Modeling Social Aspects of Multi-Agent Systems
The AML Approach

Radovan Cervenka, Ivan Trencansky, and Monique Calisti

Whitestein Technologies, Panenska 28, 811 03 Bratislava, Slovakia
Tel +421 (2) 5443-5502, Fax +421 (2) 5443-5512
{rce,itr,mca}@whitestein.com
http://www.whitestein.com

Abstract. This paper presents modeling concepts and mechanisms of the *Agent Modeling Language* (*AML*) to model social aspects of multi-agent systems. The modeling of structural, behavioral as well as attitudinal aspects of multi-agent systems from the social perspective is discussed and demonstrated on examples.

1 Introduction

Multi-agent systems (*MAS*) are generally perceived as systems comprised of a number of autonomous agents situated in a common environment. Agents in such systems are rarely isolated. More often they are required to interact with each other so that the desired functionality and properties of the systems could emerge. These features of MAS are not always derivable or representable solely on the basis of properties and capabilities of their individual component agents, but also arise from agents' mutual relationships, interactions, coordination mechanisms, social attitudes (e.g. common or individual beliefs, goals, intentions, desires, commitments), etc., which are commonly referred to as social aspects of multi-agent systems. Social ability of agents is thus one of their most fundamental properties and therefore of central concern for the most of the MAS modeling approaches.

From a social perspective, the following aspects are commonly considered in MAS models:

- *Social structure* concerning mainly with the (1) identification of societies (groups, organizations, institutions, etc.) which can evolve within the system, (2) specification of their properties, (3) identification of comprised roles, social entities that can participate in such societies, roles they can play, (4) specification of the society structure (in terms of social relationships), etc.
- *Social behavior* covering such phenomena as (1) social dynamics, i.e. temporal relationships and causality of social events (i.e. changes of the society state) such as the formation/abolition of societies, the entrance/withdrawal of an entity to/from a society, acquisition/disposal/change of a role played by an entity, modification of properties of a society or its members, etc., (2) social interactions, i.e. how individuals and/or societies interact with others

in order to exchange information, coordinate their activities, etc., (3) social activities, i.e. activities of single social entities or aggregate/emergent activities of societies which influence or are influenced by the state or behavior of other members of the society or the society itself, and (4) norms, i.e. rules or standards of behavior shared by members of a society.
- *Social attitudes* addressing the individual and/or common tendencies (usually expressed in terms of motivations, needs, wishes, intentions, goals, beliefs, commitments, etc.) to anything of a social value.

The purpose of this paper is to present the AML approach to modeling the above-mentioned social aspects of MAS. The focus, however, due to limitation in paper length, is on the structural aspects and role modeling. Details of modeling behavior (used for modeling social behavior) and mental attitudes (used for modeling social attitudes) will be described in forthcoming papers.

The rest of the paper is structured as follows: Section 2 presents a short summary of other approaches to modeling social aspects of MAS. Section 3 provides a brief introduction to AML, Section 4 discusses modeling of social structures, Section 5 modeling of social behavior, and Section 6 modeling of social mental aspects in AML. The conclusions and future work directions are drawn in Section 7.

2 Previous Work

Most of the currently available agent-oriented modeling languages and methodologies consider social modeling as one of the crucial activities. The proposed approaches, however, differ in the scope and supported modeling concepts.

Several approaches (e.g. Gaia [1, 2], MaSE [3], INGENIAS [4], PASSI [5], MESSAGE [6]) model MAS societies in terms of organizations (or groups) composed of a collection of roles related to one another and participating in patterns of interactions with other roles. The agents are then specified in terms of a set of roles they can play. These approaches explicitly assume that the inter-agent relationships and the abilities of agents do not change at run-time, and that all the agents are explicitly designed to cooperatively achieve common goals.

In *Gaia* [1], during the analysis phase the roles model (identifying and specifying the key roles in terms of responsibilities, permissions, activities, and protocols) and interaction model (specification of details of interaction protocols in terms of purpose, initiator, responder, inputs, outputs, and processing) are developed. In the design phase, the analysis models are refined by the agent model (specification of various agent types in terms of a set of roles they play), the service model (identification and specification of services associated with the roles), and the acquaintance model (a directed graph identifying the communication links between agents types).

