Modeling Languages for Multiagent Systems: A case study using AUML and MAS-ML

Juliana P. Campos[1], Alcione de Paiva Oliveira[1], José Luís Braga[1],
[1]Departamento de Informática – Universidade Federal de Viçosa (UFV),
Avenida Peter Henry Rolfs, s/n – 36570-000 – Viçosa - MG - Brasil
{jucampos,alcione,zeluis}@dpi.ufv.br

Abstract. Agent Unified Modeling Language (AUML) and Multi-Agent System Modeling Language (MAS-ML) are modeling languages that extend Unified Modeling Language (UML) for modeling Multiagent Systems (MAS). This paper presents a comparison between these two languages achieved through a case study: a MAS that schedules lectures in conferences. The aim of this case study was to exemplify the usage of these languages, identify difficulties in modeling and carry out a comparison between the two extensions.

Keywords: AUML; MAS-ML; Multiagent Systems; modeling comparison.

(Received March 04, 2010 / Accepted July 02, 2010)

1 Introduction

During the process of software development, models are built with the goal of representing the relevant aspects of the system to be developed. The models should be simple and specify the system's desired characteristics, facilitating the communication of design decisions to all persons involved in the process. The most used modeling language today is UML (Unified Modeling Language).

The development of agent-based systems are increasingly being used as a paradigm for design and development of complex systems [19]. This type of approach is particularly suited to situations that require distributed decision making with multiple competing interests. To model such a system, the representation languages must have some special features. The use of an object-oriented (OO) modeling language to model multiagent systems (MAS) is controversial, since agents have characteristics that lead to the need to develop an appropriate language. Some authors believe it is possible to model a MAS using an OO approach, but this approach does not handle some modeling issues involved in MAS [2, 16,17]. [4] demonstrate that the UML 2.0 brought many improvements to agent modeling, but there are still some untouched aspects. As mentioned [4] (p. 120): "UML has no "off-the-shelf" constructs to express: goals, agent, groups, multi-casting, generative functions such as cloning, birthing, reproduction, parasitism and symbiosis, emergent phenomena, and many other nature-based constructs that are helpful for representing agent structures." To overcome these limitations, several modeling languages have been proposed for MAS using UML as their base [8, 11, 12, 16,17,18].

This article aims to exemplify the use of the AUML and MAS-ML modeling languages, seeking to identify problems and compare the two languages. The text is structured as follows: sections 2 and 3 describe the AUML and MAS-ML languages. The section 4 presents the case study. In section 5, a comparison is carried out between the languages and, in section 6, the conclusions are presented.

2 AUML

AUML (Agent Unified Modeling Language) [1] is a modeling language based on UML, which resulted from the cooperation between the FIPA (Foundation for Intelligent Physical Agents) [7] and OMG [13], with the intent to model aspects related to the agent approach. AUML proposes to use and extend the UML diagrams to describe the requirements for modeling a MAS. The class, sequence, collaboration, activity and state diagrams are the ones extended by the AUML. They are all very similar to the UML diagrams, but they

emphasize the behavior of agents and the interactions between them.

The class diagrams were extended to represent agents, but some authors disagree about the description of the classes and relations [3, 9]. The main difference is that one of them proposes to insert only the name of the agent interaction protocols in the class diagram [9], while the other proposes the insertion of the protocols along with their messages of input, actions and output [3]. The present work will consider the Huget proposal [9], because of its simple approach and easier reading. Besides, the protocol is presented in a specific diagram. In addition to the compartments for attributes and operations, such as those in the UML class diagram, in the AUML class diagram, it is possible to include compartments for: roles represented by agents; name of the state diagrams that represent the internal processing of the agents; services provided; capacities, perceptions and agent interaction protocols; and organizations in which the agent participates. The sequence diagrams were extended to represent agent interaction protocols – AIP [11,12]. AIP is a communication standard which describes the sequence of messages exchanged between the agents and the restrictions related to the content of these messages.

