

# A Formal Model for Coordination Behavior of the Organization in Multi Agent Systems

Fatemeh Ghassemi, Naser Nemat Bakhsh, Behrouz Tork Ladani  
Isfahan University, Esfahan, Iran  
{fghassemi,nemat,ladani}@eng.ui.ac.ir

Marjan Sirjani  
Department of Electrical and Computer Engineering, Tehran University, Tehran, Iran  
School of Computer Science, IPM, Tehran, Iran  
[msirjani@ut.ac.ir](mailto:msirjani@ut.ac.ir)

## Abstract

*Multi agent systems are applied as a solution for distributed IT systems. Organizational concepts are usually applied to analyze and design such systems. Thus, a multi agent system can be seen as an organization which coordinates agent interactions. In this paper we propose a formal model to specify the coordination behavior of a multi agent system organization. This formal model enables the developers to have a cross checking between the agent interactions, the organizational structure and the coordination behavior of the organization. We can also apply this formal model to evaluate the system properties such as security.*

## 1. Introduction

Autonomous agents and multi-agent systems (MASs) are widely used by developers to design complex and distributed systems such as e-learning and e-marketing systems and business-to-business applications. Agents usually act on the behalf of a person in the virtual world of the system. An agent provides a behavior abstraction which allows the developers to naturally model and construct complex systems. These systems consist of lots of communicating agents, while their interactions are coordinated by the system. In the more complex systems, the number of involved agents and their interaction are increased such that it may lead to the system failure. It is needed to define properly how agent interactions should be coordinated with each other and what constraints affect on their interactions.

In the context of MASs, the autonomous and proactive behavior of agents suggests that applications can be designed by mimicking the behavior and structure of human organizations. Thus one can use organizational concepts to analyze and design such system. In this manner, the architecture of a multi-agent system can be naturally viewed as a computational organization, which consists of a multitude of interacting agents. Each agent plays one (or more) specific roles. However, the

organization of a multi-agent system is distinct from the individual agents that populate the system [1, 2, 3]. The responsibility of agents is to satisfy organization goals and the responsibility of the organization is to coordinate agent interactions [3,4]. Roughly speaking an organization is characterized by the organizational structure as well as organizational rules that define the requirements for the instantiation and operation of the organization as well as constraints on agent behaviors and interactions [1, 2].

Organization defines and coordinates agent interactions. So a multi-agent system is defined by a set of agents and its coordination behavior [4]. In this paper we propose a formal model to define the organization of a multi agent system. This model not only defines the coordination behavior of the organization formally but also defines how agents are allowed to act in an organization and what constraints affect on agent interactions. In this model, agents are block boxes that only their external behaviors are considered. This model enables us to specify *open organization* where, agents enter or leave dynamically. Thus agents are not known to each other and they may not be honest to each other.

*Structure of the paper:* The organization metaphor is described in Section 2. In Section 3, we explain our formal model for an organization. In Section 4, we specify an example system using our formal model. Finally in Section 5, we explain our concluding remarks.

## 2. Organizational Concepts

The field of multi-agent systems has long been interested in using social and biological concepts to analyze and design such systems. Applying organizational concepts in analysis and design of such systems helps developers to reduce the complexity and simplifies the system development procedure:

- There is no central control over the system and each agent is a locus of control and embeds most of functionalities it needs to satisfy its roles. So the number of dependencies and therefore interactions are reduced.

- These systems are usually developed to support the real world organizations. Thus applying organizational concepts to analyze and design result in a system that satisfy the real world organization needs.

The Organization of multi-agent system is the collection of roles, relationships and authority structures, which governs its behavior. All multi-agent systems posse some form of organization [5].

An organization can be specified in terms of roles and their interaction relation/structure which are usually modeled as interaction protocol. Agent-oriented methodologies such as Gaia [9] and Tropos [10] use a *role model* to show roles and the interaction protocol informally. Relationship and authority structure define an interaction relation/structure between agents. In other words it defines the position of each agent within the organization and is called *organizational structure*. Thus organizational structure derives a specific interaction protocol between agents and the admissible action in an agent interaction.

The organizational structures are usually described in terms of a variety of social and organizational concepts such as norm, trust, power, delegation of task, responsibilities, permission, access to resources and communication [4]. For instance when there is delegation relation between two agents, one agent can delegate task to another agent. So, the delegating agent has a delegation action in its interaction protocol.