The later version of Gaia [2] extends the former one in order to better suit to open MAS, i.e. systems in which agents are not designed to share common goals, have been (possibly) developed by different people to achieve different objectives,

and the composition of which can dynamically change as agents enter and leave the system. To achieve the above, two new abstractions have been identified: organizational rules and organizational structures. Organizational rules are concerned with explicit identification of the relationships and constraints between roles, protocols, or between roles and protocols, that the organization will have to respect in order to define whether and when new agents are allowed to enter the organization, and once accepted, what their position should be, and which behaviors should be considered as a legitimate expression of self-interest, and which among them must be prevented by the organization. Organizational structures are used to explicitly define organizations in terms of their topology and control regime.

In *MaSE* [3], during analysis the role model (based on the goal hierarchy diagram, use case model, and sequence diagrams) is developed to specify the roles, their associated tasks, and interactions between role tasks. The role behavior is specified by means of concurrent task model. During the design phase (1) agent class diagrams, to specify agent classes in terms of roles they play and conversations in which different agent classes must participate, are created, (2) an internal agent architecture is specified by means of the agent architecture diagram, and (3) a deployment diagram, to show the numbers, types, and locations of agent instances within the system, is developed.

INGENIAS [4] supports modeling of social aspects by modeling organizations, task and goals of agents, their dependencies among different system and agents goals, and interactions (using UML [7] collaboration diagrams, or AUML [8, 9] protocol diagrams).

The MAS society modeling in *PASSI* [5] consists of domain and communication ontology modeling (by means of UML class diagrams and OCL constraints), roles identification (supported by sequence diagrams), role description (by means of class diagrams), and the description of the protocols used in the agent interactions (modeled by means of AUML agent interaction protocols). The modeling of organizational rules (compositional and behavioral) is in PASSI explicitly considered and modeled by means of constraints within ontology and role models.

For modeling social aspects of MAS, *MESSAGE* [6] uses the following views: (1) the organization view defines the architecture of the society in terms of roles, their mutual relationships, and which agents play which roles, (2) The goal/task view shows goals, tasks, situations and dependencies between them (e.g. decomposition, temporal dependencies, etc.), and (3) the agent/role view describes what goals particular agents/roles are responsible for, what events they sense, what resources they control, what tasks they are capable to perform, etc. (4) The interaction view is used to model interaction in terms of initiators, responders, motivators (often a goal of the initiator) of an interaction, plus other optional information such as trigger conditions, information achieved and supplied by each participant, etc.

ADELFE [10] uses UML [7] and AUML [11, 12, 9] modeling languages. The social aspects of MAS are considered particularly in steps concerning identification of agent relationships, interaction languages, and elaboration of so called

non cooperative situations (i.e. incomprehension, ambiguity, incompetence, unproductiveness, concurrency, conflict, and uselessness).

In *Prometheus* [13] the social aspects are captured mainly by the acquaintance relationships between agents and the interaction protocols used within the interactions between agents.

The notions of groups (agentified and non-agentified), roles, agents and agent role assignments are the basic building blocks for defining agent societies in *AUML* (see [11, 12, 9]). The concept of agent role assignment in AUML is not considered to be static, but can change over time. For modeling changes in role playing, AUML makes use of dynamic classification of agents according to the roles which assignments to agents can change over time. For modeling social interactions AUML provides an extended version of UML interactions (for details see [9]).

AALAADIN [14] defines a meta-model of multi-agent systems based on the three main concepts of agents, groups, and roles. By means of these fundamental concepts group structures and organization structures are developed. The group structure defines all the roles and interactions that can appear within the group, as well as the interaction protocols used. The organization structure defines the group structure in the organization (system) and the correspondence between them. The groups in AALAADIN can be dynamically created, and agents can dynamically enter or leave groups. To model dynamics of organizations, i.e. temporal relation between organizational events such as the creation of groups, the entering or leaving of a group by an agent, or the acquisition or disposal of a role, Organizational Sequence Diagrams (a variant of UML Sequence Diagrams) are introduced.