The sequence and collaboration diagrams are semantically equivalent. The first one emphasizes the chronological sequence of communication, while the second emphasizes the association among the agents, but the main difference is the possibility to express constraints of time in the sequence diagrams, which is very useful in the MAS. In the collaboration diagram, the sequence of the exchanged messages is represented by numbering. The main change in this AUML diagram is that the same agent can appear representing multiples roles.

The AUML activity diagram is used to represent the flow of activities, similarly to the UML. However, it represents the activities associated with an AIP or to the activities of a role represented by an agent. The representation of the execution lines of control flows in an explicit manner is particularly useful for complex interaction protocols that involve concurrent processing [12].

The AUML state diagram represents the states and the transitions of an AIP or a role. It is used to represent known constraints and indicates which agent executes the action that causes the change from one state to another.

3 MAS-ML (Multi-Agent System Modeling Language)

MAS-ML (Multi-Agent System Modeling Language) [15,16,17] is a modeling language which extends the UML metamodel, describing new metaclasses and stereotypes to represent MAS elements and their properties. It extends the UML class and sequence diagrams and proposes two new diagrams: organization and role diagrams [16]. MAS-ML is particularly adequate to model cognitive agents, according to the BDI approach (Belief Desire and Intentions) [14], since it allows the modeler to express the beliefs, desires and intentions related to the objects and organizations.

MAS-ML is a UML extension based on the conceptual framework (metamodel) Taming Agents and Objects (TAO). This framework defines static and dynamic aspects of the MAS, clustering abstractions frequently described. The static aspects capture elements of the system, their properties and relations. The elements defined in TAO are [15]: *objects* – passive or reactive element which has state and behavior, but does not have control over it; *agents* – autonomous, adaptive and interactive element; *organization* – groups agents that execute roles and have common goals; *environments* – where objects, agents and organizations reside; and *role* – guides and limits the behavior of an agent or object in the organization. The relations that link these elements are:

- *Inhabit*: an element resides in a habitat and can leave and enter it respecting the permissions;
- *Ownership*: an element is defined in the scope of another element;
- *Play*: an element plays a role;
- *Dependency*: an element may depend on another to perform its work;
- *Specialization/Generalization*: the sub-element that specializes the super element inherits all the properties and relations defined in the super element;
- *Association*: an element can interact with another element to which it is associated;
- *Aggregation*: an element can be an aggregator of parts;
- *Control*: a controlled element must perform everything that the controlling element requests.

The extended class diagram and the role and organization diagrams proposed in the MAS-ML describe static aspects. The class diagram was extended to represent structural relations among agents, agents and classes, organizations, organizations and classes, environments and environments and classes. The main change proposed in this diagram was the use of other relations among the classes, besides those defined for the UML class diagram. The relations *association*, *aggregation*, *specialization* and *inhabit* can be used.

The role diagram is responsible for modeling the roles defined in the organizations. It defines the relations between agent roles and object roles and between these roles and classes. The *control*, *dependency*, *association*, *aggregation* and *specialization* [17] relations can be used.

The organization diagrams aim at modeling the organizations of the system and the relations between environments and agents and between environments and organizations. The following relations are used: *ownership*, *play* and *inhabit*. There must be an organization diagram for each organization in the system, exhibiting the environment where it inhabits, the roles, objects, agents and sub-organizations that play those roles.

The UML sequence diagram was extended to describe the dynamic aspects of the MAS, representing the interaction among agents, organizations, environments and objects. Besides extending the definition of the UML <<create>> and <<destroy>> stereotypes to create and destroy agents, organizations and environments, MAS-ML also introduces new stereotypes [17]: <<role_commitment>> – designates an element (agent, organization or object), registering a new role without canceling its previous role; <<role_cancel>> – plays an element canceling a role ; <<role_deactivate>> – plays an element deactivating a role – <<role_activate>>: plays an element activating a role; and <<role_change>> – plays an element changing its role.

