

Modelling a Multi-Agent System Environment

Sehl Mellouli¹, Guy W. Mineau¹, Daniel Pascot²

¹ Laval University, Computer Science Department, G1V 7P4,
Quebec, Canada
{sehl.mellouli, guy.mineau}@ift.ulaval.ca
daniel.pascot@sio.ulaval.ca
<http://www.ift.ulaval.ca/~lic>

Abstract. A Multi-Agent System (MAS) can be viewed as a software system evolving in some environment, with which it has to interact. During the MAS life cycle, many situations may occur, situations that are considered to be critical and that would have an effect on the MAS, prompting it to quickly adjust to this new situation. A design methodology of a MAS should help the designer to represent information about a changing environment and its effects on the MAS, an aspect of the modelling task which is currently lacking from agent design methodologies. We propose to add two new diagrams: an environment diagram and an agent diagram, to MAS modelling methodologies. These diagrams will conceptualise the impact of the environment on the structure of the MAS, and therefore should guide the development of the actual implementation of the MAS.

1 Introduction

A multi-agent system (MAS) consists of a number of agents that operate together in order to achieve some goals. It can be viewed as an *agent* organization (by analogy with *human* organization) or in other words, as some artificial society or organization [11]. In fact, like human organizations, a MAS evolves in a certain environment, has goals to achieve and its agents operate together to achieve these goals. So we feel that their design should be somewhat inspired from that of human organizations (based on organization theories) [6]. From the study of organization theories, we find that organization structures are viewed as metaphors (as summarized in Morgan [7]) that can be applied for the study of many sorts of organizations [2]. It is important to notice from the start that human organizations are complex and most of the time, they need more than one metaphor to be accurately described, or need some variant of a metaphor. We have chosen the four metaphors that seem to be applicable to multi-agent systems: the *machine*, *organism*, *brain* and *political* metaphors. Based on these metaphors, we have identified five key dimensions to design a human organiza-

tion that we adapt to the design of an agent organization [6]. These dimensions are: the nature of the environment, the tasks to be performed, the control, communication and collaboration relationships between agents.

Looking at agent-oriented methodologies such as Gaia [11], AUML [8] and MAS-CommonKADS [8], we find that they do not integrate any mechanism to explicitly model the environment, its evolution and its impact on the structure of the MAS [6]. They have no formal representation of the specifications of the MAS, and therefore, there is no way to implement some early validation of the behaviour of the MAS under design.

We define the environment as a set of situations that can occur during the organization life cycle, and where each situation has an impact on the structure of the organization and on its meaningful behaviour. We feel that the modelling of the environment and its impact on the organization structure of the MAS is vital to the usefulness of the system since it challenges the adaptability of the system.

We propose to add two diagrams to MAS modelling methodologies in order to conceptualise and represent the environment and its impact on the organizational structure of the MAS. These diagrams are the environment diagram and the agent diagram.

The environment diagram. It represents a state transition diagram between the different situations that the environment can undergo. A change in a situation will affect most probably the roles that the agents play and their relationships. This diagram is needed in the modelling of this dependency between environmental situations and a MAS. We define loosely the environment as being the set of the different situations that condition the behaviour of the system, and the possible sequence of occurrences of these situations. Through domain analysis and background knowledge, we determine what situations are likely to have a tremendous impact on the organization of the MAS and, for obvious complexity reasons, we restrict ourselves to them.

The agent diagram. Defined in relation to the environment diagram, it represents the structure of the agents (roles) and their relationships (control, collaboration and communication relationships), and how the environmental situations affect them.

This paper is organized as follows. In section 2, we present the limits of the Gaia [11], AUML [8] and MAS-CommonKADS [3] methodologies with regard to the modelling of the environment. Section 3 presents the environment diagram and Section 4 presents the agent diagram based on some soccer game example. Section 5 concludes.