Tropos [15] is a requirements driven methodology applying actors and goals as fundamental modeling concepts to all phases of software development process. It adopts concepts from organization theory of strategic alliance to model MAS architectures. For this purpose the strategic dependency model of i^* [16] is used, which can be described as a graph, where each node represents an actor (an agent, position, or role within an organization) and each link between two actors indicates that one actor depends on another for a goal to be fulfilled, a task to be carried out, or a resource to be made available. Tropos also introduces a catalog of organization-inspired architectural styles used to design the overall architecture of MAS, and a catalog of patterns of how goals assigned to actors of an organizational architecture are fulfilled by agents.

To model social aspects of MAS, the *TAO* methodology (accompanied with MAS-ML modeling language) [17] uses concepts like agents, object and agent roles, organizations, interactions, beliefs, goals, plans and actions. The societies (organizations) are specified in terms of object and agent roles, their properties and mutual relationships. An (object) role guides and also restricts the behavior of the instances that play it. From the point of view of the element that is related to the object that is playing a role, the role identifies the properties that the element can see and identifies the available relationships. In addition to object roles, agents roles guide and restrict agents' behavior by means of specified goals,

beliefs, and actions that an agent must or may perform while playing a given role. TAO also supports the social dynamics by modeling processes of a dynamic organization and agent instance creation, processes concerned with agents entering/leaving organizations, and processes of commitments to (disposal of) played roles.

AOR [18] introduces concepts of commitments and claims to perform certain actions, or to make sure that certain conditions hold, as one of the fundamental concepts of social interactions. AOR also differentiates between different types of manifestation of behavior within a society of agents: particularly communicative and non-communicative action events, commitments/claims (coupled with the corresponding types of action events), and non-action events. Commitment and claim processing includes their: creation, cancellation, waiving, delegation, assigning, and fulfilling. The commitment/claim processing steps are in AOR proposed to be expressed by means of reaction rules in a declarative way. Interaction frame diagrams are used to describe possible interactions between two (types of) agents by means of various types of communicative and non-communicative action events, commitments/claims, and non-action events.

3 AML

The *Agent Modeling Language* (*AML*) [19, 20] is a visual modeling language for specifying, modeling and documenting systems that incorporate concepts drawn from the MAS theory. It is specified as an extension to UML 2.0 [7] in accordance with the OMG modeling frameworks (MDA, MOF, UML, and OCL).

AML provides a consistent set of modeling constructs designed to capture the various aspects of multi-agent systems, i.e. ontologies, MAS entities, social aspects, behavior abstraction and decomposition, communicative interactions and interaction protocols, services, observations and effecting interactions, mental aspects used for modeling mental attitudes of autonomous entities, MAS deployment, and agent mobility. Details about how AML modeling elements can be used to model particular social aspects are described in following sections.

4 Modeling Social Structure

For modeling structural aspects of agent societies, to some extent, modeling elements of UML can be used. However, to allow building of more concise and comprehensive models of MAS societies, AML offers several modeling elements designated to explicitly represent various (MAS) society abstractions. In particular these are: social entities, entity roles, social relationships, play associations and role properties.

4.1 Social Entities

Entities represent objects that can exist in the system independently of other objects. AML defines the following entities: agents, resources, environments, and

organization units. Entities are usually modeled at the level of types (specialized UML classes), but can also be modeled at the instance level by UML instance specifications classified according to their corresponding types. All entities can make use of modeling mechanisms inherited from UML class, i.e. they can own features, participate in varied relationship types, be internally structured into parts, own behaviors, etc. In addition to these, AML allows to specify also possibility to own capabilities, perform speech act based interactions, provide and use of services, own perceptors and effectors, play roles, be characterized in terms of mental attitudes, etc.

Social entities are entities possessing social abilities. By *social ability* we understand the ability to (1) participate in societies and social relationships, (2) manifest social behavior, and (3) have social attitudes. Two of the above mentioned entities, agents and organization units, are also social entities.

Agent type is a specialized UML class used to model the type of *agents*, i.e. self contained entities that are capable of interactions, observations and autonomous behavior within their environment.

