

A Roadmap to a Modeling Language for Multi-Agent Systems Engineering

Viviane Silva¹

Carlos Lucena¹

Paulo Alencar²

Donald Cowan²

¹ PUC-Rio, Computer Science Department, SoC+Agent Group,

Rua Marques de São Vicente, 225 - 22453-900, Rio de Janeiro, RJ, Brazil

{lucena, viviane}@inf.puc-rio.br

² University of Waterloo, School of Computer Science

200 University Avenue West, Waterloo, ON N2L 3G1

1-519-8884690

Waterloo, Ontario, N2L 3G1 Canada

{palencar, cowan}@csg.uwaterloo.ca

Headline

Complex behavior is often achieved by a collection of individual entities working autonomously on shared goals, but with no apparent central control. Many different mechanisms such as the human body and traffic within a city seem to operate in this manner. Such autonomous operation does not always reach the goal through the most direct route, as there is often a period of mutual exploration and learning on the part of the entities, before success is achieved. Currently most computer applications operate on a model where some program or person is in charge as requests are processed. However, as computational tasks and supporting services become more distributed through technologies such as the Internet, we will likely have to re-think our processing models. Collections of computer-based agents, often called multi-agent systems (MAS), working

together with no central control on shared goals seem to be a model worth exploring and there have been numerous papers published in the literature [e.g.: 5, 12].

A B2B e-Commerce application is an example of a MAS where agents autonomously act on behalf of human individuals. Agents are created to play roles such as buyers and sellers in organizations such as auctions. Different types of organizations such as English or Dutch auctions can be created by seller agents depending on the needs of the individuals. In B2B e-Commerce the agents are involved in negotiations between multiple parties using multiple parameters.

If we are to have MAS as tools for supporting computational processes we will have to learn how to deal with novel concepts such as agent relationships, roles that agents perform in organizations and agents forming new organizations. A first step in this direction is the creation of a conceptual framework that organizes the core concepts of multi-agent systems and relates these ideas to other programming abstractions such as objects. Further we will need to create new software engineering modeling approaches and languages to specify, describe, analyze, and implement our multi-agent designs. Creating a modeling language for multi-agent systems is a complex task and we should consider following a process or roadmap similar to the one that led to the development of the Unified Modeling Language (UML) [10] and its various sub-models and supporting facilities.

In this paper we present a conceptual framework called TAO that organizes the core concepts related to multi-agent systems and assumes that agents and objects co-exist as independent and unique abstractions. We then use the TAO framework to create the multi-agent system modeling language MAS-ML, which extends UML. MAS-ML is

defined as a conservative extension of the UML metamodel, and includes agent-related notions that are part of the TAO conceptual framework while preserving all object-related concepts, which constitute the UML metamodel. In order to extend UML according to TAO non-object concepts, new metaclasses and stereotypes have been created and associated with the UML metamodel. Our approach is similar to that recently proposed by FIPA for An Agent-based Unified Modeling Language (AUML) [4,7,11] where they intend to capitalize on the UML experience and reuse UML where appropriate.

THE TAO CONCEPTUAL FRAMEWORK

The TAO (Taming Agents and Objects) conceptual framework is defined by its structural and dynamic metamodels [8].

The TAO Structural Metamodel. The structural metamodel is defined by entities, properties and relationships. There are seven entities defined in TAO: object, agent, organization, role (agent role and object role), environment and event. Because of similarities among some entities, we have defined a new abstraction called element that is the basis for the definition of most entities. An element is an entity that has properties and relationships with other elements. An element instance is a concrete manifestation of an element class to which a set of properties and relationships are applied [10]. Event is the only entity in TAO that is not based on the definition of element because events do not have state, behavior or relationships.

The properties of an element describe its state and behavior characteristics. The state of an element defines information about other elements of the system and the behavior of an element defines the actions or operations that the element can perform. Relationships link two elements and describe how these elements are related to each other. TAO defines

eight different relationships that associate objects, agents, environments, organizations and roles. The relationships defined in TAO are: inhabit, ownership, play, specialization, control, dependency, association and aggregation.

The TAO Dynamic Metamodel. The dynamic metamodel is defined by primitive (or elementary) and high-level dynamic processes. These processes involve creation of elements, destruction of elements and interaction among elements. The elementary dynamic processes are domain-independent and can be used to define high-level dynamic patterns and domain specific behavior. Primitive processes define, for example, organization instance creation, agent instance creation, and role instance creation. The high-level dynamic processes, such as agents entering an organization and creating sub-organizations, are also domain-independent and can be defined in terms of the primitive processes.

