

Organizational and Social Concepts in Agent Oriented Software Engineering

Xinjun Mao¹ and Eric Yu²

¹ Department of Computer Science, National University of Defense Technology, China

Email: xjmao21@21cn.com

² Department of Computer Science, University of Toronto, Canada

Email: eric.yu@utoronto.ca

Abstract. AOSE methodologies and models borrow various abstractions and concepts from the organization and sociology disciplines. Although they all view multi-agent system as organized society, the organizational abstractions, assumptions, concepts, and models in them are actually used in different ways. It is, therefore desirable to have a systematic way of analyzing and comparing the organizational and social concepts in AOSE. The contribution of this paper is twofold. Firstly, we describe and define the modeling construct levels and the social premises of multi-agent system that should be modeled and analyzed when developing multi-agent system, identify and classify categories of organizational and social concepts in AOSE literature that are used to deal with them from standpoints of organization abstractions. Secondly, we analyze some methodologies and models in AOSE, explain how the organizational and social concepts are used to specify and analyze multi-agent system with various social premises in different levels.

1. Introduction

Multi-agent systems (MAS for short) are rapidly emerging as a powerful paradigm for developing complex system. However, if we want the paradigm to be successfully applied in the development of complex system, and widespread deployment of MAS, the models, technologies and even the methodologies should be developed in order to support the developers to engineer such systems in a robust, reliable, and even repeatable fashion. Although the ideas of MAS paradigm are simple and natural, however it will become more difficult to develop the complex MAS if we lack the guidelines of the necessary technologies.

The researches in MAS literature have the following characteristic: they often draw on and integrate with the knowledge and experience from other disciplines such as psychology, economic, cognitive science, linguistics, artificial intelligence, etc. For example, we often investigate and analyze the interaction protocols and communication actions among agents based on the speech act theory, which actually comes from the linguistics; the abstraction tool of intentional stance has been borrowed from the cognitive science to reason and analyze the autonomous

behaviors of agents. Recently, many methodologies and models borrowing the abstractions and concepts from the organization and sociology disciplines have been put forward in order to understand, model, reason, analyze and design MAS. Although these models and methodologies all view multi-agent system as organized society, the organization abstractions, concepts, assumptions and models of them are actually different. In fact, different methodologies and models have different purposes and assumptions about organization, model the MAS in different level, use different organizational and social concepts, target for use at different stages in software engineering lifecycle, and therefore have different application domains. In addition, there are many organizational and social concepts that have been presented. Many concepts, although with different terms, actually have the similar meaning and purpose. The same concept may have different interpretations and definitions in various models and methodologies.

Although there are many papers to compare and evaluate the methodologies in agent oriented software engineering (AOSE for short) such as [23,24,26,27,28], there is few efforts to compare them from the standpoints of organizational and social abstractions and modeling, especially to analyze the organizational and social concepts in AOSE literature. For example, what is the relationship between the agent and organization? why and what the organizational and social concepts are needed to model MAS? what are the diverse social premises assumed in different MAS and how to deal with them by means of organizational and social concepts, etc. As the organizational and social abstractions have been gained great attentions in AOSE and play central roles in designing the AOSE methodologies, it is necessary to have a map of the research on organizational and social concepts in AOSE. The reminders of the paper are structured as follows. Section 2 describes and defines the modeling construct levels and the social premises of MAS, identify and classify the organizational and social concepts in AOSE literature and explain in detail how are used to specify and analyze the MAS with various social premises in different modeling construct levels. Section 3 analyzes a number of AOSE methodologies and models that are influential in AOSE, and explains what the organization assumptions about the social premises of MAS are made, what and how the organizational and social concepts are used, especial the meta-models of organizational and social concepts in each methodology or model are developed. At last, the conclusions and future work are made in section 4.

2. Organizational and Social Concepts

There are many organizational and social concepts in AOSE literature. They are various and used in different abstract levels and for different purposes. In this section, we will analyze the modeling construct levels and the social premises that we should deal with when developing MAS, identify and classify a number of organizational and social concepts in the AOSE literature and explain how they can be used to model the MASs with different social premises in various levels.

2.1 Modeling Construct Level

The MAS may be complicated: there are many agents with different goals and tasks running in the system, the structural and behavioral relationships between agents are extremely complex as they are involved in complicated interactions. However, a number of modeling construct level can be identified according to the system constructs that we have to deal with when developing MAS.

- **Single agent**

In this level, the autonomous behaviors of agents should be specified and analyzed in an abstract way. Generally, the functionalities and activities of agents are the most important aspects that should be modeled independent of the details. For example, what are the functionalities of agents? what the resources and/or activities should they have in order to accomplish their functionalities?, etc. The models describing the single agent are important constituents of the system requirement specification to guide the design of software agents.