Organization unit type is a specialized environment type[1] used to model the type of an *organization unit*. From an external perspective, organization units represent coherent autonomous entities, the features and behavior of which are both (1) emergent properties and behavior of all their constituents, their mutual relationships, observations and interactions, and (2) the features and behavior of organization units themselves. From an internal perspective, organization units are types of environment that specify the social arrangements of entities in terms of structures, interactions, roles, constraints, norms, etc.

Organization units are thus used to model societies (groups, organizations, institutions, etc.). They are not considered to be static, i.e. their properties, structure, behavior, social attitudes, comprised roles, participating entities and their features, etc. can change over time (for details see Sect. 5).

Fig. 1 (a) shows an example of a class diagram depicting a generic organization structure of a software development project. The project teams (ProjectBoard, TechnicalTeam, AnalysisTeam, etc.) are modeled by means of organization unit types, their social relationships by means of social associations (Sect. 4.2), and a project role (ProjectManager) by means of the entity role type (Sect. 4.3).

Using the specified types, the internal structure of the organization unit type SoftwareDevelopmentProject is modeled in Fig. 1 (b). It defines the parts which represent comprised entity role (pm) and lower-level organization units (pb, ana, imp, and tst). Connectors between them declare instances of the social associations specified at the class level, and therefore use the same adornments as the corresponding associations.

[1] *Environment type* is an element used to model a specific aspect of a system's inner environment, i.e. the logical or physical surroundings of entities which provide conditions under which the entities exist and function.

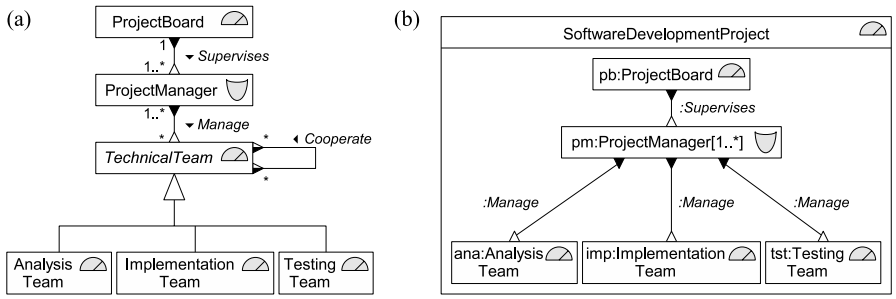

Fig. 1. Example of class-level organization structure model

4.2 Social Relationships

Social relationship is a particular type of connection existing between social entities related to or having to deal with each other. Apart from other general-purpose UML relationships applicable in social models (generalization, aggregation, association, etc.), AML defines a special type of property, called the *social property*, used to model social relationships. It can be used either in the form of an owned *social attribute* or as the end of a *social association*. Social property, in addition to UML property, allows to specify the relationship's social role kind. AML supports two predefined kinds of social relationships, peer-to-peer and superordinate-to-subordinate, and the analogous social role kinds: peer, superordinate, and subordinate (for details see [19]). The set of supported social role kinds can be extended as required (e.g. to model producer-consumer, competitive, or cooperative relationships).

Fig. 1 (a) shows superordinate-to-subordinate social associations between ProjectBoard (superordinate—shown as a filled triangle placed at the association end) and ProjectManager (subordinate—shown as a hollow triangle placed at the association end), and ProjectManager (superordinate) and TechnicalTeam (subordinate). Particular technical teams are related by the peer-to-peer relationships which is represented by the social association Cooperate attached to the TechnicalTeam organization unit type (peer social property kind is shown as a half-filled triangle placed at the end of association). Connectors representing instances of the previously defined social associations are depicted in Fig. 1 (b).

4.3 Entity Roles

In AML, *social roles* (i.e. abstractions of features, behavior, attitudes, participation in interactions, and services required or provided to other roles or social entities in a particular social situation) are modeled by entity role types. Entity roles types thus can be used to specify (1) social structures, (2) positions[2],

[2] A *position* is a set of roles typically played by one agent [21]. Positions are in AML explicitly modeled by means of composed entity roles types.

and also (3) required structural, behavioral and attitudinal features of their constituents.