The dynamic processes are clearly related to each other. For example, the creation of an agent is related to the creation of a role and to the creation of an organization since an agent plays at least one role in an organization. The creation of an organization by an agent depends on the creation of a role instance to be played by the organization and on the creation of another role instance to be played by the agent in the organization. While the dynamics of creating an organization instance involve a general process that can be used to create any organization in the system, the dynamics of an agent creating an organization involve the motivation for an agent to create a sub-organization and how that task is accomplished.

Because of the richness of the multi-agent model there is a potentially infinite number of domain-independent high-level dynamic behaviors involving agents, roles and


organizations. A systematic classification of these high-level dynamics may lead to the definition of new and complex behavioral patterns.

FOLLOWING THE ROADMAP

Defining a modeling language for MAS-ML requires following a roadmap similar to the one that led to the development of the UML. Following such a roadmap may include the need to define a counter-part of the Meta-Object Facility (MOF).

In order to explain the relationship among TAO, MAS-ML and UML better, we use the four-layer metadata architecture described in the MOF specification [6]. The four architectural layers are: meta-metamodel layer, metamodel layer, domain model layer and instance layer (see Table 1). The meta-metamodel layer consists of a description of the structure and semantics of the meta-metadata. The Object Management Group (OMG) has specified a meta-metamodel called MOF that defines an abstract language and a framework for specifying, constructing, and managing technology neutral metamodels. Following this approach, in TAO we use the ER model (Entity-Relationship model) to describe the entity and relationship meta-metadata that appear in this layer. ER is used as a meta-metamodel because it describes the entity and relationship meta-metadata that provides the basic definitions to describe the different entities and relationship instances which appear in the metamodel layer.

Table 1 – Four-layer metadata architecture

Layers	Models	
Meta-metamodel layer	MOF meta-metamodel	ER meta-metamodel
Metamodel layer	UML metamodel	TAO metamodel
		
Domain model layer	MAS-ML models	
Instance layer	Instances of the domain models	

The metamodel layer consists of a description of the structure and semantics of metadata that are informally aggregated as metamodels. Metamodels are instances of meta-metamodels and metadata are instances of meta-metadata. OMG has defined the UML metamodel as an instance of the MOF meta-metamodel. We have defined the TAO metamodel that we call a conceptual framework [8] as an instance of the ER meta-metamodel.

The UML metamodel specifies a modeling language that incorporates the object-oriented community's consensus on core modeling concepts. The TAO metamodel specifies the MAS core concepts (abstractions and their relationships) that incorporate object-oriented concepts and new concepts defined for agent-oriented development based on our experience and on work described in the literature. Following the roadmap, one of our goals is to define a MAS-ML metamodel that extends the UML metamodel based on the concepts described in the TAO metamodel. The MAS-ML specifies a modeling language

that incorporates both object- and agent-oriented concepts. In this sense, the MAS-ML unifies the UML metamodel and the TAO metamodel.

The domain model layer depicts the data specific to the application domain generating MAS-ML models. The metadata in the metamodel layer is instantiated into concrete data through domain models using domain-related information.

The instance (information) layer characterizes the possible domain model applications. This layer describes the specific instances of the domain model data that may occur during the lifetime of the modeled applications.

MAS-ML AS A TAO-BASED EXTENSION OF THE UML METAMODEL

To extend UML according to the TAO non-object concepts, new metaclasses and stereotypes have been created and associated with the UML metamodel. Thus, new diagrams such as Organization Class and Role Class have been created and UML diagrams such as Class Diagram and Sequence Diagram have been adapted.

TAO defines three main concepts – elements, properties and relationships – that have been mapped to the UML metamodel. Some elements defined in TAO were directly mapped to existing UML metaclasses and some properties were directly associated with existing UML features through the use of stereotypes. Other elements caused the creation of new metaclasses and other properties caused the creation of new UML features. A set of relationships such as association and aggregation described in TAO have been associated with the respective relationships defined in the UML Class diagram. Some relationships that are typical of MAS have been created and incorporated into new static diagrams.

As initial steps, we have adapted the static Class diagrams and the interaction Sequence diagram based on TAO concepts. These two diagrams have been chosen because they are the most commonly used and because it is possible to use them to illustrate both the structural and dynamic aspects of the TAO metamodel.