- **Two agents**

Agents in MAS are not isolated from one to another. Two agents may have various relationships between them like structural ones and behavioral ones. For example, one agent depends on another agent to get the resources required to accomplish its tasks, or should explicitly interact with other agent by some interaction protocol (e.g., contract net) to acquire the resources or the assigned tasks; one agent may be the supervisor of another one and has the authorities to assign the tasks to it. The information about the relationships between agents should be explicitly specified and analyzed in support of the requirement specification and analysis and further guide the software architecture design.

- **Two or more agents acting in a coordinated way**

In some MASs, two or more agents are organized together as a group and act in a coordinated way in order to achieve some common purposes. Agents in one group are often cooperative and have some common goals and joint behaviors. For such MASs, it is necessary in the analysis and design phase to identify and define the groups in the system, specify them in detail about the structural information (e.g., how agents in the group are organized) and the behavioral information (e.g., what the common goals of agents in the group) of them.

- **All agents**

In this level, all agents in the system organize together as one organization. Therefore, the organization information should be specified and analyzed. For example, what is the organization structure of the system? Are there any global constraints in the organization that govern all agents in it, etc. To model such information is of particular importance for the specification and analysis of system requirement and further guide the software architectural design.

2.2 Social Premise

The organization abstractions view MAS as an organization, which can vary

greatly from system to system. For example, some systems are open, others may be closed; the agents in some systems are cooperative, however others may be self-interested. Therefore, a number of social premises about MAS organization can be identified, which, we think, are important to specify the organizational characteristics of MAS and require special concepts and models to specify and analyze when developing MAS with such social premises.

- **Open or closed**

The system is open if the system has no definite boundary, and allows the new, maybe unknown agents to enter or leave it from time to time in the lifecycle of the system. Therefore, the components (e.g., agents) in open system may change and can not completely be defined at the design-time. For instance, Internet is just one typical example of such a system. For the closed system, we mean one that has definite system components and they will not vary at run-time, therefore can be well-defined at design-time by the software developer. Obviously, the open system is unpredictable and much more difficult to develop than the closed one.

- **Dynamic or static**

The system is dynamic if the system elements, especially the abilities of agents in the system and the services they provide and/or the inter-agent relationships, can change at run-time. For example, the roles that agent plays may vary in different contexts and situations, and therefore the inter-agent relationships (e.g., the interactions and/or dependencies) may also change. For static system, all of the system elements are invariable. Apparently, the dynamic system is more complicated and difficult to develop than the static ones.

- **Cooperative or Self-interested**

The agents in some system may be cooperative in certain social context. They share some common goals and interact with each other in a cooperatively way to willingly provide resources and services, achieve the tasks or goals. Conversely, the self-interested agent does for itself, and may refuse to provide services or resources for other agents. In addition, conflicts are much easier to occur between self-interested agents in the systems, especially ones that have scarce resources to be shared by a number of agents. For example, the goals of self-interested agents may conflict with other or there are resource conflicts when a number of agents share common resources to achieve their goals. The systems with self-interested are more complicated than the ones with cooperative agents, and much more difficult to model and analyze.

- **Hierarchical**

Some system frequently takes the form of a hierarchy. That is, a system composed of interrelated sub-system, each of which is in turn hierarchic in structure, until the lowest level of elementary sub-system is reached [9]. Furthermore, there are various relationships between the sub-systems. A typical example is enterprise system, which is composed of a number of departments or factories that in turn can be decomposed as a number of sectors or working groups. However, not all systems are hierarchic. In contrast, the hierarchic systems evolve more quickly than non-hierarchic ones of comparable size, which makes it more difficult to deal with. However, the decomposition principle in

software engineering literature to partition such systems conduces to controlling the system complexity.

- **Global Constraint**

In some systems, there are global constraints (e.g., social laws) that should be respected by all agents in the system and will govern the relationships (e.g., interaction) among them. For example, the social law will constrain to some extents the behaviors of agents in the organization. The explicit identification of the global constraints is of particular importance in the context of open system with self-interested agents, which will simplify the system analysis and design.

The systems in the real world have various social premises mentioned above. It is obvious that the systems with such social premise as open, dynamic, self-interested, etc., are often more difficult to develop than ones that are closed, static, cooperative, etc. In AOSE literature, we can find, no methodologies and models are all-powerful. Various restrictive assumptions about the social premises of the systems are often made in different methodologies to idealize the systems in order to simplify the systems and methodologies to be developed. Therefore, different organizational and social concepts are developed and integrated in different methodologies and models in order to satisfy the social premises that they assume.

2.3 Modeling Concepts

Now we turn to analyzing what the social premises mean in different modeling construct levels, what the organizational and social concepts are required to model the MAS in these levels, and how they are used to specify and analyze the systems with various social premises. Although the organizational and social concepts are diverse in AOSE literature, a clear taxonomy of these concepts can be made according to their modeling purpose and the system construct level that they intend to deal with. In each category, the organizational and social concepts can be further divided into a number of groups. The concepts in each group often have similar semantics and modeling purpose.