Technically, entity role types are specialized UML classes which can own capabilities, perform speech act based interactions, provide and use services, own perceptors and effectors, be characterized in terms of mental attitudes, etc. Each entity role type should be realized by a specific implementation possessed by a social entity type which can play it. An instance of the entity role type is called *entity role*[3]. It represents the execution of behaviors, usage of features and/or participation in interactions as defined by the particular entity role type. A given entity role exists only while a behavioral entity instance plays it.

The AML approach provides the possibility to model social roles at both the class level (where the required types of features and behavior are defined) and the instance level (where the concrete property values and behavior realization of a particular role playing can be specified) explicitly.

4.4 Entity Role Playing

The ability of a social entity to play an entity role is modeled by special structural feature called *role property*.

Role property is a specialized UML property used to specify that an instance of its owner (an entity type) can play one or several entity roles of the entity role type specified as the property's type. An instance of a role property's owner is called the *entity role player* (or simply *player*). An instance of the role property's type represents the played entity role. The role property can be used either in the form of a *role attribute* or as the member end of a *play association*.

One entity can at each time play several entity roles. These entity roles can be of the same as well as of different types. The multiplicity defined for a role property constraints the number of entity roles of given type, the particular entity can play concurrently. Additional constraints which govern playing of entity roles can be specified by UML constraints.

The AML approach to mode role playing allows:

- Specification of the *possibility* to play particular entity roles by entities expressed at the class level, and the *actual playing* of entity roles by instances expressed at the instance level.
- Separation of entity's own features and behaviors from the features and behaviors required for playing an entity role in a particular situation.
- Separation of a specification of the features, behavior, and attitudes required (or expected) from a potential player of that entity role, from their actual realization by actual players.
- Specification of the behavior related to role playing, e.g. role playing dynamics, life cycle of roles, reasoning about roles, etc. (for more details see Sect. 5).

[3] AML uses the term "entity role" to differentiate agent-related roles from the roles defined by the UML 2.0, i.e. roles used for collaborations, parts, and associations.

Fig. 2 shows an example of specifying entity role types (abstract ProjectMember and all its concrete subclasses), their attributes, and social associations. The possibility to play instances of concrete entity role types by agents of the type Person, represented by the play associations is also depicted.

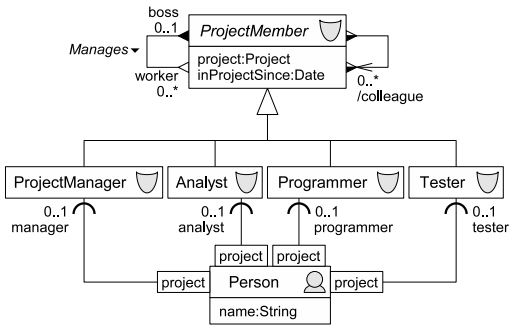

Fig. 2. Example of entity roles types and play associations

Fig. 3 shows the instantiation of the previously defined types in the model of a system's snapshot, where the agent Alan, of type Person, plays two entity roles (testerAlan and analystAlan), and agent John, also of type Person, as the project manager (entity role pmJohn) is Alan's boss. This example also demonstrates the ability of AML to explicitly specify slots and social links of entity roles played under certain circumstances (specified for analystAlan).

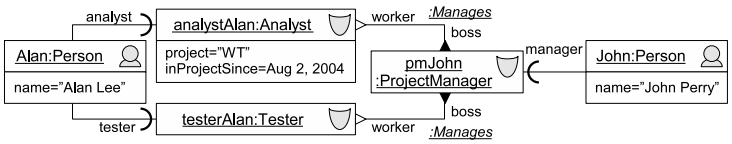

Fig. 3. Example of the entity role instantiation and playing

5 Modeling Social Behavior

Social behavior is the behavior of a social entity (behavior of a single social entity, or emergent behavior of a society) which influences or is influenced by the state (social features, attitudes, etc.) or behavior of other social entities (members of the society or the society itself). Social behavior thus covers social dynamics, social interactions, and social activities.

This section briefly describes how AML extensions to UML behavioral models can be used to model social behavior.

