

A Formal Model for Organization of Multi-agent Systems

Fatemeh Ghassemi

Naser Nemat Bakhsh

Behrouz Tork Ladani

Isfahan University, Isfahan, Iran
{ fghassemi, Nemat, ladani }@eng.ui.ac.ir

Marjan Sirjani

Tehran University, Tehran, Iran
msirjani@ut.ac.ir

Abstract Multi-agent systems are rapidly used by IT developers as a solution for complex and distributed systems. Applying organizational concepts in analysis and design of such systems helps IT developers to reduce the complexity and makes the development of such systems easier. In this paper, we propose a formal model for the specification of multi-agent organization. The model helps us not only to reduce the ambiguity in the system specification but also to evaluate the system behavior (e.g. performance, security, data flow, etc) according to the organizational structure. Furthermore to demonstrate the applicability of the proposed model, we describe an example system and specify its organization formally using the proposed method.

Keywords: Organization, Formal Methods, Organizational Rule, Organizational Structure, Coordination Artifact

1. INTRODUCTION

Autonomous agents and multi-agent systems (MASs) are widely used by IT developers to design complex and distributed systems such as e-learning and e-marketing systems, electronic auctions and business-to-business applications. An agent provides a behavior abstraction which allows the IT developers to naturally model and construct complex systems [4, 5].

The field of multi-agent systems has long been interested in using social and biological concepts to analyze and design such systems. 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 the architecture of a multi-agent system can naturally be viewed as a computational organization which consists of a multitude of autonomous interacting entities (an organized society of individuals) in which each agent plays one (or more) specific roles [1, 2, 3]. However, the organization of a multi-agent system is distinct from the individual agents that populate the system [1, 10]. While agents play roles within the organization, their roles do not constitute the organization. Roughly speaking organizations are characterized by the organizational structures 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]. Thus we can separate organizational responsibility from agent responsibilities. Agent's responsibility is to accomplish their assigned tasks and the organization responsibility is to coordinate agent's tasks in the context of organization.

The organizational structure of a multi-agent system coordinates the interaction of agents and determines the overall behavior of the multi-agent systems [1, 5]. Thus an organization can be viewed as a coordination artifact, which coordinates the behavior and interaction of self-interested agents in a MAS.

In spite of strengths and powers in the agent-based approaches, there are lots of pitfalls in the agent-based system design which leads to the failure of systems from the beginning [13]. Some of pitfalls can be addressed by applying formal methods in the system specification to minimize the ambiguities and inconsistencies. The formal specification leads developers to the proper analysis and design of the system.

In this paper, we propose a formal model for describing the organization of multi-agent systems. It enables us to evaluate the properties of the organization according to its specification. For example, it has been repeatedly shown that the organization of a system can have a significant impact on its short and long-term performance [7]. In general, using a formal description for the organization helps us in evaluation of the overall system's performance, security, flow of information, etc.

Related works: A formal model for organization has been provided in [3] which captures the notions central to agent-oriented software engineering, such as goals, roles and agents. This model provides the capability for designing adaptive systems according to the internal and external changes, but it doesn't consider agents as abstract entities to show their interactions and their coordination behavior. A multi-modal logic proposed in [6] to describe the structure of organization. We have applied the result of this method to describe the structure of an organization and the impact of such organizational structure on the design of a system.

There are several formal methods to specify MASs such as DIOA [14], Agent UML [15] but these methods do not use social concepts (e.g. organizational concepts) for system description. In Section 2, we will explain why using organizational concepts helps us in analysis and design of such systems.

Structure of the paper: The organization metaphor is described in Section 2. In Section 3, we explain our formal model for an organization. Section 4, describes an example system and specifies its organization using our formal model and finally Section 5 explains our concluding remarks and future works.

2. ORGANIZATION

In the traditional design of concurrent and distributed systems, the architecture derives from the decomposition functionalities and data required by the system to achieve its goals as well as the definition of their inter-dependencies [12]. However, using organizational concepts to design such systems, leads to a number of agents, each with specific roles in the system. In this model, interactions are no longer an expression of inter-dependencies, rather they are viewed as a means for an agent to accomplish its role in the organization. In this model, each agent is a locus of control, in charge of accomplishing its role and being responsible for that. Thus agents embed most of the functionalities they need to accomplish their roles. Thus the interactions of agents are reduced which makes the design less complex and easier to manage.

On the other hand, most MASs are intended to support or control some real-world organizations. In such cases, an organizational-based MAS design reduces the conceptual distance between the software system and the real world system it has to support. Thus applying organizational concepts in analysis and design of MASs, reduces the complexity and makes the development easier to manage.