4 Case study: MAS to Schedule Lectures in Conferences

To exemplify and evaluate the use of the AUML and MAS-ML languages, a MAS case study was carried out to achieve the best possible schedule for lectures in a conference. Each agent has a list of subjects of interest and a timetable with sessions of one hour, from 8 a.m. to 6 p.m. Each hour in the timetable may be free or busy with some other activity. A lecture about a subject of interest can be scheduled at the free hours. Both the list of interest and the initial state of the timetables are started at random. The agents can play two roles: organizer or participant. The organizer tries to schedule a lecture about a certain subject (one of the subjects of the list of interests) and achieves the highest possible number of participants. The participant tries to attend the highest possible number of available lectures in his list of subjects of interest. The organizer must attend the lecture he organizes, in other words, he plays the role of organizer and participant in the system.

The organizing agent announces a subject and waits for the feedback from the interested agents. After receiving the message from those interested, he proposes a free hour from his timetable for the lecture. The interested agents respond if they agree with the hour proposed or propose another hour. The negotiations must continue until an hour is found that can meet the highest number of interested agents.

Several diagrams were modeled before the implementation of the MAS, but not all of them are presented in the following sections. The goal is to briefly present the relevant considerations about modeling using each one of the languages.

4.1 MAS Modeling Using AUML

The MAS modeling to schedule lectures using AUML started with the class modeling, illustrated in the Fig. 1. In this diagram, it is demonstrated that the organizing agent can play two roles.

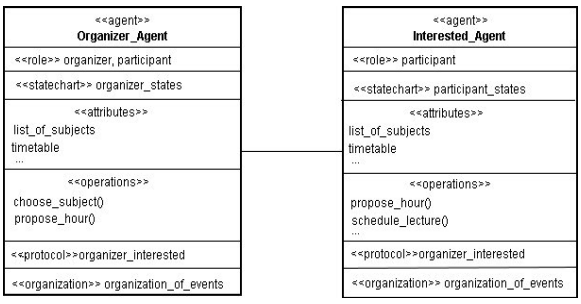

Fig. 1. UML class diagram for MAS in study.

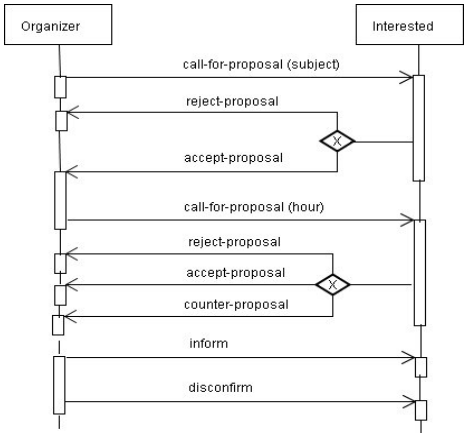

Fig. 2. Agent Interaction Protocol for MAS in study.

Fig. 2 illustrates the AIP that describes the sequence of the ACL messages exchanged between the organizer and the interested. The organizing agent sends a message proposing a subject. The interested verifies if the proposed subject is in his list of interests and can refuse or accept the proposal. If he accepts the proposal, the organizer sends a message proposing a certain hour. The interested can: refuse the proposal, accept the proposal or reject the hour and propose another. When the organizer receives an acceptance of the proposal, he must send a message confirming or canceling the lecture.

The collaboration and activity diagrams were also used to represent the interactions among the agents. Since the organizer must attend the lecture he organizes, the collaboration diagram (Fig. 3) was essential to specify that he plays the role of participant when he tries to negotiate the time of the lecture.