5.1 Social Dynamics

The central modeling mechanism for modeling social dynamics are state machines as the most appropriate mechanism for modeling state transitions in reaction to events. Incorporation of AML specific actions into the UML state machines allows explicit modeling of: the formation/abolition of societies, the entrance/withdrawal of an entity to/from a society, acquisition/disposal/change of a role by an entity, etc.

5.2 Social Interactions

To model social interactions, AML defines specialized modeling constructs for modeling speech act based interactions, observations and effecting interactions.

The extensions toward modeling social interactions are twofold: generic and communicative interaction specific. Generic extensions to UML interactions are provided to in order to model interactions between groups of entities (using *multi-message* and *multi-lifeline*), dynamic change of object's attributes (to express changes in internal structure of organization units, social relationships, or played entity roles, etc.) induced by interactions (using *attribute change*), modeling of messages and signals not explicitly associated with an invocation of corresponding methods and receptions (using *decoupled message*). Communicative interaction specific extensions comprise: modeling of speech-acts (using *communicative acts*), speech act based interactions (by *communicative interactions,* which are specialized UML interactions), and patterns of interactions (by means of *interaction protocols*).

AML furthermore defines several constructs for modeling observations (i.e. the ability of entities to observe features of other entities) and effecting interactions (i.e. the ability of entities to manipulate, or modify the state of, other entities). Observations are modeled as the ability of an entity to perceive the state of (or to receive a signal from) an observed entity by means of *perceptors*. *Perceptor types* are used to specify (by means of *perceiving acts*) the observations an owner of a perceptor of that type can make. The specification of which entities can observe others, is modeled by a *perceives* dependency. Different aspects of effecting interactions are modeled analogously, by means of *effectors*, *effector types*, *effecting acts*, and *effects* dependencies.

Fig. 4 shows an example of the communicative interaction in which the attribute change element is used to model change of entity roles played by agents. The diagram realize the scenario of replacing a project manager by another person, as described by the scenario shown in Fig. 5. An agent worker1 is a manager (modeled by its role property manager). After receiving a message resignResponsibility from the project board (pb) it stops playing the role of project manager. At the same time another person, worker2, takes the responsibility, as the result of previously received message takeResponsibility sent by the project board, and starts to play the role of the project manager (modeled by the manager property of worker2).

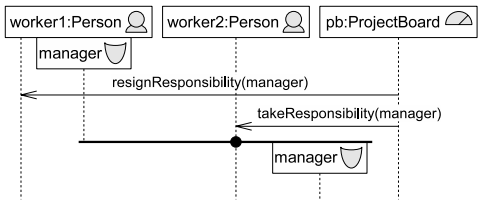

Fig. 4. Example of a social interaction with entity role changes

5.3 Social Activities

For modeling social activities UML activities can be used. However, to allow development of more concise and comprehensive models, AML offers several additional modeling concepts and mechanisms.

To allow modeling of modification of social features (i.e. social relationships, roles played, social attitudes), all of them are modeled as structural features of entities. This allows to make use of some UML actions of manipulation with structural features, to model modification of social structures, reasoning about played entity roles, access and reason about social attitudes, execute social behavior, etc.

Furthermore, AML defines specific actions to: create/play and dispose entity roles, send and receive messages of social communicative interactions, percept and effect other entities, commit to and decommit from goals, etc.

Fig. 5 shows an example of the activity describing the scenario of replacement the project manager by another person. The project board (organization unit pb) decides about the replacement, and informs the current project manager (entity role pm) together with a new potential project manager (agent potentialPM) about the decision. The current project manager stops playing its role of the project manager (expressed by the dispose role action Resign) and the new project manager starts to play its new entity role (specified by the create role action Become a new project manager creates a new entity role newPM).

6 Modeling Social Attitudes

For modeling types of social attitudes AML offers constructs of mental modeling, particularly: *beliefs*, *goals*, *plans*, and *mental relationships* (used to represent logical relationships between mental states such as means-ends, decomposition, correlation, contribution, etc.). Social attitudes of entities are modeled by means of special type of UML property called *mental property*. It can be used either in the form of an owned *mental attribute* or as the end of a *mental* asso*ciation*.