2.3.1 Concepts for Modeling Single Agent

The organizational and social concepts in this level are used to specify and model the individuals (i.e., agent) in MAS and relatively in a low and micro abstraction level. In general, the functionalities, activities and resources of agents should be specified and analyzed independent of their concrete details.

In addition, according to the social premises described in section 2.2, agents in MAS may be dynamic or static, cooperative or self-interested. The dynamic agents may have different functionalities and activities in their lifecycles. For the self-interested agents, their functionalities, activities and resources may conflict with each other. Therefore, these social premises about agents also should be explicitly modeled and analyzed if the target system has these social properties.

- **Role, Position and Actor**

A *role* is an abstract characterization of the behaviors of agents within some

specified context of organization. Generally, an agent can play multiple roles and a role can be played by a number of agents in MAS. Other concepts similar to role are *position* and *actor* used in *i** and Tropos. *Position* is a collection of roles that are occupied by one agent and *actor* is a generic concept to denote the intentional entity that may be an agent or role or position.

These concepts are important to abstractly model the agents in MAS, and helpful to manage the complexity of MAS without considering the concrete details of agents (e.g., implementation architectures and technologies). They present an effective way to naturally model the entities in the system. In general, the system's roles that agents play are specified in the role model like ones in Gaia, MaSE, etc. Therefore, the *role* concept, we can find, has been integrated into almost all of the methodologies based on the organizational and social abstractions.

The dynamic properties of agent can be viewed as that agent plays different roles in different context and situation, which will facilitate to model the dynamic MAS. However, we believe, the traditional role models like ones in MaSE, Gaia, etc., are unable to model such dynamic information. Therefore, other system model based on the role concept should be developed like one in [36] to show how agents dynamically enter or leave roles in different social situations.

- ***Responsibility and Goal***

These concepts are used to specify and analyze the functionalities of a role. In Gaia, *responsibilities* are divided into two types; *liveness properties* and *safety properties*. *Liveness properties* describe those states of affairs that an agent must bring about given certain environment conditions. In contrast, *safety properties* correspond to the invariants in the multi-agent system that agent must maintain. The *goal* of a *role* represents its strategic interests or intentions. In *i** and Tropos, two kinds of *goals* can be distinguished: *HardGoal* and *SoftGoal*. The latter denotes the goal that has no clear-cut definition or criteria for decision whether it is satisfied or not, and is typically used to specify the non-functional requirements.

Generally the functionalities of roles should be specified and analyzed in requirement phase in order to understand the behaviors of roles and guide the software design that implements the roles' functionalities. In contrast to the tasks, actions and plans of roles, the responsibilities or goals of roles are relatively high-level and stable, even in open and dynamic system, and therefore easy to elicit and specify. In addition, roles are typically goal-driven, therefore the goals or responsibilities of roles are related with their tasks, plans and interactions. The explicit identification and specification of the goals or responsibilities of roles will facilitate to elicit and model the tasks or plans that roles have, the resources and interactions that roles need, the rule it should obey in order to achieve its goal or responsibilities. Moreover, they are also helpful to analyze the potential goals conflicts between the self-interested agents.

- ***Permission, Right and Resource***

These concepts are used to specify and analyze what the roles require in order to realize their functionalities. *Permissions* in Gaia are the "rights" associated with a *role*. The *permission* of a role identifies the *resources* that are available to

that role in order to realize its *responsibilities*. In the information system, the *permission* tends to be the information *resources* [8]. Other analogous concepts are *rights* in [3] and *resource* representing a physical or an informational unintentional entity in *i**, Tropos, and SODA.

Usually the resources are needed when agents intend to achieve its goals or responsibilities. In most cases, they are distributed in the environments that agents situate and may be dynamic. The resources in the environment are often limited and shared by a number of agents. To explicitly specify permission or resource of roles and model the environment that agents situate is significant to analyze how agent interacts with the environment, and the dependency between roles (e.g., some agents need resources while others produce resources). It is of particular importance to investigate the resource or “right” conflicts that may occur between the self-interested agents in dynamic system with limited resources.

- ***Activity, Plan, Task***

These concepts are used to specify and analyze the behaviors that roles should have in order to accomplish their functionalities. The *activity* of a role in Gaia is actually the “private” action that may be carried out by the agent without interacting with other agents in order to realize its *responsibilities*. The *plan* concept in Tropos (analogous to the concept *task* in *i** framework) represents, at an abstract level, a way of doing something. The execution of the *plan* can be a means for satisfying a goal [16]. The *tasks* in SODA, however, can be classified as individual ones and social ones and expressed in term of the *responsibilities* they involve, of the competence they require, and of the *resources* they depend on. Typical, *social tasks* are those that require a number of different competences and the access to several different *resources*, whereas *individual tasks* are more likely to require well-delimited competence and limited *resources* [29].