Extending the UML Foundation Package

To map TAO concepts into UML, we have created new stereotypes, tags and metaclasses. Table 2 describes three templates used to guide the definition of each new element based on the UML specification style [10].

Table 2 – Template attributes used to define new stereotypes, tags and metaclasses.

Stereotypes	Tags	Metaclasses
<i>Stereotype:</i>	<i>Tag:</i>	<i>Metaclass:</i>
<i>Base Class:</i>	<i>Stereotype:</i>	<i>Description:</i>
<i>Description:</i>	<i>Description:</i>	<i>Parent:</i>
<i>Parent:</i>	<i>Type:</i>	<i>Attributes:</i>
<i>Tags:</i>	<i>Multiplicity:</i>	<i>Name</i>
<i>Constraints:</i>		<i>Associations:</i>
		<i>Name</i>
		<i>Stereotypes:</i>
		<i>Name</i>

An element declared in TAO is represented as an extension of the *Classifier* metaclass defined in UML since they share similar characteristics. A *Classifier* has structural features, behavioral features and relationships to other *Classifiers* and an element has properties (state and behavior) and relationships with other elements. Figure 1 presents a set of metaclasses of the UML metamodel and all the extensions that we have proposed.

This figure shows the new metaclasses and the new stereotypes that have been created. The icons that represent the stereotypes are associated with the metaclasses on which the stereotypes are based.

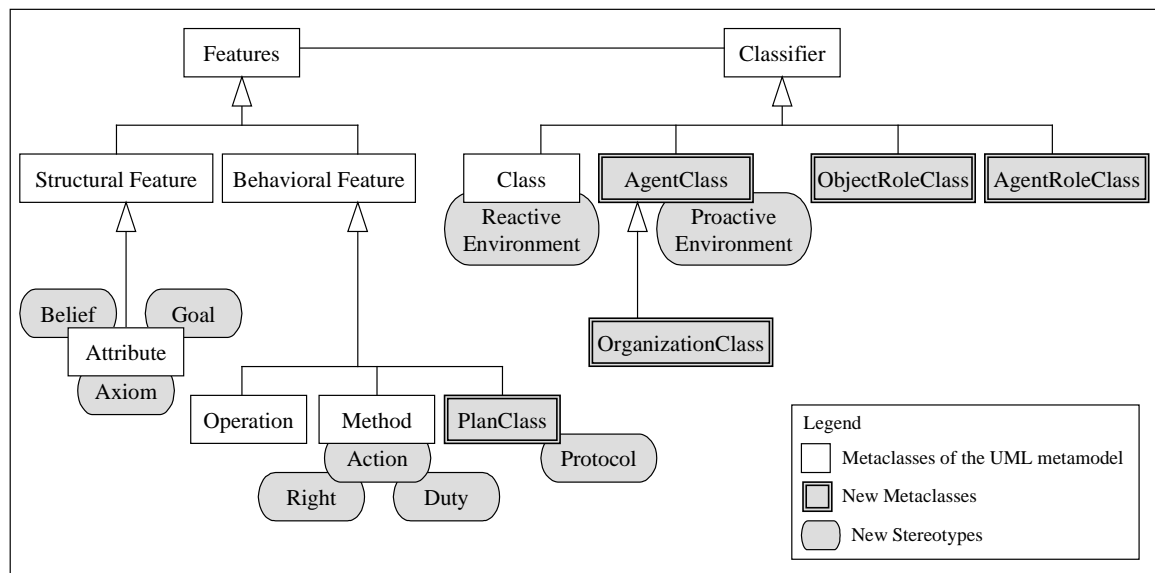


Figure 1 - The extended UML metamodel

Structure Diagrams

When introducing new abstractions in the UML metamodel, new structure diagram elements and features need to be created to represent the new elements and their relationships. We have adapted the UML Class diagram and created two new structure diagrams – Organization diagram and Role diagram – to focus on different aspects of extensions.

The TAO definitions of object/class and event entities are similar to the definitions of the UML metaclasses Class and Event. So, we have not created new structure diagram elements to represent these two entities. New diagram elements have been associated

with the metaclasses AgentClass, OrganizationClass, EnvironmentClass, AgentRoleClass and ObjectRoleClass.

The association, specialization, aggregation and dependency relationships described in TAO are also described in UML with the same semantics. The difference between the relationships defined in TAO and the ones defined in UML are the entities that participate in the relationship. Although TAO applies these relationships to elements such as organization, agent and agent role, we did not define new diagram elements to represent them. New diagram elements have been created and associated with new relationships defined in TAO that do not exist in UML. Those relationships are Inhabit, Ownership, Play and Control.