In general, two kinds of social attitudes can be recognized:

1. social attitudes shared by several entities within a society, e.g. common beliefs and goals, plans which include collaboration of several entities, etc., and

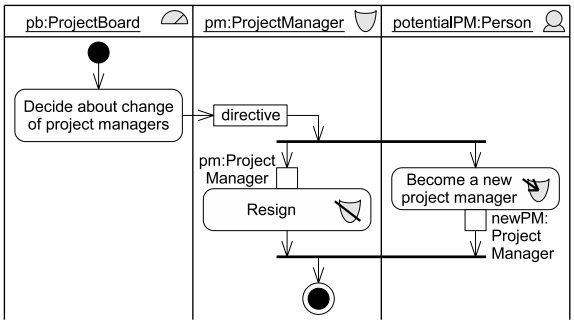

Fig. 5. Example of activity comprising the social actions inducing changes of entity roles

2. social attitudes of individual entities to anything of a social value, e.g. commitment to perform a social action, beliefs in some facts about other entities.

Social mental models often contain explicitly modeled relationships between mental states of different socialized entities. For instance cooperative entities share their goals, trusted entities share their beliefs, superordinate entities dictate their goals or form goals of subordinate entities, competitive entities have goals in contradiction, etc. These situations can be explicitly expressed by mental relationships.

Fig. 6 depicts a model excerpt showing social mental model of cooperating organization unit types AnalysisTeam and ImplementationTeam. They share common goal SatisfyQualityCriteria and belief that "the produced/consumed analysis is complete and consistent". Apart from the common mental attitudes, the teams have also specific purposes modeled as decidable goals, particularly the AnalysisTeam has the AnalyzeSystem and the ImplementaionTeam has the ImplementSystem. Positive necessary contribution to the overall project's goal to develop the system (DevelopSystem) is also shown.

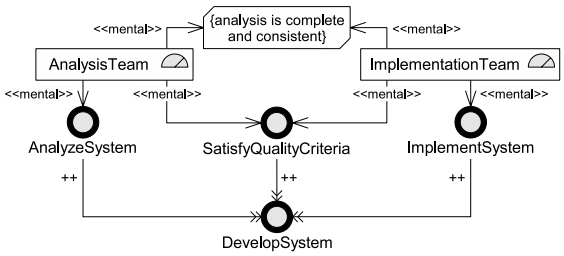

Fig. 6. Example of individual and common goals and beliefs

7 Conclusions and Further Work

AML provides a rich set of modeling constructs aimed at modeling social aspects of MAS. It covers modeling of social structure, behavior and attitudes. Moreover, the presented modeling mechanisms are integrated into the framework of AML (defined by means of metamodel and notation), which provides comprehensive means to complexly model systems in terms of MAS abstractions from various perspectives.

Even if the current version of AML provides well-defined, sufficiently comprehensive and stable generic mechanisms for modeling social aspects, by exploiting the UML extensibility mechanisms and flexible architecture of AML specification, some language improvements and extensions are foreseen. It is also planned to create technology-specific modeling frameworks (re-usable model libraries) and AML extensions which will customize and extend the generic AML modeling constructs to enable modeling of specific architectural concepts of particular (MAS) technologies.