These concepts describe in more detail the behaviors of roles and are necessary in the requirement analysis phase to show how to accomplish the roles’ goals or responsibilities, and guide the software design that naturally encapsulate and implement these behaviors. Therefore we can find, most of the methodologies in AOSE support to model the role’s activity, plan, or task to some extent.

2.3.2 Concepts for Modeling Two Agents

The organizational and social concepts in this level are used to model the relationships between individual agents. In general, the structural relationship and the behavioral relationship between two agents should be modeled when developing MAS. The relationships between agents may change for the dynamic system when the roles that agents play vary. Therefore, such dynamic relationships between agents also should be specified and analyzed if the target systems are dynamic.

- ***Dependency and Interaction Protocol***

One of the most important relationships between agents may be the interactions, which describe the behavioral relationship between agents, and are often specified by *interaction protocol* which defines the ways that agents can interact with each

other. Most of methodologies, we can find, have developed various models to explicitly specify the interactions between agents, e.g., the interaction model in Gaia, communication model in MaSE. The *dependencies* between agents mainly describe the structural relationships between agents. They, in the i^* and Tropos, are used to indicate that one role depends, for some reason, on the other in order to attain some goal, execute some plan, or deliver a resource. The dependencies between roles can be classified as four kinds: *HardGoal dependency*, *SoftGoal dependency*, *task dependency*, *resource dependency*.

Both the structural relationship and the behavior relationship between agents should be modeled when developing MAS. They are also helpful to define the acquaintance model to show what agents or components in MAS are related with each other, which is important to analyze the system requirements and design the software architecture. The explicit specification and analysis of the dependencies between agents are also of particular importance to elicit and define the organization structure and the organization pattern. For dynamic MAS, there are still few works and efforts to model the dynamic relationships between agents. However, one possible way to deal with it is to define multiple models, for example the dependency model and/or interaction model, for agents with dynamic relationship in different organization context and situation.

2.3.3 Concepts for Modeling Two or More Agents Acting in a Coordinated Way

The organizational and social concepts in this level are used to specify and model the groups of MAS, in which agents act in a coordinated way. Group is an effective tool for partitioning and decomposing the organization, and organizing the agents with some common goals or purposes together, which is of particular importance for the hierarchic MAS. As for the dynamic system, the groups of MAS can also change from time to time. For example, the new group is created, agents dynamically leave the on-going groups and enter into a new one, the common characteristics (e.g., goals) of the agents in the group are formed. Therefore, in this level, the dynamic and hierarchic properties of the groups should be explicitly modeled if the target system has these social properties.

- ***Group and Group Structure***

A *group* is a set of agents sharing some common characteristics and used as a context for a pattern activities and for partitioning organizations [35,36,37]. It is similar to the concepts of *sub-organization* in [6,9] to decompose the system. Therefore, the *group* concept defines the atomic sets of agent aggregation. The *group structure* is the abstract description of a *group*. It actually identifies and specifies the structural information of groups such as all the *roles* and the *relationships* that can appear with a *group* [35].

- ***Common Goal, Joint Intention(Commitment)***

The behavioral information of group is another important part that should be specified and analyzed when developing MAS with groups. The common characteristic of agents in group that will govern the behaviors of them is often

specified by such concept as *common goal*. In addition, the concepts such as *joint intention and joint commitment*, are often used to describe how agents in group behave in a coordinated way to accomplish their common goals.

The group concept is useful to analyze, decompose and partition the hierarchic organization with clear group boundary, which is helpful to control the system complexity. The group structure can be used to instantiate various groups that can be created dynamically in the organization. However, additional models should be developed based on the concepts such as group and role in order to specify and analyze the whole dynamic information in this level. For example, [36] develops a model called as organization sequence diagram to specify and analyze how the groups are created and abolished dynamically, how agents in the organization dynamically enter or leave the groups in different organization context and situation.

2.3.4 Concepts for Modeling All Agents

The organizational and social concepts in this level are used to specify and model the macro organization information in MAS, especially for representing and analyzing the organization structure and the global constraints in the organization. The organization composed of all agents in the system may be open, which means that new, maybe unknown agents are allowed to enter into the organization. It is believed that such a system is difficult to develop. In addition, some organization has the global constraints such as social laws to govern the running of the agents (especially self-interested ones) in the organization. Therefore, in this level, the open and global constraint premises of the organization should be modeled if the target system has these social premises.

- ***Organization***

An *organization* is viewed as a collection of roles that stand in certain relationships to one another and take part in systematic institutionalized patterns of interactions with other roles [8]. However, Ferber pointed out such a definition lacks a very important feature of organization, i.e., partitioning, a tool to partition the system. Organizations are structured as aggregates of several partitions which may overlap and each partition may itself be decomposed into sub-partitions [36].