The extended class diagram represents the relationships between the resources (objects) found in the environment and between the resources and the agents that use the resources. The metaclasses that may appear in this diagram are Agent, Class and other metaclasses defined by UML. The relationships used in this diagram are those defined by UML plus association (between agents and objects), aggregation (between agents), and specialization (between agents).

The organization diagram focuses on the relationships between organizations and other elements. The metaclasses that may be used in organization diagrams are OrganizationClass, AgentClass, AgentRoleClass, Class, ObjectRoleClass and Environment. The relationships that may be used are: ownership (between organizations, roles and sub-organizations), play (between agents, sub-organizations, objects and their roles), inhabit (between the environments and other elements), association (between organizations and objects) and specialization (between organizations).

The role diagram is responsible for clarifying the relationships between the agent roles and object roles. This diagram shows the relationships between *AgentRoleClass* and *ObjectRoleClass*. The set of relationships used in this diagram is: control (between roles), dependency (between roles), and association (between object and agent roles, object roles, and agent roles), and aggregation and specialization (between object roles and agent roles).

As an example, Figure 2 illustrates an organization diagram presenting the *Market Place* organization of a market place system. This organization defines three different roles classes – *Auction Market*, *Market of Used Goods* and *Supermarket* – that may be played by three different sub-organizations classes – *English Auction Market*, *Books Market* and *Retail Market*, respectively. The organization *Retail and Wholesale Market* is a specialization of *Retail Market*. All the elements defined in the *Market Place* inhabit the same environment called *Virtual Market Web Service*.

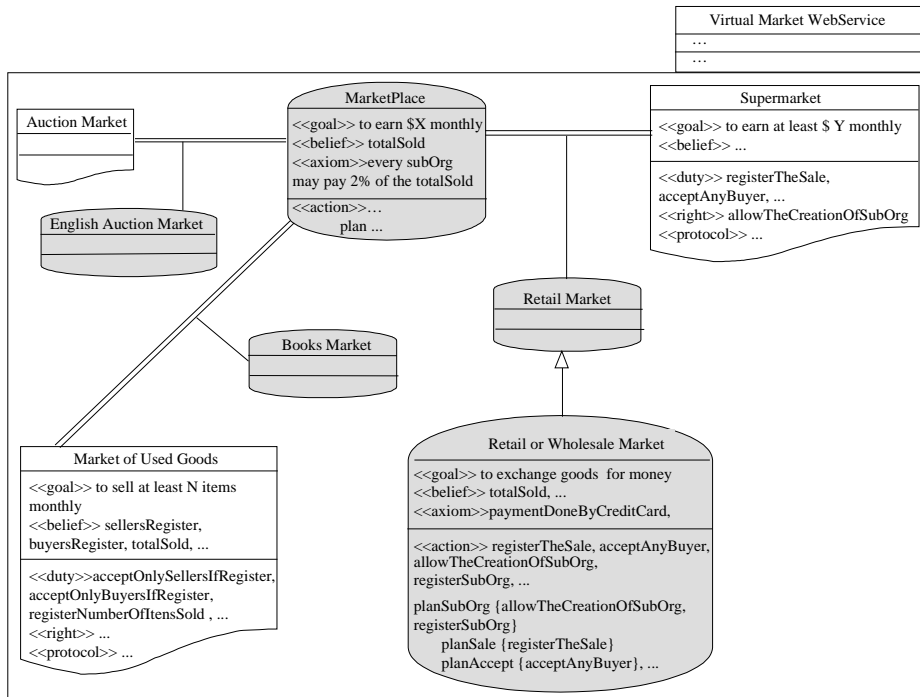




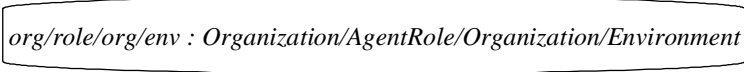






Figure 2 - Organization Diagram

Sequence Diagram

A UML sequence diagram presents a set of interactions between objects playing roles in collaborations. We propose to extend the sequence diagram to represent the behavior of agents, organizations and environments.

While extending the UML sequence diagram, three sequence diagram elements were created and the existing diagram element called object was modified. Table 3 identifies the instances that may appear in a sequence diagram and its associated diagram elements.