In [5] organizational structure is basically seen in three types of relationship namely, *power*, *control* and *inform*. Thus agents can interact by delegating tasks to each other, taking responsibility for each other and passing information to each other.

The organizational structure is satisfied (implemented) correctly if all agents have knowledge about their delegated tasks and all agents satisfy their roles. The first condition is easily implemented if an inform structure exists next to the delegation structure. One can use control structure to satisfy the second condition but the higher level agent can not have a controller agent. So, in general the second condition can not be satisfied.

Organizational rules express general, global (supra-role) requirement for the proper instantiation and execution of a MAS [1, 2]. These rules indicate some constraints between two communicating agents or an agent and organization.

### 3. Formal Model

As described in Section 1, an organization coordinates agent interactions. Thus an organization can be viewed as a coordination artifact that coordinates the behavior and the interaction of agents in terms of long-term goals of system.

A formal model has been proposed in [7] to specify an environment-based coordination artifact. In the environment-based coordination model, agents are

coordinated via data existed in the environment. Thus, agents do actions using operations defined by the user interface of the artifact. When artifact receives an action, it is responsible to execute the action, and reify proper data to keep track of agent actions. These data define the coordination status of the coordination artifact. In this model, agents do not have a direct communication and they communicate via data reserved in the environment. The operating instructions of the artifact define for each agent how to exploit coordination service.

We have extended the formal model proposed in [7] with organizational concepts to specify an organization. The usage interface defines what actions an agent can do and the set of operating instructions defines the interaction protocol between agent and organization. The coordination behavior of the artifact defines how the organization coordinates the interactions of agents. In our model, agents can communicate directly, which may be synchronous or asynchronous.

An organization is specified by a tuple  $\langle R, A, \psi, \alpha, \beta, \rho, \delta, \rightarrow\sigma, \gamma \rangle$ . Some of these parameters are in common with the model in [7]. The set  $R$  defines the set of roles required within the organization to reach its goals. The set  $A$  defines the set of agents and the roles they play.

The meta-variable  $\psi$  is the set of binary relations, which defines the organizational structure of MAS in three dimensions of control, power and information:

$$\psi ::= \{power(r,s), control(r,s), inform(r,s), r,s \in R\}$$

These relations are not limited and users can define other (social) relations. For instance, the power relation specifies the agent enacting role  $r$  delegates tasks to the agent enacting role  $s$ . Note  $\psi$  is exploited to cross-check the consistency between organizational structure, agent interaction protocols and the coordination behavior. For example when a power relation exists between two roles, the delegating agent is allowed to delegate a task and the delegated agent should receive the task either synchronously or asynchronously.

The meta-variable  $\alpha$  ranges over the operations allowed by the organization to the agents and it defines the actions an agent can do/initiate. The meta-variable  $\beta$  ranges over the perceptions of action completion and it may contain some information about the outcome of the action. Therefore, the set  $L$  of interactions between agents and the organization, ranged over by  $l$ , is defined by the syntax as follows:

$$l ::= id!\alpha \mid id?\beta$$

The  $id!\alpha$  represents an agent identifier  $id$ , executes an action  $\alpha$ , and  $id?\beta$  represents agent  $id$  perceives the completion  $\beta$  for the action  $\alpha$ .

The function  $\rho$  associates to each agent identifier  $id$  the usage instruction  $I$  he is committed to follow in the organization and it defines the admissible actions and perceptions. Instructions can be defined by exploiting typical process algebra operators, i.e. by the syntax:

$$I ::= 0 \mid !\alpha \mid ?\beta \mid I+I \mid I;I \mid I|I$$

Where, 0 is the void instruction,  $!a$  is execution of an action,  $?b$  is perception of a completion, operator “+” is used for choice between instructions, “;” for sequential composition of instructions and “||” for parallel composition of instructions. The definition can be recursive. As an example, the definition  $I := !a ; (?b || I)$  means that the agent is initially allowed to do an action  $a$  and later, while it can do the whole protocol again, doing another action of  $a$ , it can perceive the completion of previous actions (i.e.  $b$ ) of  $a$ .