- ***Organization Rule, Social rule and Interaction rule***

These concepts define the global constraint information in the organization that will govern the running of the whole organization or society of MAS. *Organization rule* defines the general and global (supra-role) constraints requirements for the proper instantiation and execution of MAS that the actual organization will have to respect and expresses the information about how the organization is expected to work [4,5,6]. Generally, it will restrict the behaviors of agents and the interactions among them in MAS, and should have such properties as global, mutual consistent, satisfiable, stable and persistent, etc. Some methodologies such as Gaia [6] and MaSE [11] have introduced the concept as a fundamental element to model and analyze the MAS. Another analogous concept is *social law* in [1,2], which is actually for the artificial agent societies and

guarantees the successful coexistence of multiple programs and programmers. The *interaction rule* in SODA, however, is a special kind of organization rule which will governs the interactions among the social role and resources so as to make the group accomplish its social task [29].

- ***Organization Structure and Organization Pattern***

Organization structure defines the specific class of organization and control regime to which agents/roles have to conform in order for the whole MAS to work efficiently and according to its specified requirements. It is a design choice that expresses which kind of organization best fits requirements [4,5]. The structural organizational relationships are the most important parts that should be specified when defining the organization structure, such as control, peer, benevolence, dependency, and ownership, etc. The *organization pattern* defines and expresses pre-defined and widely used organizational structures that can be reused from system to system [6].

The concepts such as organization rule, social law, etc, are the natural abstraction to the global constraints in the organization. The explicit identification and specification of the *organizational rules* is of particular importance in the context of open systems. With the arrival of new, previously unknown, and possible self-interested agents, the overall organization must somehow enforce its internal coherency despite the dynamic and untrustworthy environment. The identification of global organizational rules allows the system designer to explicitly define: whether and when to allow newly arrived agents to enter the organization, and once accepted, what their position in the organization should be; which behaviors should be considered as an expression of self-interest, and which among them should be prevented [6]. The organization rule is also useful to develop the system with self-interested agents and limited resources in order to govern the autonomous behaviors of agents and the interactions among them to some extents and prevent systems from falling into chaos. Furthermore, the explicit definition of the *organization structure* is also helpful to the open system as the structure of the organization should persist when components or individual enter or leave an organization, and characterize the organization in the abstract or organization level. To reuse organization structure is an effective way to improve the software quality and development efficiency. Therefore, the specification and analysis of the organization structure and pattern are of particular importance not only for defining the structural information of system in the requirement specification and analysis phase, but also for promoting the software reuse.

3. Analyzing the Organizational and Social Concepts in AOSE Methodologies and Models

Up to now, there are many methodologies and models based on the organizational and social abstractions. A complete list of methodologies can be found in [23,24]. In this section as the space restriction we only analyze three of them, which are

influential in AOSE, to show what the restrictive assumptions about the social premises of organization have been made explicitly or implicitly, what and how the organizational and social concepts are presented and used to model MAS with these organization assumptions, and the meta-models of organizational and social concepts in these methodologies and models. The analysis will help understand why the social assumptions are made, the organizational and social concepts are used, and what the application domains they are suitable for.

3.1 MaSE

The MaSE (Multiagent Systems Engineering) methodology [12] provides guidelines for developing MASs based on multiple steps process covering the analysis and design phase. Although mostly based on the object-oriented technology, it also presents a number of organizational concepts and models to specify the MAS and guide the high-level design.

It is implicitly assumed in MaSE that the systems are static (i.e., the interactions between agents do not change at run-time) and closed, the agents in the systems are also static (i.e. their goals and tasks do not change at run-time), there is no conflict in the system and therefore agents are cooperative with each other. Such systems, we believe, are simple and easy to develop.

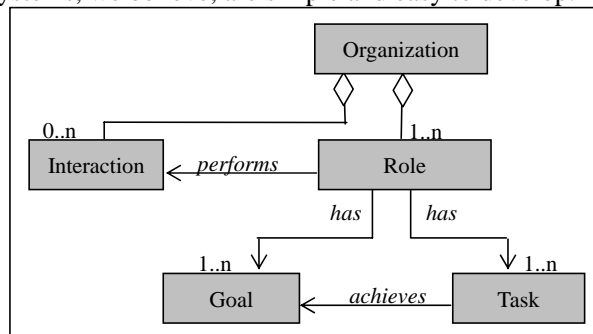


Fig. 1. The meta-model of the organizational and social concepts in MaSE

In the analysis phase, the system's coarse-grain global *goal* can be iteratively decomposed as a number of fine-grain sub-goals, based on which the system *roles* can be identified. Each *role* has a number of system *goals* to represent its functionalities and concurrent *tasks* to achieve its *goals*. The relationships among roles are specified as the *interactions* performed by agents playing the roles. The *roles*, the *goals* and *tasks* of roles, and the *interactions* between agents are all to be represented in the role model, which further guides the system high-level architecture design, i.e., defining agent class encapsulating the functionalities and activities of one or more roles and the conversations among them. The organizational and social concepts in MaSE are elementary to model MAS. It is acknowledged by the developer that the methodology is just suitable for the

closed and static MASs. Figure 1 shows the meta-model of organizational and social concepts in MaSE.