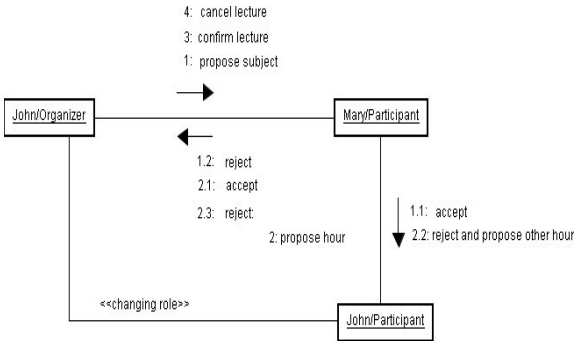

Fig. 3. Collaboration diagram for MAS in study.

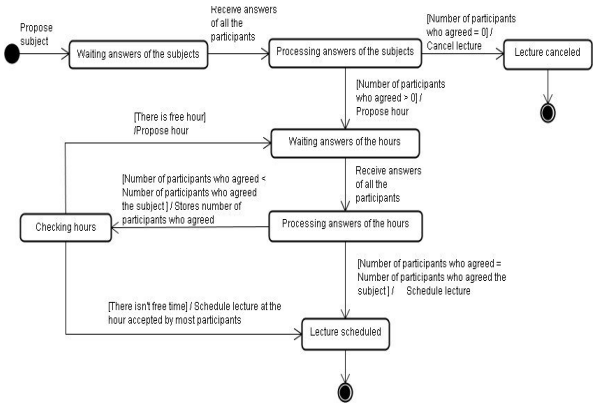

Fig. 4. AUML state diagram for the organizer role.

The state diagram was used to represent the internal processing of the organizer and participant roles. In Fig. 4, it is possible to observe that the organizer only proposes an hour after receiving the answers of the proposal of the subject of all the participants and that if there is no free hour common to all the interested, the lecture is scheduled for the hour at which the highest number of interested is free. These and other restrictions are well represented in the state diagrams.

4.2 MAS Modeling Using MAS-ML

The MAS modeling to schedule lectures using MAS-ML started with the modeling of the organization diagram (Fig. 5). It is an example of the so-called "organization of events", the roles of the organizing and participant agents defined by it and the organizing and interested agents that represent these roles. Two roles of object, called offer and desire, were defined. They are exerted by the instances of the lecture class. In this diagram, it is specified that the organizing agent plays more than one role in the organization through the relation *play*.

Next, class diagrams were created for organization, agents, roles and objects defined in the diagram of the organization. For example, Fig. 6 exemplifies the participant role class. The participant role aims at (<<*goal*>>) attending lectures. To achieve this objective, the agents playing this role negotiate with the agents playing the organizer role through the "schedule lecture" protocol, which describes how the entities that are playing these roles should interact. The decision of the participant to accept or refuse the subject proposed

represents the rights (<<*right*>>) of the participant role, as well as the decision to accept or refuse the hour proposed later.

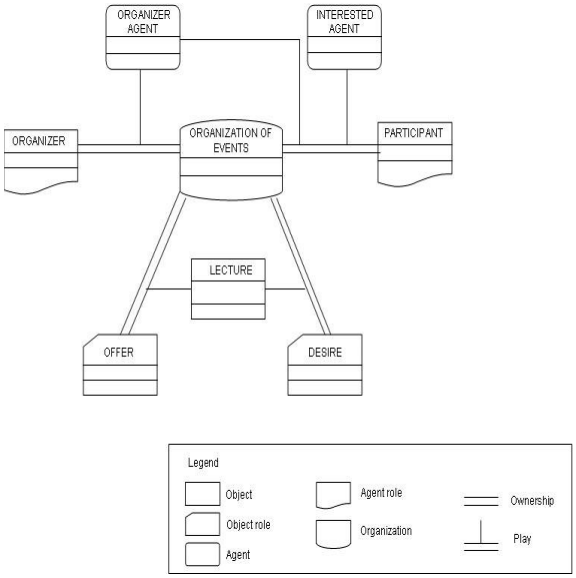

Fig. 5. MAS-ML organization diagram for the MAS.