The meta-variable  $\delta$  ranges over the data reified into the organization (like databases or temporary containers) to possibly keep information of organization. Agents may not communicate directly with each other to coordinate their actions, thus they reify data into the organization which then taken by another agent to coordinate their behaviors. The meta-variable  $\sigma$  ranges over the set of  $\Sigma$  of states of the organization, which is defined as follows:

$$\sigma ::= 0 \mid \delta \mid l \mid (\sigma \parallel \sigma)$$

The operator  $\parallel$  is characterized by the following rules:

$$\sigma \parallel 0 \equiv \sigma, \sigma \parallel \sigma' \equiv \sigma' \parallel \sigma, \sigma \parallel (\sigma' \parallel \sigma'') \equiv (\sigma \parallel \sigma') \parallel \sigma''$$

Thus, each state  $\sigma$  is defined by the parallel composition of elements  $\delta$  and interactions  $l$ . The  $l$  is used to represent the pending actions to be executed and pending completions waiting to be perceived.

The state of organization is changed when an interaction occurs and is modeled by the transition relation  $\rightarrow \subseteq \Sigma \times \Sigma$ , representing the fact that a state  $\sigma$  may eventually move to another  $\sigma'$ , when a new pending action has to be computed which typically causes a change in the data reserved into the organization.

The meta-variable  $\gamma$  ranges over predicate logics to define the organizational rules using propositional logic. It is defined by the syntax as follows:

$$\gamma ::= a \mid \neg \gamma \mid \gamma \wedge \gamma \mid \gamma \vee \gamma \mid \gamma \Rightarrow \gamma$$

Where “ $a$ ” is the set of propositions exist within the organization. These rules usually define the pre-conditions required for the interactions between agents, or an agent and the organization.

The coordination behavior of organization is described by a transition system  $\langle C, \rightarrow, L \cup \{\tau\} \rangle$ .  $C$  is the set of configurations of the organization, which is defined by the composition of  $\rho$  and  $\sigma$  shown by  $\rho \otimes \sigma$ , where the function  $\rho$  associate to each agent the instruction it currently has to follow, and the  $\sigma$  defines the current state of the organization. The transition function  $\rightarrow \subseteq C \times L \times C$  is defined by the following rules:

$$\frac{\rho(id) \xrightarrow{!a} I}{\rho \otimes \sigma \xrightarrow{id!a} \rho[id \ \alpha \ I] \otimes \sigma \parallel id!a} \quad \text{Rule 1}$$

$$\frac{\rho(id) \xrightarrow{?b} I}{\rho \otimes \sigma \parallel id?b \xrightarrow{id?b} \rho[id \ \alpha \ I] \otimes \sigma} \quad \text{Rule 2}$$

$$\frac{\sigma \xrightarrow{\tau} \sigma'}{\rho \otimes \sigma \xrightarrow{\tau} \rho \otimes \sigma'} \quad \text{Rule 3}$$

The first rule defines an agent  $id$  can do/initiate an action, if the  $\rho(id)$  allows this action and then this action will reified in the state  $\sigma$ . The second rule defines the completion  $\beta$  to action  $\alpha$ ; if this is reified into the state  $\sigma$  and the  $\rho(id)$  allows perception of the completion. The third rule is derived from the actual coordination task inside the organization; when the  $\rightarrow \sigma$  defines changes in the states of the organization, there is a silent change in the system configurations, which is shown by a silent transition ( $\tau$ ).

In the next Section, we will specify a conference management system using our formal model.

## 4. Example

In this section we specify the conference management system as defined in [2, 8]. The conference management system is an open multi-agent system supporting the management of various sized international conference that require the coordination of several individuals and groups. There are four distinct phases in which the system must operate: submission, review, decision, and final paper collection. During the submission phase, author should be notified of paper receipt and given a paper submission number. After the deadline for submission has passed, the program committee (PC) has to review the papers. The program committee chair (PCC) will partition papers between the program committee members (PCM) to review the papers. They will review the papers on their own or assign them to the referees they have contract with. The PCC receives the reviews and notify the authors about the acceptance of the paper.

This organization can be viewed as being made up of agents associated to the persons involved in the process (author, PC Chair, PC Members and Referees). Thus the agents involve in the conference management system are author, PC chair and PC members and referees. The roles played by the agents to satisfy the conference system goal are the *partitioner*, *assigner*, *reviewer*, *collector*, *decision maker*, *author*, *data collector* [2]. The PCC can play the partitioner, collector and decision maker roles and The PCM can play the assigner and reviewer roles. The author can play the author and reviewer roles and finally a referee can play the reviewer role. The data collector role can be implemented by a database management system.