The Organization of multi-agent system is the collection of roles, relationships and authority structures which governs its behavior. All multi-agent systems possess some form of organization. Just like human organizations, it guides members how to interact with one another over the long-term course of a particular goal or set of goals [7]. Agent-oriented methodologies such as Gaia [16], use a *role model* to show roles within the organization and the interaction protocols among them. Relationship and authority structures define the position of agent in an organization. In other words, they define the topology or organizational structure of the system. An organizational structure defines the specific class (among the many possibilities) of organization and control regime to which the agents/roles have to conform in order for the whole MAS to work efficiently and according to its specified requirement [1]. The shape, size and characteristics of the organizational structure can affect the behavior of the system [7]. These 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 [5]. In [6], each organizational structure is seen along three dimensions: power, coordination, and control. Power defines

the delegation of task between two roles. Coordination defines the flow of information between two roles and the control relation defines the supervisory relation between two roles. We have applied the result achieved in [6]. When there is a power relationship between two agents, it means the first agent delegates its task to second agent which requires a communication path (directly or indirectly) between two agents that enables the first agent to delegate its task to second one. When there is an inform relationship between two agents, it requires a direct communication link between agents to allow the flow of information between two agents and when there is a control relationship between two agents, there must be a communication link between the output ports of first (controlled) agent to the second one (controller). This organizational structure also affects the activities of agents. For example when an agent informs some new information to another agent, if there is no inform relationship between these two agents, then the information does not necessarily create any knowledge in the recipient agent. Since we considered agents as a block-box we don't consider these effects on the agent activities and on the internal structure of agent such as belief, goals, knowledge and intentions.

Organizational rules express general, global (supra-role) requirement for the proper instantiation and execution of a MAS [1, 2]. These organizational rules express the conventional rules, within an organization.

Therefore, to formally describe an organization, we should define, what agents constitute the organization and how they interact to each other and we should define how organization coordinates the interaction of agents and what is the behavior of organization when an agent performs an action in the organization. We should also define the affect of organizational structure and rules on the behavior of organization.

3. FORMAL SPECIFICATION

As described in Section 1, an organization coordinates agent interactions. Thus organization can be viewed as a coordination artifact that coordinates the behavior and interaction of agents in terms of long-term goals of system. We consider organization as an open system, where agents are self-interested and can enter and leave organization. The organizational rules define when they can enter and leave the organization and organizational structure defines the position of agent in the organization.

A coordination artifact is generally characterized by [9]:

- a *usage interface*, defined in terms of a set of operations. Agents execute actions on the artifact, by specifying the artifact operations involved and perceive information about the completion of such action.
- A *set of operating instructions*, which defines how to use the artifact to exploit its coordination service.
- A *coordination behavior specification* that describes the coordination behavior of the artifact, in terms of coordination rules required for enacting the coordination service.

We have extended the formal model for coordination artifact with organization concepts to specify an organization. The

usage interface defines the tasks an agent is allowed to do in an organization and the set of operating instructions is the interaction protocol between agent and organization and, coordination behavior specifies the behavior of organization that how it coordinates the interaction of agents.

An organization is defined as a tuple $\langle \alpha, \beta, \rho, \delta, \rightarrow_{\sigma}, \gamma, \psi, A \rangle$. α is a meta-variable ranging over the operations allowed by the organization, namely, the actions the agent can execute in it. β is the meta-variable ranging over perceptions of action completion, which may possibly contain some information about the outcome of the action. Correspondingly, the set L of interaction between agents and the organization, ranged over by l , is defined by syntax

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

where $id! \alpha$ represents agent id executing action α , and $id? \alpha \beta$ represents agent id perceiving the completion β to action α .

ρ is a function associating to each agent identifier id the usage instruction I he has to follow. Instructions can be defined by exploiting typical process algebra operators, e.g. by the syntax [8]:

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

Here, 0 is a void instruction, $! \alpha$ is execution of an action, $? \beta$ is perception of a completion, operator “+” is used for choice between instructions, “;” for sequential composition of instructions, “||” for parallel composition of instructions, and D is invocation of a recursive definition. As an example, the definition $I := ! \alpha ; (? \beta || I)$ means that the agent is initially allowed to execute action α and later, while it can do the whole instruction again, doing another action of α , it can perceive the completion of previous actions of α .

