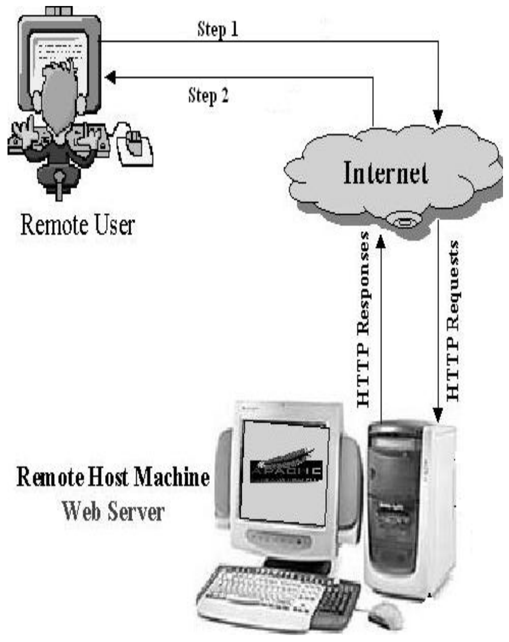


Fig. 6. General architecture

4.1 Virtual Worlds using VRML

VRML's main advantage is that it is currently the de facto standard for web based 3D visualizations. VRML [17] allows for easy definition of geometric shapes in hierarchical groupings and it also provides many advanced 3D graphics functions such as lighting models and surface materials. VRML allows for simple interactions between a user of a virtual world and the various objects within the world, such as clicking on an object to activate an object's script. VRML is currently well supported with various end-user browser and modelling programs, simplifying the task of both creating and viewing virtual worlds.

4.2 Application Description

The goal of this application [18] is to control the road traffic, to help in the decision-making according to the results of the simulation, to insure an easy flow of the traffic by anticipating the situations of congestion and blocking, to find solutions for complex situations of the road traffic, to have an explanatory role of the reasons of certain situations as the accidents and to make statistics about the critic periods and zones to find potential solutions. This web application can be used by the ministry of transport as virtual environment for training in different situations.

Let's take a simple example of a road traffic situation (see Fig. 7) like the passage of the train. This simulation brings in the following agent classes:

- The Vehicle agent: this agent tries to overrun the environment from a departure position by avoiding the conflicts (collisions, automatic barrier). This agent has to plan its road, to observe its environment and interact with it as well as with the other agents.
- The Environment agent: this agent allows a graphic representation of the simulation.

- The Automatic_Barrier agent: it manages a given barrier.

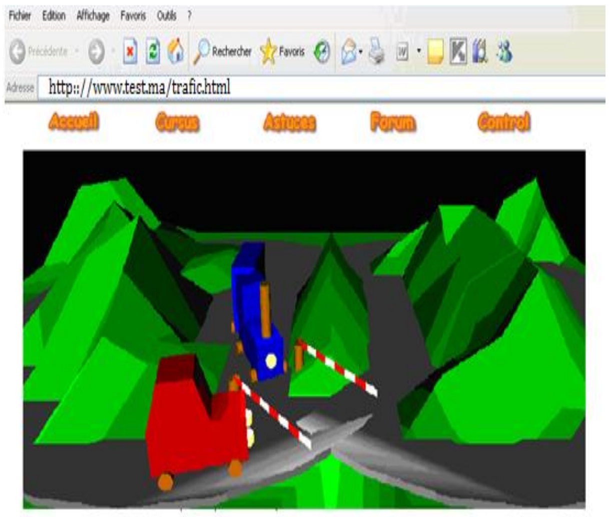

Fichier Edition Affichage Favoris Outils ?
Rechercher
Favoris
Adresse
http:://www.test.ma/trafic.html
Accueil
Cursus
Astuces
Forum
Control

Fig. 7. Web interface

This situation brings in two agent vehicles “vehicle1” and “vehicle2” approaching the automatic barrier. From this situation we can extract the interaction named “meet_automatic_barrier”.

Interaction **meet_automatic_barrier** (Vehicle vehicle, Automatic_Barrier barrier)

```
{
vehicle.move( ) →
if (barrier.getstate ( ).equals ("closed")) then
vehicle.stop ( )
else
vehicle._call
end if
barrier.changestate( )→
if (barriere.getstate( ).equals ("opened") && vehicle.stop (
)) then
Vehicle.move ( )
End  if
}
```

This interaction pattern connects two objects: vehicle and automatic barrier, and describes interactions rules which express the controls that must be operated on the connected agents. The interaction pattern “meet_automatic_barrier” describes the behavior of a vehicle approaching the automatic barrier and indicates that if the vehicle is moving towards the automatic barrier which is closed, the vehicle has to stop; and if the automatic barrier opens, the vehicle can go ahead.