3.2 Gaia

Gaia is a complete methodology based on the organization metaphor for the development of MAS and views the process of analyzing and designing MAS as one of the constructing computational organizations. It is, to our knowledge, the first AOSE methodology that explicitly takes into account organization as first-class abstraction. There are two versions of Gaia methodology up to now. The original one is proposed by Wooldridge [8], and the recent version is one extended by Zambonelli [6]. As two versions are different in the organization assumption, abstractions, concepts and models, therefore we will explicitly distinguish them with different names: the former will be called as Gaia¹, the latter Gaia² and Gaia for two of them.

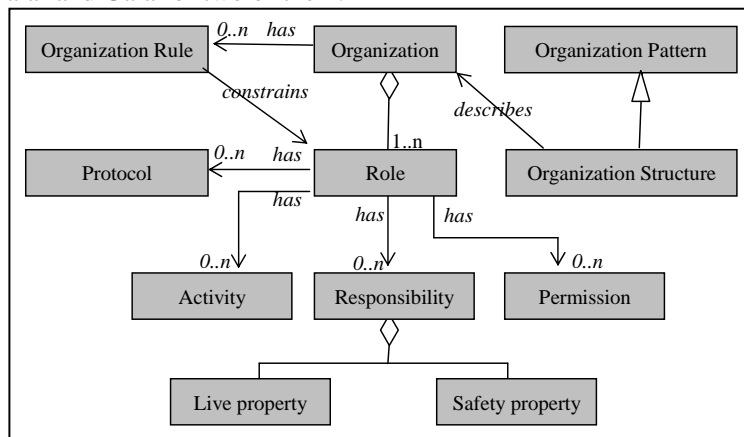


Fig. 2. The meta-model of the organizational and social concepts in Gaia

Gaia¹ views MAS as an organization composed of a collection of *roles* standing in certain relationships to one another and taking part in systematic patterns of *interaction* with other *roles*. It explicitly assumes that the systems to be developed are static (i.e., the inter-agent relationships do not change at run-time) and closed, the agents in the system are also static (i.e., the abilities of agents and the services they provide do not change at run-time), there is no true conflict in the system and therefore the agents in the system are cooperative with each other.

In order to analyze and design such kind of systems, Gaia¹ presents the role models and interaction models in the analysis phase, and agent model, service model and acquaintance model in the design phase. Each *role* in the role model is defined by four attributes: *responsibility*, *permission*, *activity* and *protocol*, which respectively define the functionalities, resources, private actions and interactions

of roles. The relationships among roles are simply defined as the *interactions* specified as the *interaction protocols* defined in the interaction model. The role model and interaction model specified in the analysis phase will guide the software architecture design specified by the agent model, service model and acquaintance model. According to organization assumptions, obviously Gaia¹ is just appropriate to the close and static system. A complete criticism on the limitations of the Gaia¹ methodology can be found in [20].

Gaia² is an extension to Gaia¹. The purposes of Gaia² are to introduce more organization abstractions into methodology and provide clear guidelines for the analysis and design of complex and open system. It assumes that the organization to be developed has global constraints that are represented by organization rule and hierarchic structure that can be divided into a number of sub-organizations. The system can be open or closed, dynamic or static. Agents in the system can be cooperative or self-interested. The extensions are mainly based on three high-level organization abstractions and concepts such as *organization rule* to represent the global organization constraints that, we have discussed in the last section, is beneficial to model the open system with self-interested agents, *organization structure* and *organizational pattern* that are helpful to specify and analyze the dynamic system and can be reused from system to system. It also explicitly models the environment information and extends the development process. These extensions, while preserving the simplicity of the Gaia¹ proposal, enable it to be used in the analysis and design of the open MASs with self-interested agents to some extent. Figure 2 shows the meta-model of organizational and social concepts in Gaia.

3.3 ALAADDIN

AALAADDIN [35] is actually an abstraction and generic MAS model and particularly focuses on the organization-centered MAS modeling in order to resolve the drawbacks of classical agent-centered technologies.