Meta-variable δ ranges over the data reified into the organization (like databases or temporary containers) to keep the track of the state of the coordination task. This helps us to describe the environments-based coordination. There is not always a communication path¹ between two agents. Thus to coordinate their behaviors they reify a data which then taken by another agent. The communication links are defined (and affected) by an organization structure. The meta-variable σ ranges over the set of Σ of states of the coordination artifact, which is defined as:

$$\Sigma ::= 0 \mid \delta \mid l \mid (\sigma || \sigma')$$

Operator $||$ is characterized by the following congruence rules:

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

Thus, elements σ are easily understood as parallel composition of elements δ and interactions l . The l is used to

¹ A communication link specifies a direct and point-to-point communication. A communication path is a sequence of communication links between two agents to describe a direct or indirect communication.

represent the pending actions to be executed and pending completions waiting to be perceived.

The state changes as interactions occur: this dynamics is modeled by the transition relation $\rightarrow_{\sigma} \subseteq \Sigma \times \Sigma$, representing the fact that a state σ may eventually move to another σ' , which typically happens when a new pending action has to be computed.

The meta-variable γ ranges over first order predicates to define the organizational rules using propositional logic and has the syntax:

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

Where “ a ” is the set of atomic propositions exist in the organization.

ψ is the set of binary relations, which defines the organization structure of MAS in three dimensions of control, power and coordination:

$$\psi ::= \{ \text{power}(r,s), \text{control}(r,s), \text{coord}(r,s), r,s \in A \}$$

Where “ A ” is the set of agent's identifiers playing role in the organization.

So, while α and β define the shape of interactions allowed by the organization, ρ defines the protocols allowed to the agents, while δ and \rightarrow_{σ} define the actual coordination task. The coordination behavior of organization is described by a transition system $\langle C, \rightarrow, L \rangle$. C is the set of configuration of organization, which are of the kind $\rho \otimes \sigma$ namely, the composition of a function ρ associating to each agent the instruction it currently has to follow, and the current state of data reified into the organization.

Note that the γ and ψ impose constraints on the actual coordination of organization. ψ defines if there is a communication path between two agent or they should coordinate their behavior through environment (data reified into the environment) and γ imposes some pre-conditions for state transitions.

4. EXAMPLE

In this section we specify the conference management system as defined in [11]. 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. After the reviews are complete, a decision on accepting or rejecting each paper must be made after which authors are notified of the decisions and are asked to produce the final version if their paper was accepted.

The conference management system consists of an organization whose members may change at each stage of the process. This organization can be viewed as being made

up of agents associated to the persons involved in the process (author, PC Chair, PC Members). Thus the agents involve in the conference management system are author, PC chair and PC members. The roles played by each agent reflect the ones played by the associated person in the conference system. They may require agents to interact both directly with each other and indirectly, via an environment composed of papers and review forms. Since an agent is directly associated with a person, and its behavior can be influenced by that person, opportunistic behavior can emerge in the application. For example, an author could attempt to review their own paper. We use abbreviations PCM and PCC to denote PC member and PC chair.

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

$\alpha := \text{get_abstract} \mid \text{send_assign} \mid \text{get_review} \mid \text{submit_result} \mid \text{get_assign} \mid \text{get_paper} \mid \text{submit_review} \mid \text{submit_paper} \mid \text{get_result}$

The set of completion, resulted form an action are defined as follows:

$\beta := \text{receive_abstract} \mid \text{ok_assign} \mid \text{receive_review} \mid \text{ok_submit_result} \mid \text{receive_assign} \mid \text{receive_paper} \mid \text{ok_submit_review} \mid \text{ok_submit_paper}(\text{pn}) \mid \text{receive_result}$

The interaction protocol for each agent is defined as below:

$\text{PCC} := (!\text{get_abstract}(\text{pn}); ?\text{receive_abstract}(\text{abstract})); (\text{send_assign}(\text{pn}, \text{id}_{\text{PCM}}); ?\text{ok_assign} \parallel \text{PCC}) + (\text{get_review}(\text{pn}); ?\text{receive_review}(\text{review})); (\text{submit_result}(\text{result}, \text{pn}); ?\text{ok_submit_result} \parallel \text{PCC})$
 $\text{PCM} := !\text{get_assign}; ?\text{receive_assign}(\text{pn}); !\text{get_paper}(\text{pn}); ?\text{receive_paper}(\text{paper}); !\text{submit_review}(\text{review}, \text{pn}); ?\text{ok_submit_review} \parallel \text{PCM}$
 $\text{Author} := !\text{submit_paper}(\text{paper}); ?\text{ok_submit_paper}(\text{pn}); !\text{get_result}; ?\text{receive_result}(\text{result})$