References

1. Wooldridge, M., Jennings, N.R., Kinny, D.: The Gaia Methodology for Agent-Oriented Analysis and Design. Journal of Autonomous Agents and Multi-Agent Systems **3** (2000) 285–312
2. Zambonelli, F., Jennings, N., Wooldridge, M.: Developing Multiagent Systems: The Gaia Methodology. ACM Trans. on Software Engineering and Methodology **12** (2003) 317–370
3. DeLoach, S., Wood, M., Sparkman, C.H.: Multiagent Systems Engineering. International Journal of Software Engineering and Knowledge Engineering **11** (2001) 231–258
4. Pavon, J., Gomez-Sanz, J.: Agent Oriented Software Engineering with INGENIAS. In Maik, V., Muller, J., Pchouek, M., eds.: Multi-Agent Systems and Applications III: 3rd International Central and Eastern European Conference on Multi-Agent Systems, CEEMAS 2003, LNCS 2691 / 2003, Springer-Verlag (2003) 394
5. Cossentino, M.: Different perspectives in designing multi-agent systems. In: AGES '02 workshop at NODe02, Erfurt (2002)
6. Evans, R., Kearny, P., Stark, J., Caire, G., Garijo, F., Sanz, G., Leal, F., Chainho, P., Massonet, P.: MESSAGE: Methodology for Engineering Systems of Software Agents. Methodology for Agent-Oriented Software Engineering. Technical Report Eurescom project P907, EDIN 0223-0907, EURESCOM (2001)
7. OMG: Unified Modeling Language: Superstructure. Version 2.0. ptc/03-08-02 (2003)
8. Odell, J., Parunak, H., Bauer, B.: Representing Agent Interaction Protocols in UML. In Ciancarini, P., Wooldridge, M., eds.: Proceedings on the First International Workshop on Agent-Oriented Software Engineering (AOSE 2000) , Limerick, Ireland, Springer (2000) 121–140
9. Huget, M.: Agent UML Notation for Multiagent System Design. IEEE Internet Computing **8** (2004) 63–71

10. Bernon, C., Camps, V., Gleizes, M., Picard, G.: Designing Agents' Behaviours and Interactions within the Framework of ADELFE Methodology. In: Proceedings of the Fourth International Workshop: Engineering Societies in the Agents World (ESAW'03), Imperial College London, UK, Springer-Verlag (2003) 156–169
11. Odell, J., Parunak, H., Fleischer, M.: The Role of Roles in Designing Effective Agent Organizations. In Garcia, A., Lucena, C., Zambonelli, F., Omicini, A., Castro, J., eds.: Software Engineering for Large-Scale Multi-Agent Systems, Lecture Notes on Computer Science volume 2603, Berlin, Springer (2003) 27–28
12. Odell, J., Parunak, H., Brueckner, S., Fleischer, M.: Temporal Aspects of Dynamic Role Assignment. In Giorgini, P., Muller, G., Odell, J., eds.: Agent-Oriented Software Engineering (AOSE) IV. LNCS 2935, Berlin, Springer-Verlag (2004)
13. Padgham, L., Winikoff, M.: Developing Intelligent Agent Systems. A practical guide. John Wiley & Sons Ltd (2004)
14. Ferber, J., Gutknecht, O.: A meta-model for the analysis and design of organizations in multi-agent systems. In: 3rd Int. Conference on Multi-Agent Systems (ICMAS'98), IEEE Computer Society (1998) 128–135
15. Bresciani, P., Giorgini, P., Giunchiglia, F., Mylopoulos, J., Perini, A.: TROPOS: An Agent-Oriented Software Development Methodology. Autonomous Agents and Multi-Agent Systems **2** (2004) 203–236
16. Yu, E.: Modelling Strategic Relationships for Process Reengineering. PhD thesis, Department of Computer Science, University of Toronto, Canada (1995)
17. Silva, V., Lucena, C.: From a Conceptual Framework for Agents and Objects to a Multi-Agent System Modeling Language. In: Autonomous Agents and Multi-Agent Systems. Volume 9. Springer Science+Business Media B.V. (2004) 145–189
18. Wagner, G.: The Agent-Object-Relationship Meta-Model: Towards a Unified Conceptual View of State and Behavior. Information Systems **28** (2003) 475–504
19. Cervenka, R., Trencansky, I.: Agent Modeling Language: Language Specification. Version 0.9. Technical report, Whitestein Technologies (2004) URL: http://www.whitestein.com/pages/solutions/meth.html.
20. Cervenka, R., Trencansky, I., Calisti, M., Greenwood, D.: AML: Agent Modeling Language. Toward Industry-Grade Agent-Based Modeling. In Odell, J., Giorgini, P., Muller, J., eds.: Agent-Oriented Software Engineering V: 5th International Workshop, AOSE 2004, Springer-Verlag (2005) 31
21. Alencar, E., Castro, J., Cysneiros, G., Mylopoulos, J.: From Early Requirements Modeled by the i* Technique to Later Requirements Modeled in Precise UML. In: Anais do III Workshop em Engenharia de Requisitos, Rio de Janeiro, Brazil (2000) 92–109