It assumes that systems to be developed are closed and have hierarchic structure that can be decomposed as a number of interrelated groups, and agents in the system are cooperative. The groups in the organization can be dynamically created, and agent can dynamically enter or leave the group. In order to model such systems, AALAADDIN introduces organizational concepts such as *group*, *role* and *structure*. In AALAADDIN, an *organization* is composed of a number of overlapping *groups*, each of them aggregates a number of agents that is an active entity playing *roles* within the *group*. An agent can be a member of multiple groups at the same time. AALAADDIN place no constraints on the internal architecture of agents and does not assume any formalism for individual agent. *Group* and *group structure* are introduced to partition *organization* and specify the group information. The *group structure* defines the *roles* and the *interactions* between the roles that can appear within the group. The concept of *organization structure* is also introduced, which defines the *group structure* in the organization

and the correspondences between them. A recent work in [36] extended the AALAADIN: the organization description consists in two aspects: a structural aspect and a dynamic aspect. The structure aspect of an organization is made of two parts: a partitioning structure and a role structure. The dynamic aspect of an organization is related to the institutionalized patterns of interactions that are defined within the roles, such as the creation of groups, the entering and leaving of a group by an agent, or acquisition of a role in relation, which can be specified by Organizational Sequence Diagram, an extension of sequence diagram in UML. Although AALAADIN is just an abstract MAS model, it is useful to support the analysis and specification of the system requirements and further guide the design of MASs. Figure 3 shows the meta-model of organizational and social concepts in AALAADIN.

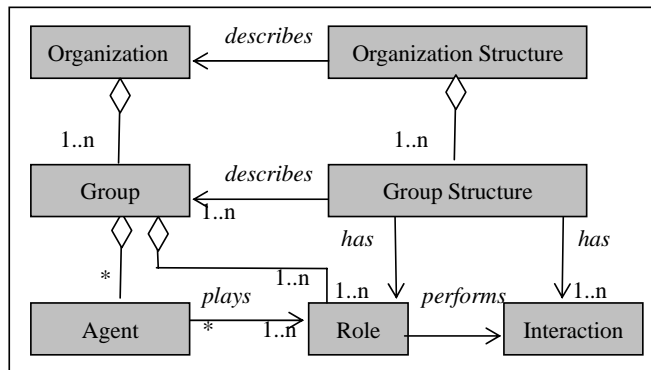


Fig. 3. The meta-model of the organizational and social concepts in AALAADIN

4. Conclusions

Organization and social metaphor is important in AOSE literature because it provides us such tools as abstraction and decomposition, etc. that are significant in software engineering area to naturally model the MAS and effectively control the system complexity. They are also easy to understand not only for the developers but also for the users and stakeholders, and helpful to reduce the concept distance between the systems in the real world and the models that we develop based on these concepts. Recently many methodologies and models based on the organization and society abstractions have been put forward. Although they all view MAS as organized society, they are actually diverse and different in the modeling purposes and organizational assumptions about the system to be developed, need different organizational and social abstractions, concepts and models, and therefore are suitable for different application domains.

The research in this paper shows that different organizational and social concepts have different modeling purposes and are used in different modeling construct levels for various MAS with different social premises. Although some

methodologies and models claim that they support the development of open and complex system, the existing methodologies are still weak in modeling the open, dynamic organization with self-interested agents. We believe that more modeling mechanisms such as concepts and models should be introduced and developed in order to specify and analyze the self-interested agent and the open and dynamic properties of systems. In addition, the organizational and social concepts in AOSE literature are somewhat messy. Some organizational and social concepts even having different terms actually have similar semantics and denotation. Most of concepts still have no widely-accepted definitions, and have different interpretations in different methodologies and models. Therefore, the standardization of the organization concepts, their semantics and representation syntax are of particular importance if we want the methodologies and models to be used effectively in wider domains.

References

1. Y Shoham and M Tennenholtz, On the synthesis of the useful social laws for the artificial agents societies, in Proc. AAI-92, AAAI press, 1992.
2. Y Shoham and M Tennenholtz, On Social Laws for Artificial Agent Societies: Off-Line Design, Artificial Intelligence, Artificial Intelligence 73(1-2), 1995.
3. E Alonso, Rights for Multi-agent Systems, UKMAS 2002, LNAI 2403, 2002.
4. F. Zambonelli, N. Jennings, M. Wooldridge, Organizational Rules as an Abstraction for the Analysis and Design of Multi-agent Systems, Journal of Knowledge and Software Engineering, 11(3), 2001.
5. F. Zambonelli, N.R.Jennings, M Wooldridge, Organizational Abstractions for the Analysis and Design of Multi-agent System, AOSE'2001, LNCS2222, Springer, 2002.
6. F. Zambonelli, N. R. Jennings, and M. Wooldridge, Developing Multiagent Systems: The Gaia Methodology, ACM Transactions on Software Engineering Methodology, 12(3), 2003.
7. F. Zambonelli, A.Omicini, and M. Wooldridge, Agent-Oriented Software Engineering for Internet Applications, in Coordination of Internet Agents: Models, Technologies and Applications, Springer-verlag, 2000.
8. Wooldridge, N R.Jennings, and D.Kinny, The Gaia Methodology for Agent-Oriented Analysis and Design, International Journal of Autonomous Agents and Multi-agent System,3, 2000.
9. N.R.Jennings, Building Complex Software System: the Case for an Agent-based Approach, Communication of ACM, 44(4), 2001.
10. N.R.Jennings, On Agent-based Software Engineering, Artificial Intelligence, 117(2), 2000.
11. S A. DeLoach, Modeling Organizational Rules in the Multiagent Systems Engineering Methodology, Proc. of the 15th Canadian Conference on Artificial Intelligence Calgary, Alberta, Canada. May 27-29, 2002.
12. S A. DeLoach, M F.Wood, and C H.Sparkman, Multiagents Systems Engineering, International Journal of Software Engineering and Knowledge Engineering, 11(3), 2001.
13. E. Yu, Agent-Oriented Modelling: Software Versus the World, Proc. Of Agent-Oriented Software Engineering, LNCS 2222, Springer-Verlag, 2001.
14. E. Yu, et.al., From Organization Models to System Requirements: A Cooperative Agents Approach, in Cooperative Information Systems: Trends and Directions, 1997.
15. E Yu, Towards Modeling and Reasoning Support for Early-Phase Requirements