The data that reified into the system to track the coordination is defined follows:

$\delta := 0 \mid \text{Author}(\text{id}_{\text{Author}})/\text{pn} \mid \text{review}/\text{pn} \mid \text{result}/\text{pn} \mid \text{PCC}:t \mid \text{PCM}_{\text{id}:t} \mid \text{Review}_{\text{pn}:r}$
 $t := 0 \mid \text{pn} \mid t.t$
 $r := 0 \mid \text{review} \mid r.r$

The PCC:t keeps track of paper numbers that PC chair has received their abstract and can assign them to any PC member. Each PC member keep s track of the papers has been assigned to him. Each paper identified by a “pn”, will have several reviews by different reviewers. So system keeps track of reviews for a paper by data “Review_{pn}:r”.

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. For example, when an author submits a paper into the system, a data which binds paper to a “pn” is reified into the system and the completion of action received by author while it contains the “pn” assigned to the paper (rule 9 below).

- 1) $\text{PCC}:t \parallel \text{id}_{\text{PCC}}!\text{read_abstract}(\text{pn}) \rightarrow \text{PCC}:t.\text{pn} \parallel \text{id}_{\text{PCC}}?\text{receive_abstract}(\text{abstract})$
- 2) $\text{PCC}:t.\text{pn}.t' \parallel \text{id}_{\text{PCC}}!\text{send_assign}(\text{id}_{\text{PCM}}, \text{pn}) \rightarrow \text{PCC}:t.\text{pn}.t' \parallel \text{id}_{\text{PCM}}?\text{ok_assign}$
- 3) $\text{Review}_{\text{pn}:t} \parallel \text{review}/\text{pn} \parallel \text{id}_{\text{PCC}}!\text{get_review}(\text{pn}) \rightarrow \text{Review}_{\text{pn}:t}.\text{review} \parallel \text{review}/\text{pn} \parallel \text{id}_{\text{PCC}}?\text{receiev_review}(\text{review})$
- 4) $\text{id}_{\text{PCC}}?\text{submit_result}(\text{result}, \text{pn}) \rightarrow \text{result}/\text{pn} \parallel \text{id}_{\text{PCC}}?\text{ok_submit_result}$
- 5) $\text{PCM}_{\text{id}:t} \parallel \text{id}_{\text{PCM}}!\text{get_assign} \rightarrow \text{PCM}_{\text{id}:t}.\text{pn} \parallel \text{id}_{\text{PCM}}?\text{receive_assign}(\text{pn})$
- 6) $\text{PCM}_{\text{id}:t}.\text{pn}.t' \parallel \text{id}_{\text{PCM}}!\text{get_paper}(\text{pn}) \rightarrow \text{PCM}_{\text{id}:t}.\text{pn}.t' \parallel \text{id}_{\text{PCM}}?\text{receive_paper}(\text{paper})$
- 7) $\text{PCM}_{\text{id}:t}.\text{pn}.t' \parallel \text{id}_{\text{PCM}}!\text{submit_review}(\text{review}, \text{pn}) \rightarrow \text{PCM}_{\text{id}:t}.\text{pn}.t' \parallel \text{review}/\text{pn} \parallel \text{id}_{\text{PCM}}?\text{ok_submit_review}$
- 8) $\text{id}_{\text{Author}}!\text{submit_paper}(\text{paper}) \rightarrow \text{Author}(\text{id}_{\text{Author}})/\text{pn} \parallel \text{id}_{\text{Author}}?\text{ok_submit_paper}(\text{pn})$
- 9) $\text{Author}(\text{id}_{\text{Author}})/\text{pn} \parallel \text{result}/\text{pn} \parallel \text{id}_{\text{Author}}!\text{get_result} \rightarrow \text{Author}(\text{id}_{\text{Author}})/\text{pn} \parallel \text{result}/\text{pn} \parallel \text{id}_{\text{Author}}?\text{receive_result}(\text{result})$

The organizational rules are as follows:

- 1) $\forall \text{pn}, \#(\text{reviewer}(\text{pn})) \geq 3$
- 2) $\forall \text{id}_{\text{PCM}}, \text{pn}, \forall t' : \text{PCM}_{\text{id}:t}.t' \wedge t' \neq \text{pn} \wedge \text{send_assign}(\text{pn}, \text{id}_{\text{PCM}}) \Rightarrow \text{PCM}_{\text{id}:t}.\text{pn}$
- 3) $\forall \text{id}_{\text{Author}}, \text{pn}, \text{Author}(\text{id}_{\text{Author}})/\text{pn} \wedge \neg \text{PCM}_{\text{id}_{\text{Author}}:t}.\text{pn}.t'$
- 4) $\forall \text{pn}, \#(\text{review}/\text{pn}) > 2 \wedge \text{submit_result}(\text{result}, \text{pn}) \Rightarrow \text{result}/\text{pn}$

The first rule indicates that the number of reviewers assigned to a paper should not be less than three and the second rule indicates that when a paper has been assigned to reviewer, it should not be assigned again. The third rule indicates that no author can review their own papers and the forth rule indicates when the PCC submits a result; there must be more than two reviews in the system. Rule 2 and 3 are the pre-conditions for the fifth rule of \rightarrow_{σ} . The forth rule is a pre-condition for the forth rule of \rightarrow_{σ} . The first rule defines a monitoring process in the organization that should force PC chair to assign more than two reviewers for a paper.

The organizational structure of system is a two level structure; the PC chair assigns task to each PC members. Thus the organizational structure is defined as follows:

$$\psi = \{\text{power}(\text{PCC}, \text{PCM})\}$$

This structure indicates that PC chair directly assigns a paper to a PC member and there is a communication path between these agents. If there was not such a structure, PC chair should reify into the system the assignment, (for example we need “id_{PCM}/pn”) and the PC chair gets its task from the data reified into the system. Thus PC chair and PC member are coordinated via environment. However, by having such structure, in rule 2 and 5 of \rightarrow_{σ} , there is no need to reify the assignment. Therefore, organizational rules and organizational structure impose constraint on agent interactions and organization behavior. The behavior of organization is defined by the composition of ρ and σ . For

example, when PC chair assigns a paper to a PC member, The PC member should receive the assignment. Thus the ρ function defines what tasks done by agents should be synchronized in the system. Thus the ρ function defines what tasks done by agents should be synchronized in the system. It is defined by the following rule:

```
PCMid:t || PCC:t.pn.t' || idPCC! Send_assign(pn, idPCM) ||
idPCM!get_assign → PCMid:t.pn || PCC:t.pn.t' || idPCC?okassign
|| idPCM? Receive_assign(pn)
```

5. CONCLUSIONS

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 [9]. 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 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.).

Since agents in an organization are autonomous, they can coordinated exogenously using an exogenous coordination language like Reo [10, 17]. Reo has an operational semantic which can be used to evaluate the implementation of organization according to its specification. We will apply the formal specification of organization to implement systems using Reo language in next paper.

6. REFERENCES

- [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*, Orlando, Florida, 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. pp. 66-73. July 25-29, 2004, San Jose, California.
- [4] "Why, When, and Where to Use Software Agents", agentbulider.com, last visited 2004
- [5] 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)*, July 25-29, 2005, Utrecht, The Netherlands.
- [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 (AAMAS 2005)*, July 25-29, 2005, Utrecht, The Netherlands.
- [7] 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
- [8] M. Virko, A. Ricci and A. Omicini, A Semantic for the Interaction of Agents with Coordination Artifacts, Technical Report, 2003
- [9] 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, In *3rd international Joint Conference on Autonomous Agents and Multiagent Systems*, ACM, volume 1, pages 286-293, New York, USA, 2004.
- [10] G. Boella, L. van der Torre and H. Verhagan, Introduction to Normative Multiagent Systems. Technical Report, In *1st International Symposium on Normative Multiagent Systems*, 2005
- [11] 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.
- [12] G. Booch, *Object-Oriented Analysis and Design (Second Edition)*, Addison Wesley, Reading (MA), 1994.
- [13] M. Wooldridge, N.R Jennings, "Pitfalls of Agent-Oriented Development", In *Proceedings of the Second International Conference on Autonomous Agents*, ACM Press, 1998, pages 385-391.
- [14] P. Attie and N. Lynch, Dynamic Input/Output Automata: A Formal Model for Dynamic Systems, In *proceeding of Concurrency Theory, 12th International Conference*, 2001
- [15] B. Bauer, J. Müller and J. Odell, "An extension for UML by protocol for multiagent interaction", In *Proceeding of Fourth International Conference on Multi-Agent Systems (ICMAS-00)*, 2000
- [16] 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
- [17] F. Arbab. Reo: A channel-based coordination model for component composition. *Mathematical Structures in Computer Science*, 14(3)329-366, 2004.