Table 3 - The sequence diagram elements

Instances	Diagram Element				
Object					
Agent					
Organization					
Environment	<table border="0" style="width: 100%;"> <tr> <td style="text-align: center; padding-right: 20px;"><i>Proactive element</i></td> <td style="text-align: center;"><i>Reactive element</i></td> </tr> <tr> <td style="text-align: center;">  </td> <td style="text-align: center;">  </td> </tr> </table>	<i>Proactive element</i>	<i>Reactive element</i>		
<i>Proactive element</i>	<i>Reactive element</i>				
					

In addition, we have created new stereotypes to identify new interaction types – <<role_commitment>>, <<role_cancel>> and <<role_change>> – and have extended the definition of the stereotype <<create>> and <<destroy>> defined in the UML metamodel. The extensions proposed to the UML sequence diagram make it possible, for instance, to represent the dynamic processes previously described in this paper.

Using the market place system as example, Figure 3 shows a *User Agent* called *Bob* searching for an organization to play a role. *Bob* is interested in entering an organization to buy a wholesale item. This figure illustrates interactions between agents playing different roles, an environment and an organization. The figure also illustrates an agent committing to another role and an agent canceling a role.

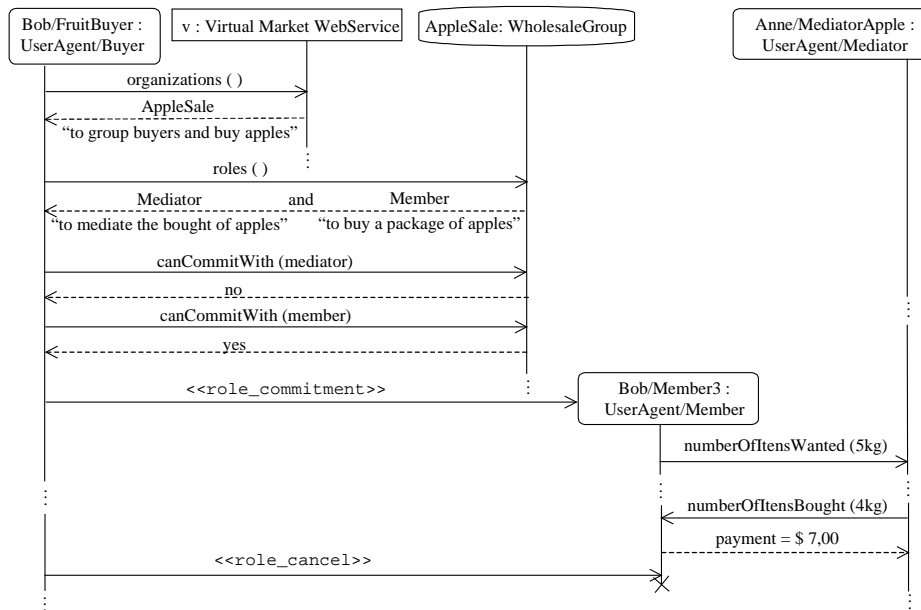


Figure 3 – Partial sequence diagram representing illustrative interactions

RELATED WORK

Organizations such as FIPA and the OMG Agent Work Group have been proposing AUML [4,7], an UML extension based on characteristics identified in MASs. AORUML (Agent-Object-Relationship UML) [11] has a similar intent. In these approaches stereotypes are used to create agent classes based on the metaclass Class that describes object classes. However, stereotypes may be used to indicate a difference in meaning or usage between two model elements with identical structure and, therefore, they are not appropriate to describe the different structures defined by objects and agents. In addition, these approaches do not describe possible relationships between objects and agents, and do not describe other entities such as environment and organization that are frequently used in MAS models. They also do not describe how to represent in their proposed sequence diagrams the interactions between agents, organizations and environment, features such as agents playing more than one role, and how to represent agents entering

an organization and migrating to another environment. The main difference between our approach and others presented in the literature is the proposal of a clear definition and representation of the elements that compose MASs and their behaviors. Some proposals describing the dynamic process of MASs are presented in [2,3]. However, in [2] they do not provide explicit representations for dynamic processes that describe the primitive and high-level interactions between agents and organizations such as role instance creation or agents playing roles in organizations. In [3] the authors describe a process for the reorganization of societies where agents enter societies to play roles but their approach focuses on utility functions and not on the representations.

CONCLUSIONS

In this paper we have proposed a modeling language for multi-agent systems, which we call MAS-ML. We have begun by developing a conceptual framework for MASs called TAO and describing how this framework led to the definition of the MAS modeling language called MAS-ML, a conservative extension of the UML metamodel which includes the agent-related notions from TAO.