- Engineering, Proceedings of the 3rd IEEE Int. Symp. on Requirements Engineering, 1997.
16. F Giunchiglia, John Mylopoulos and A Perini, The Tropos Development Methodology: Processes, Models and Diagrams, AAMAS, 2002.
 17. P Massonet, Yves Deville and C Neve, From AOSE Methodology to Agent Implementation, Proc of AAMAS'02, 2002.
 18. Massimo Cossentino, Different Perspective in Designing Multi-Agent Systems, Proc. of AGES '02 workshop, Germany, 2002.
 19. G Caire, et.al., Agent Oriented Analysis Using MESSAGE/UML, Proc. of Second International Workshop on Agent-Oriented Software Engineering, 2002.
 20. Thomas Juan, Adrian Pearce and Leon Sterling, ROADMAP: Extending the Gaia Methodology for Complex Open System, Proc. of AAMAS'02, 2002.
 21. Thomas Juan, Leon Sterling and Michael Winikoff, Assembling Agent Oriented Engineering Methodologies from Feature, In Proc. of AOSE, 2002.
 22. J Pavon and J Gomez-Sanz, Agent Oriented Software Engineering with INGENIAS, Proc of CEEMAS 2003, LNAI 2691, 2003.
 23. Gerhard Weib, Agent Orientation in Software Engineering, The Knowledge Engineering Review, Vol.16(4), 2001.
 24. Ofer Arazy and Carson C.Woo, Analysis and design of agent-oriented information systems, The knowledge engineering review, vol.17:3, 2002.
 25. M Kim, et.al., Agent-Oriented Software Modeling, Proc. of Sixth Asia Pacific Software Engineering Conference, 1999.
 26. K H Dam and M Winikoff, Comparing Agent-Oriented Methodologies, in the proceedings of the Fifth International Bi-Conference Workshop on Agent-Oriented Information System, 2003.
 27. Luca Cernuzzi and Gustavo Rossi, On the Evaluation of Agent Oriented Modeling Methods, Proceedings of the 3rd International Conference on Enterprise Information Systems, 2001.
 28. Amon Sturm and Onn Shehory, A Framework for Evaluating Agent-Oriented Methodologies, Proc. Of AOIS, 2003.
 29. Andres Omicini, SODA; Societies and Infrastructures in the analysis and Design of agent-based System. Proc. of AOSE, 2001.
 30. Andrea Omicini, From Object to Agent Societies: Abstractions and Methodologies for the Engineering of Open Distributed Systems, AI*IA/TABOO Joint Workshop, 2000.
 31. H.V D Parunak and J J.Odell, Representing Social Structure in UML, AOSE 2001, LNCS 2222, 2002.
 32. M. Dastani, V. Dignum, F. Dignum, Organizations and Normative Agents. In Proceedings of the First Eurasian Conference on Advances in Information and Communication Technology, 2002.
 33. Christian Lemaître and Cora B. Excelente, Multi-Agent Organization Approach, 2nd Iberoamerican Workshop on Distributed Artificial Intelligence and Multi-Agent Systems, 1998
 34. Mahdi Hannoun, Olivier Boissier, Jaime Simão Sichman, Claudette Sayettat: MOISE: An Organizational Model for Multi-agent Systems. IBERAMIA-SBIA, 2000.
 35. J. Ferber and O. Gutknecht. A meta-model for the analysis and design of organizations in multi-agent systems. In Proceedings of Third International Conference on MultiAgent Systems, IEEE Computer Society, 1998.
 36. J Ferber, Olivier Gutknecht, Fabien Michel: From Agents to Organizations: An Organizational View of Multi-agent Systems. Proc of AOSE 2003.
 37. J Ferber, et.al., Organization Models and Behavioural Requirements Specification for Multi-Agent Systems, Proc. of the ECAI 2000 Workshop on Modelling Artificial Societies and Hybrid Organizations, 2000.