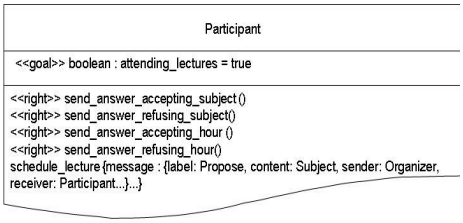

Fig. 6. Participant role Class.

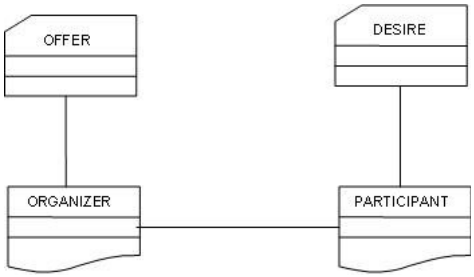

Fig. 7. Role diagram.

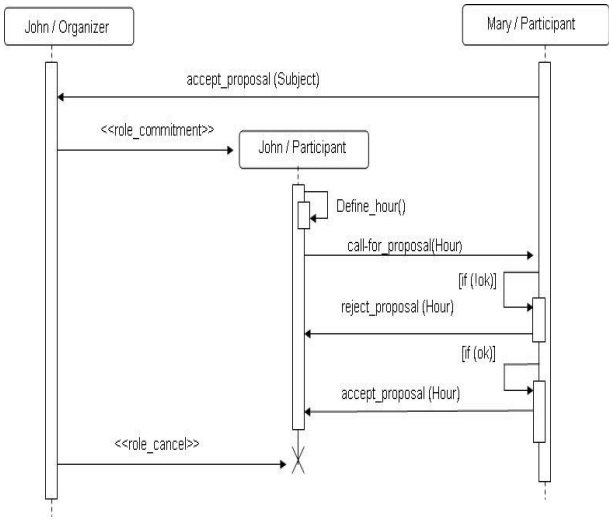

Fig. 8. Part of the MAS-ML sequence diagram exemplifying the change of roles.

A role diagram was built to illustrate the relations between the roles of the agents and the roles of the objects. Organizers and participants have different visions about the lectures that they negotiate. The diagram of the Fig. 7 shows that a lecture is a desire for the participants and an offer for the organizers.

To finish modeling, the dynamic aspects of the system were described, using the MAS-ML sequence diagram. The diagram describes all the actions and messages sent and received related to the "schedule lecture" protocol, defined in the organizer and participant classes. A part of this diagram that shows the change of roles of the agent is exemplified in the Fig. 8. When an agent playing the organizer role receives the answer of a participant agent accepting the subject proposed, he starts to play the participant role to negotiate the timetable. This role is canceled when the hour proposed is accepted.

5 Comparative Analysis Between the AUML and MAS-ML

It is possible to observe in Table 1 that the MAS-ML language presents more diagrams for the description of static aspects than the AUML language. It presents a clear definition of the elements that compose the MAS,

such as environments and organizations. It also makes a distinction between the representations of the agents and those of the objects, which does not occur in the AUML. Therefore, in the modeling of the static characteristics, the MAS-ML allowed a more comprehensive specification of the classes, roles and organizations and the relations between these entities.

In modeling dynamic characteristics, the AUML language extends four diagrams, while the MAS-ML extends only the sequence diagram. These AUML diagrams were essential for the understanding of the communications and behaviors of the agents. On the other hand, some users may feel confused with so many diagrams when they try to select the best one to represent a requirement. It is necessary to be attentive to the characteristics of each diagram in order to make the best choice. For example, the collaboration and sequence diagrams are equivalent, but to emphasize the multiple roles represented by the agents, the collaboration diagram must be used.

Table 1. Diagrams of the AUML and MAS-ML languages.