However, we still need to produce fully developed MAS software engineering techniques, including modeling languages. For example, in defining MAS-ML, we have only provided UML extensions related to two of the UML set of diagrams. The extension of UML according to TAO impacts other diagrams and we are in the process of analyzing and extending other UML diagrams. There are still challenges such as the refinement of our structural and dynamic process representations in cases where agents and roles are destroyed, when organizations with internal agents playing different roles are destroyed, or when agents migrate between environments. We also need to develop a comprehensive

multi-agent counter-part of the Meta Object Facility and analyze the extension of UML sub-models and sub-languages such as OCL (Object Constraint Language) to handle constraints among goals, beliefs, events, actions and plans, and defining extensive MAS declarative approaches to experiment with our models based on technologies such as XML schemas [1].

ACKNOWLEDGEMENTS

This work has been partially supported by the National Sciences and Engineering Research Council of Canada (NSERC), the National Research Council of Brazil (CNPq), PRONEX, ESSMA, and IBM.

REFERENCES

- [1] Alencar, P.; Oliveira, T.; Silva, V.; Garcia, A.; Lucena, C.; Cowan, D.: Towards a Monitored Data Consistency and Business Processing based on Declarative Software Agents. In: Garcia, A.; Lucena, C.; Zamobnelei, F.; Omicini, A; Castro, J. (Eds.): Software Engineering for Large-Scale Multi-Agent System. Springer-Verlag, LNCS, 2003.
- [2] Ferber, J.; Gutknecht, O.: A meta-model for the analysis and design of organizations in multi-gents systems. In: Demazeau, Y., (Ed.): Proc. of the International Conference on Multi-Agent Systems, pages 128 135. IEEE Press, 1998.
- [3] Glaser, N; Morignot, P.: The Reorganization of Societies of Autonomous Agents, Multi-Agent Rationality. In: Boman, M., Velde, W. (Eds.): Proc. of the 8 th European Workshop on Modeling Autonomous Agents in a Multi-Agent World, SpringerVerlag, Berlin, Germany, 1997.

- [4] Huget, M.: Agent UML Class Diagrams Revisited. In: Bauer, B., Fischer, K., Muller, J., Rumpe, B. (Eds.): Proc. of Agent Technology and Software Engineering (AgeS), Erfurt, Germany, October 2002.
- [5] Jennings, N.R. and Wooldridge, M. Intelligent Agents: Theory and Practice. In: The Knowledge Engineering Review, Volume 10, Number 2, pages 115-152, 1995.
- [6] Meta Object Facility (MOF) Specification. Version 1.4, April, (2002). Available at URL <http://www.omg.org/cwm/a>
- [7] Odell, J.; Parunak, H.; Bauer, B.: Extending UML for agents. In: Wagner, G., Lesperance, Y., Yu, E. (Eds.): Proc. of the Agent-Oriented Information Systems Workshop (AOIS), pp. 3-17, 2000.
- [8] Silva, V.; Garcia, A.; Brandao, A.; Chavez, C.; Lucena, C.; Alencar, P.: Taming Agent and Object in Software Engineering. In: Garcia, A.; Lucena, C.; Zamobnelei, F.; Omicini, A; Carstro, J. (Eds.): Software Engineering for Large-Scale Multi-Agent System. Springer-Verlag, LNCS, 2003.
- [9] Silva, V.; Lucena, C., From a Conceptual Framework for Agents and Objects to a Multi-Agent System Modeling Language, Technical Report CS2003-10, School of Computer Science, University of Waterloo, Canada, 2003.
- [10] Unified Modeling Language (UML) Specification. Version 1.4, April, (2002). Available at URL <http://www.omg.org/uml/>
- [11] Wagner, G.: A UML Profile for External AOR Models. In: Proc. of 3rd International Workshop on Agent-Oriented Software Engineering (AOSE-2002), held at Autonomous

Agents & Multi-Agent Systems, Palazzo Re Enzo, Bologna, Italy - July 15, 2002.

Springer-Verlag LNAI, 2002.

[12]Zambonelli, F., Jennings, N. and Wooldridge, M.: Organizational Abstractions for the Analysis and Design of Multi-agent Systems. In: Ciancarini, P. and Wooldridge, M., (Eds.): Agent-Oriented Software Engineering, Springer-Verlag, 2001.