The agent enacting the partitioner role (PCC) delegates papers to the agent enacting role assigner (PCM) who will assign them to the reviewers (referees). Thus, there is a multi level delegation structure between agents. As explained in Section 2, an inform structure is needed to provide agents to be informed about their tasks. The agents playing roles may require interacting both directly with each other and indirectly, via an environment composed of papers and review forms.

Thus there should be a inform relation between data collector role and other roles to provide them the required data and coordinate them indirectly. Thus the organizational structure is defined as follows:

$$\psi ::= \{power(partitioner, assigner), power(assigner, reviewer), inform(partitioner, assigner), inform(assigner, reviewer), inform(decision maker, author), inform(data collector, partitioner), inform(data collector, reviewer), inform(reviewer, data collector), inform(data collector, decision maker)\}$$

The set of actions allowed by the system to the roles are defined as follows:

$$\alpha ::= get\_abstract(pn) | send\_assign(set(pn), id) | send\_part(set(pn), id) | get\_part | get\_assign | get\_reviews | submit\_result(result, pn) | send\_reviews(pn, set(review)) | get\_paper(pn) | submit\_review(review, pn) | submit\_paper(paper) | get\_result(result)$$

The “pn” parameter defines the number of a paper. The “set(pn)” parameter defines a set of paper number. The set of completion, resulted form an action are defined as follows:

$$\beta ::= receive\_abstract(pn, abstract) | OK_{send-assign} | OK_{send-part} | receive\_part(set(pn)) | receive\_assign(set(pn)) | receive\_reviews(set(review)) | OK_{submit-result} | OK_{send-review} | receive\_paper(paper) | OK_{submit-review} | OK_{submit-paper}(pn) | receive\_result$$

The interaction protocol for each role is defined as follows:

$$partitioner := (!get\_abstract(pn); ?receive\_abstract(pn, abstract)) + (send\_part(set(pn), id_{assigner}); ?ok_{send-part}); partitioner.$$

$$assigner := (!get\_part; ?receive\_part(set(pn))) + (!send\_assign(set(pn), id_{reviewer}); OK_{send-assign}) + 0; assigner.$$

$$reviewer := (!get\_assign; ?receive\_assign(set(pn))) + (!get\_paper(pn); ?receive\_paper(paper)) + (!submit\_review(review, pn); ?OK_{submit-review}) + 0; reviewer$$

$$decisionMaker := get\_reviews(pn); ?receive\_reviews(set(review)); submit\_result(result, pn); ?OK_{submit-result}; decisionMaker + 0.$$

$$data collector := (!send\_reviews(pn, set(review)); ?OK_{send-reviews}) + (!get\_review; ?recieve\_review(pn, review)) + (!send\_paper(pn, paper); ?OK_{send-paper}) + (!send\_abstract(pn, abstract); OK_{send-abstract}) + (!get\_submit; ?receive\_submit(paper)) || data collector$$

$$Author := !submit\_paper(paper); ?OK_{submit-paper}(pn); !get\_result; ? receive\_result(result).$$

The interaction protocol of the partitioner indicates that it can receive the abstract of a paper defined by its number or assign a set of paper numbers to an assigner. The data collector interaction protocol indicates that it provides services as providing the reviews of a paper to the decision maker or receiving paper reviews and papers. The data that reified into the system to track the coordination is defined as follows:

$$\delta := 0 | Author(id_{Author})/pn | Assigner:t | reviewer_{id:t}$$

$$review_{pn:r} \\ t := 0 | pn | t.t \quad t \in integer \\ r := 0 | review | r.r \quad r \in text$$

The Author(id<sub>Author</sub>)/pn keeps tracks of paper number assigned to the authors (It is considered that only one paper will be submitted by an author). The Assigner:t keeps track of paper numbers that an assigner has received and the reviewer<sub>id:t</sub> keeps track of paper numbers assigned to a review. The review<sub>pn:r</sub> keeps track of reviews for a paper having the identifier pn.