	AUML	MAS-ML
Description of static aspects	- Class diagram	- Class diagram - Role diagram - Organization diagram
Description of dynamic aspects	- Sequence diagram - Collaboration diagram - State diagram - Activity diagram	- Sequence diagram

Although the AUML collaboration diagram specifies the change of the role of the organizing agent during the attempt to negotiate the hour of the lecture, it is difficult to understand when the agent starts and stops playing this role. It is represented in a more simple form with the use of the new stereotypes created (*role-commitment* and *role-cancel*) in MAS-ML in the sequence diagram. The model that used MAS-ML lacked the support to express concurrent lines of interaction, which is present in the AUML. Therefore, modeling decisions in the MAS-ML sequence diagrams was more difficult and harder to be understood after modeled. Observe the modeling of the decision to accept or reject the hour proposed in the Figs 2 and 8, which represent an AIP in AUML and MAS-ML, respectively. The comparative chart in the Table 2 shows desirable characteristics in the MAS modeling and if they are met or not by the AUML and MAS-ML languages.

Table 2. Comparative chart of the AUML and MAS-ML languages.

Desirable characteristics	AUML	MAS-ML
Definition of environments and organizations	-	X
Distinction between agents and objects	-	X
Representation of multiple roles for agents	X	X
Representation of the registered/canceled role	-	X
Express concurrent interaction lines in the AIP modeling.	X	-

As for the modeling tools, the site of the AUML language makes a *link* available for several CASE UML tools that can be used. The MAS-ML presents two new diagrams with symbols different from those used in UML diagrams. That is why it needs a specific modeling tool.

6 Conclusions

Some works also present comparisons among modeling techniques. The work of [6] analyzes and compares seven modeling techniques used to represent the social organization of a MAS. It emphasizes the AGR, MOISE+ and ISLANDER and briefly describes AUML and MAS-ML. Its main objective was to present a proposal of an ontology of concepts to represent MAS organizations. Another work [10] focuses on the project stage of AIP. This work considers the languages AUML and FIPA-AUML to provide some criteria for the comparison of agent modeling languages. [8] compare AUML to UML2 and conclude that it is possible to represent MAS by means of UML 2, but that it still presents great limitations to represent the intentional aspects of the systems. Therefore, it is not adequate for a project of cognitive agents, such as agents based on the cognitive metaphor of the BDI type [14]. None of the

works investigated presented a detailed comparison between the AUML and MAS-ML languages.

During the modeling of the case study, it was possible to observe how the characteristics of the agents are treated in the diagrams, the differences between the languages and the advantages and disadvantages of each one. Modeling the same system in the two languages allowed more understanding of the MAS, compared to modeling in just one of them, since the AUML models the dynamic characteristics in a more comprehensive way, while the MAS-ML does the same to the static characteristics of the system. An advantage of the AUML is that it does not change the UML metamodel, allowing modeling in the existing CASE tools, in opposition to the MAS-ML, which demands its own modeling tool, but has the advantage of expressing better the static aspects, providing a more detailed model of the system that is being modeled.

Since they extend the UML, both languages present as an advantage the fast learning curve for software programmers who know the UML and how to use it. These people will have no difficulties to understand and use AUML and MAS-ML.

It can be concluded that, in environments where there are plenty of roles that agents may undertake, the MAS-ML would be a better choice as a modeling language, as it best reflected the adoption of roles by agents.

The software engineering based on agents is a new area, in which the modeling tools have not been stabilized yet. Other languages and methodologies are being proposed, such as the AORML (Agent-Object Relationship Modeling Language) [18] and the TROPOS methodology [5]. While there is not a "prevailing" language, such as the case of the UML for the OO project area, comparative analyses, similar to that carried out in this article, will be useful to guide modelers of MAS.