5. CONCLUSION

In this paper, we began by emphasizing the limits of the conventional agent approach in term of interaction development. We proposed an approach which separates the interaction-related behaviors and functionalities from the algorithmic parts of the agents. An application of road traffic simulation served as experimental base which allowed us to validate our approach.

As perspectives, we are studying the various possibilities offered by the merging mechanism to define more complex behaviors and to integrate them easily.

6. REFERENCES

[1] Le parc, P., Ogor, P., Vareille, J., Marce, L., Web based remote control of the mechanical systems, Proceedings of the IEEE International Conference on Software Telecommunications and Computer Networks, Split, Croatie (Year of Publication: 2002).

[2] A. Sayouti, Conception et Réalisation d’une Architecture de Contrôle à Distance Via Internet à Base des Systèmes Multi-Agents. Phd. Thesis, ENSEM, Hassan II University 2009.

[3] A. Sayouti, F. Qrichi Aniba, H. Medromi, Remote Control Architecture over Internet Based on Multi agents Systems. International Review on Computers and Software (I.RE.CO.S), Vol 3, n. 6, pp. 666 – 671, November 2008.

[4] J. Ferber, Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence, Addison Wesley Longman, Harlow, UK, 1999.

[5] Sayouti, A., Qrichi Aniba, F., Medromi, H., Lakhouili., A., Applying the Interaction-Oriented Approach to Remote Control of Robotic Systems. Proceedings of the Conferência Ibérica de Sistemas e Tecnologias de Informação (CISTI). Porto. Portugal. 2007.

[6] Blay, M., Ensellem, D., Occello, A., Pinna, A.M., Riveill, M., Fierstone, J., Nano, O., Chabert, G., Un service d’interactions : principes et implémentation. Proceedings of Journées Composants, Grenoble, France, Octobre 2002.

[7] Y. Secq. RIO: Rôles, Interactions et Organisations, une méthodologie pour les systèmes multi-agents ouverts. PhD. Thesis, University of Sciences and Technologies of Lille, France, 2003.

[8] Finin, T., Fritzson, R., McKay, D., McEntire, R., KQML as an Agent Communication Language. Proceedings of the 3rd International Conference on Information and Knowledge Management CIKM'94, 1994.

[9] Foundation for Intelligent Physical Agents. Agent Communication Language. FIPA 97 Specification, Part 2, 1997.

[10] P. Noriega, C. Sierra. Auctions and multiagent systems. Intelligent Information Agents. Springer, p. 153-175, 1999.

[11] Foundation for Intelligent Physical Agents, Agent Communication Language. FIPA 99 Specification Draft, 1999.

[12] Barbuceanu, M., Fox, M.S., COOL : A Language for describing coordination in multi-agent systems. Proceedings of the ICMAS’95, 1995.

[13] Kuwabara, K., Ishida, T., Osato, N., Agentalk: describing multiagent coordination protocols with inheritance.

Proceedings of the 7th Int. Conf. On Tools for Artificial Intelligence (ICTAI-95), 1995.

[14] Khalfaoui, S., Lejouad Chaari, W., Pinna Dery, A.M., Interactions entre composants pour environnements Multi-Agents. Proceedings of the Journée Multi-agents et Composants (JMAC'2004), Paris, France, November 2004.

[15] Sayouti, A., Qrichi Aniba, F., Medromi, H., Tele-robotic over Internet Based on Multi-agents System. Proceedings of the International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES'2008), Dalian, China, July 27-31, 2008.

[16] Sayouti, A., Qrichi Aniba, F., Medromi, H., Interactions between agents as shared resources in multi-agents systems. Proceedings of the Second IEEE International Conference on New Technologies, Mobility and Security Program (NTMS'2008), Tanger, Morocco, November 5-7, 2008.

[17] VRML Consortium, ISO/IEC 14772-1 1997 The Virtual Reality Modeling Language (VRML97), http://www.vrml.org, 1997.

[18] Sayouti, A., Medromi, H. "Book Title: Les Systèmes Multi-Agents : Application au Contrôle sur Internet." Academic Publishing in Europe, August 2012.