The state changes of organization are defined by the changes in the data reified into the system and the completion of a pending action done by an agent. The transition relation is defined as follows:

$$Assigner_{id:t} || id_{assigner} ! get\_part || id_{partitioner} ! send\_part(set(pn), id_{assigner}) \rightarrow Assigner_{id:t}.set(pn) || id_{partitioner} ? OK_{send-part} || id_{assigner} ? receive\_part(set(pn))$$

$$Assigner_{id:t}.set(pn) || reviewer_{id:t} || id_{assigner} ! send\_assign(set(pn), id_{reviewer}) || id_{reviewer} ! get\_assign \rightarrow Assigner_{id:t} || reviewer_{id:t}.set(pn) || id_{reviewer} ? receive\_assign(set(pn)) || ?ok_{send-assign}$$

$$review_{pn:r} || reviewer_{id:t}.pn.t' || id_{reviewer} ! submit\_review(pn, review) || id_{collector} ! get\_review \rightarrow review_{pn:r}.review || reviewer_{id:t} || id_{reviewer} ? OK_{send-review} || id_{collector} ? receive\_review(pn, review)$$

$$Author(id_{author})/pn || id_{decisionMaker} ! Submit\_result(pn, result) \rightarrow Author(id_{author})/pn || id_{decisionMaker} ? OK_{submit-result} || id_{Author} ? recieve\_result(result)$$

$$id_{Author} ! submit\_paper(paper) || id_{DBPaper} ! get\_submit \rightarrow Author(id_{author})/pn || id_{Author} ! ? OK_{submit-paper}(pn) || id_{DBPaper} ? receive\_submit(paper)$$

$$review_{pn:r}.set(review) || id_{decisionMaker} ! get\_reviews(pn) \rightarrow review_{pn:r}.set(review) || id_{decisionMaker} ? receive\_reviews(set(review))$$

$$id_{partitioner} ! get\_abstract(pn) || id_{DBPaper} ! send\_abstract(pn, abstract) \rightarrow id_{partitioner} ? receive\_abstract(pn, paper) || id_{DBPaper} ? OK_{send-abstract}$$

The first rule explains that an assigner receives a set of paper numbers in synchronous with the partitioner, and then this set will be added to the assigner set. The second rule explains that an assigner assigns a set of paper numbers in synchronous with reviewer. This set is omitted from the assigner set and added to the reviewer set. The organizational rules are defined as follows:

$$\forall t' \in reviewer_{id:t}.t_1.t_2 \wedge t' \neq pn \Rightarrow send\_assign(pn, id_{reviewer})$$

$$\forall id_{Author} Author(id_{Author})/pn \wedge \neg reviewer_{id_{Author}:t}.pn.t'$$

$$\forall pn, review_{pn:r}.NumberOf(r) \geq 2 \Rightarrow submit\_result(result, pn)$$

The first rule indicates that when a paper has been assigned to a reviewer, it should not be assigned him again. The second rule indicates that no author can review its own paper and the third rule indicates that decision maker can submit the result for a paper if at

$$\begin{array}{c}
\frac{\rho(\text{Assigner}) \xrightarrow{!get\_part} \rightarrow_I I}{\rho \otimes \sigma \xrightarrow{\text{Assigner}!get\_part} \rho[\text{Assigner} \alpha I] \otimes \sigma \| \text{Assigner}!get\_part}} \\
\frac{\rho(\text{Partitioner}) \xrightarrow{!send\_part(set < pn >, \text{Assigner})} \rightarrow_I I}{\rho \otimes \sigma \xrightarrow{\text{Partitioner}!send\_part(set < pn >, \text{Assigner})} \rho[\text{Partitioner} \alpha I] \otimes \sigma \| \text{Partitioner}!send\_part(set < pn >, \text{Assigner})}} \\
\frac{\sigma \rightarrow_{\sigma} \sigma''(\text{rule1})}{\rho \otimes \sigma \xrightarrow{\tau} \rho \otimes \sigma'} \\
\frac{\rho(\text{Assigner}) \xrightarrow{?receive\_part(set < pn >)} \rightarrow_I I}{\rho \otimes \sigma \| \text{Assigner}?receive\_part(set < pn >)} \xrightarrow{\text{Assigner}?receive\_part(set < pn >)} \rho[\text{Assigner} \alpha I] \otimes \sigma \\
\frac{\rho(\text{Partitioner}) \xrightarrow{?OK\_send\_part} \rightarrow_I I}{\rho \otimes \sigma \| \text{Partitioner}?OK\_send\_part} \xrightarrow{\text{Partitioner}?OK\_send\_part} \rho[\text{Partitioner} \alpha I] \otimes \sigma
\end{array}$$

**Figure1. The coordination behavior of conference management system for assignment of a partition to the assigner**

least two reviews exist. The first and second rules put some constraints over the communication between assigner and a reviewer. In other word an assigner should assign a paper in synchronous with reviewer and this paper should not be assigned before and the reviewer should not be the author of the paper. The third rule defines the pre-condition (constraint) for a decision maker.