References

1. AUML. Agent Unified Modeling Language**.** Available on line at: http://www.auml.org. Last access: 22/11/2008. (2008)
2. Bastos, R. M. "O Planejamento de Alocação de Recursos Baseado em Sistema Multi-Agentes", Porto Alegre: CPGCC da UFRGS. Doctoral thesis. (1998)
3. Bauer, B. UML Class Diagrams Revisited in the Context of Agent-Based Systems, In: Agent-Oriented Software Engineering II: Second International Workshop, Lecture Notes in Computer Science 2222, Springer-Verlag, pp.101—118. (2002)
4. Bauer, B., Odell, J. UML 2.0 and agents: how to build agent-based systems with the new UML standard. Journal of Engineering Applications of Artificial Intelligence Volume 18, Issue 2 , March 2005, Pages 141-157. (2005)
5. Bresciani, P., Giorgini, P., Giunchiglia, F., Mylopoulos, J., Perini, A. TROPOS: An Agent-Oriented Software Development Methodology. In Journal of Autonomous Agents and Multi-Agent Systems. Kluwer Academic Publishers. (2004)
6. Coutinho, L. R., Sichman, J.S., Boissier, O. "Modeling Organization in MAS: A Comparison of Models". In: Proceedings of the 1st International Workshop on Software Engineering for Agent-Oriented Systems (SEAS '05). Pages 1–10. (2005)
7. FIPA. Foundation for Intelligent Physical Agents. Available on line at: www.fipa.org. Last access: 04/11/2008. (2008)
8. Guedes G.T. A., Vicari, R. M. Applying AUML and UML 2 in the Multi-agent Systems Project. Advances in Conceptual Modeling - Challenging Perspectives. Berlim : Springer, 2009. v. 5833. p. 106-115. (2009)
9. Huget, M. P. Agent UML Class Diagrams Revisited. In: Bauer, B., Fischer, K., Muller, J., and Rumpe, B. (Eds.) Proceedings of Agent Technology and Software Engineering (AgeS), Erfurt, Germany.. (2002)
10. Koning, J. Huget, M. P., Wei, J. and Wang, X. Extended Modeling Languages for Interaction Protocol Design. *In Proceedings of the Second International Workshop on Agent-Oriented Software Engineering* (AOSE-2001), Montreal, Canada. (2001)
11. Odell, J., Parunak, H. V. D. and Bauer, B. Extending UML for agents. In: Proceedings of the agent-oriented information systems workshop at the 17th national conference on artificial intelligence, 2000. (2000)
12. Odell, J., Parunak, H. V. D. and Bauer, B. Representing agent interaction protocols in UML, *In: Proceedings of the 1st International Workshop on Agent-Oriented Software Engineering*, Lecture Notes in Computer Science, vol. 1957, Springer-Verlag, New York, pp. 121–140. (2001)
13. OMG. Object Management Group. Available on line at: www.omg.org. Last access: 03/11/2008. (2008)

14. Rao, A. S., Georgeff, M. P.. Modeling Rational Agents within a BDI-Architecture. In Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning, pages 473-484. (1991)
15. Silva, V., Garcia, A., Brandao, A., Chavez, C., Lucena, C. and Alencar, P. Taming Agents and Objects in Software Engineering, In: Garcia, A., Lucena, C., Zamboneli, F., Omicini, A., Castro, J. (Eds.) Software Engineering for Large-Scale Multi-Agent Systems, LNCS 2603,Springer-Verlag.. (2003)
16. Silva, V. T. and Lucena ,C. J. P. A UML Based Approach for Modeling and Implementing Multi-Agent System. In: Journal of Autonomous Agents and Multi-Agent Sytems, ACM 1-58113-864-4/04/0007. (2004)
17. Silva, V.T. and Lucena, C.J.P. From a conceptual framework for agents and objects to a multi-agent system modeling language. In: Journal Autonomous Agents and Multi-Agent Systems, 8: 1-45. (2004)
18. Wagner, G. The Agent-Object-Relationship Metamodel: Towards a Unified View of State and Behavior. In: Information Systems, v.28, n.5, pp. 475—504.. (2003)
19. Wooldridge, M. An Introduction to Multiagent Systems, John Wiley & Sons. (2002)