The coordination behavior of the organization is defined according to the rules explained in Section 3. The coordination behavior of organization when a partitioner assigns a set of paper numbers to an assigner is shown in Figure 3. The assigner is allowed to receive a partition if its interaction instruction indicates that it can do this action. When the assigner does the action to receive a partition, it will be reified and pending in the organization. On the hand, the partitioner can send a partition to an assigner if its interaction protocol indicates that it can do this action. When the partitioner sends a partition to an assigner, this action remains pending in the organization. According to the first rule, there is a silent changes in the system configuration and the partitioner assigns a partition in synchronous with the assigner. Now both agents (roles) can receive the completion of their actions.

As explained in Section 3, our formal specification enables us to have a cross checking between the organizational structure, the interaction protocol and the coordination behavior of the organization. For example the power structure between the partitioner and the assigner enables the partitioner to delegate a partition to the assigner and the assigner to receive this partition. Thus the act of sending and receiving the partition is admissible in the partitioner and the assigner interaction protocol. The coordination behavior of the organization coordinates the interaction of the partitioner and the assigner.

We can also use the formal model to evaluate system properties. For example we can use the inform relation between agents to evaluate the flow of information between agents and investigate the data security.

## 5. Conclusion

In this paper, we propose a formal model for the specification of multi-agent organizations. We have extended the formal model for coordination artifacts with organizational concepts in [7]. This specification formally defines what tasks an agent is allowed to do in an organization and when it is allowed to do and in synchronization with which actions in the system. It also defines what the pre-condition of each task is and how the organizational structure affects on the interactions between agent and the coordination behavior of the organization. Our formal specification capable developers to evaluate the behavior of organization in terms of different organizational structure (there should be shared data, if there is no direct communication and etc.).

## 6. Acknowledgment

Special thanks to Mehdi Dastani for his comments on organization concepts.

## 7. Reference

- [1] F. Zambonelli, N. Jennings and M. Wooldridge, Organizational Abstractions for the Analysis and Design of Multi-Agent System. In *First International Workshop on Agent-Oriented Software Engineering at ICSE*, 2000
- [2] S. DeLoach, Analysis and Design of Multi-Agent Systems Using Hybrid Coordination Media, In *proceeding of Software Engineering in Multiagent Systems*, July 2002
- [3] S. DeLoach and E. Maston, An Organization Model for Designing Adaptive Multiagent Systems, The AAAI-04 Workshop on Agent Organizations: Theory and Practice (AOTP 2004). Technical Report WS-04-02. AAAI Press. , San Jose, California, July 25-29, 2004, pp. 66-73.
- [4] M. Dastani, F. Arbab and F. de Boer, Coordination and Composition in Multi-Agent Systems, In *4rd International Joint Conference on Autonomous Agents and Multiagent*

*Systems* (AAMAS 2005), Utrecht, The Netherlands, July 25-29, 2005.

[5] B. Horling and V. Lesser, A Survey of Multi-Agent Organizational Paradigms, *The Knowledge Engineering Review*, Volume 19, Issue 04, December 2004, pp 281-316

[6] D. Grossi, F. Dignum and M. Dastani, Foundations of Organizational Structures in Multiagent Systems, In *4rd International Joint Conference on Autonomous Agents and Multiagent Systems*, 2005, pp 690–697.

[7] A. Omicini, A. Ricci and M. Viroli, Coordination Artifacts: Environments-based Coordination for Intelligent Agent. In N. R. Jennings, C. Sierra, L. Sonenberg, and M. Tambo, editors, *3rd international Joint Conference on Autonomous Agents and*

*Multiagent Systems*, ACM, New York, USA, 2004, volume 1, pages 286-293.

[8] F. Zambonelli, N. Jennings and M. Wooldridge, Organizational Rules as an abstraction for the Analysis and Design of Multi-Agent Systems, *International Journal of Software Engineering and Knowledge Engineering*, volume 11, Number 3, June 2001, Pages 303-328.

[9] M. Wooldridge, N.R Jennings, “The Gaia Methodology for agent-oriented analysis and design”, *Journal of Autonomous Agents and Multi-Agent Systems*, 3(3):285-312, 2000

[10] P. Giorgini and M. Kolp and J. Mylopoulos and M. Pistore, The Tropos Methodology: An Overview. In *Methodologies and Software Engineering for Agent Systems*, Kluwer, 2004.