2. Limits of agent oriented methodologies

In this section, we focus our study on three methodologies: Gaia [11], AUML [8] and MAS-CommonKADS [3]. We have chosen these methodology since Gaia is a recent one, AUML is based on UML which is a standard in object oriented development, and MAS-CommonKADS which is based on CommonKADS used to develop expert systems. We model some strategic tactics of a soccer team using Gaia, AUML and MAS-

CommonKADS. We centre our argumentation around the modelling of the environment (a set of situations that impacts the organization structure of the MAS) and show how these methodologies do not allow the designing of a MAS that could readjust its overall behaviour to an evolving environment. We also show how the five dimensions that we identified from [6] (environment, task, control, communication and collaboration) as being important to the modelling of any organization are taken into account by these methodologies. From that presentation we aim at showing that no provision was made to explicitly model the environment under these three methodologies.

2.1 The Gaia Methodology

Gaia [11] is based on three phases. The first phase is requirements statement; the second phase is analysis and the third phase is design. The first phase, requirements, is independent from any implementation. It corresponds to use cases in object oriented design. The second phase, analysis, specifies the different roles to be developed in the software system, and their interactions. It is composed of the role model and the interaction model. The former identifies the key roles that must be undertaken by agents in the system. The latter identifies the interactions between roles according to a set of protocol definitions. The third phase, design, defines the different agent types and instances (the agent model), the different services offered by each agent (service model) and the communication between the agents (acquaintance model). Each agent can be associated with a set of roles.

The basic notion in Gaia is the role. It is very important to precisely identify the key roles of the MAS since all models produced by Gaia are based on them. When modelling a soccer team, the key roles are : Goal Keeper (GK), Defender (DF), MidFielder (MF) and Striker (ST). In Gaia, each role is described by a schema (see Table 1). In this schema we:

- Give a textual description of the role: the role of the defender is to stop adverse forward players,
- Determine the different protocols and activities that will be played by the role: the defender has the activity to stop adverse players (stopPlayers) and must use protocols to guide his team mate in the overall defence strategy of the team (guideDefense),
- Determine the permissions associated to the roles. A player always has the possibility to kick the ball (kickBall),
- Define the responsibilities of the role that are composed of liveness and safety responsibilities. The liveness responsibilities are the functionalities of the role. The defender must stop adverse players (stopPlayers) and guide his team mate in defence (guideDefense). To fulfil its role, the defender must warranty that no adverse player scores a goal in his team's net. This constitutes the safety responsibilities.

According to the dimensions identified in organization theories (environment, task, control, communication and collaboration), Gaia does not deal with the environment, control and communication relationships. In fact, if we need to represent a situation

where a player is injured, there is no way to do it. Also, if a defender must become a middle fielder for tactical reasons, there is no way to represent the evolution of the role of a player during the game. Each agent has a specific role determined during the design phase and cannot evolve over time. There are no mechanisms (at least during the design phase) to foresee and model this evolution. Also, if we need to represent that an agent controls another agent, there is no mechanism to do it.

Table 1. Schema for role Defender.

Role schema:	Defender
Description:	Stops adverse forward players
Protocols and Activity	stopPlayers, guideDefense.
Permissions	kickBall
Responsibilities	
Liveness:	stopPlayers, guideDefense
Safety:	No adverse player scores

2.2 AUML

AUML is an extension of UML for modeling MAS [8]. It identifies agents by a class diagram describing the roles of each agent [1]. An agent class is identified by its name, its different roles, a state-description, actions, methods, capabilities, constraints and agent-head-automata-name. For the soccer example, if there is an agent player called A6 whose role is a DF (Defender), then the agent class (part of it) is described as follows (see Figure 1).

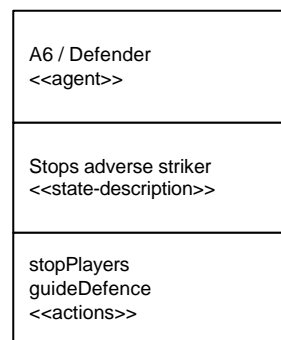


Fig. 1. A part of the agent class as defined in AUML

According to the dimensions identified in organization theories, AUML deals only with communication, collaboration and tasks (via roles) but does not provide any way

to deal with the environment and control relationships. In fact, if we need to represent a situation where a player is injured, there is no way to do it. Also, if a defender must become a midfielder for tactical reasons, there is no way to represent the evolution of the role of a player during the game. Each agent has a specific role determined during the design phase and cannot evolve over time. Also, if we need to represent that an agent controls another agent, there is no mechanism to do it. But we note that in [10], there is an attention paid to the important role played by the environment in designing a MAS. Nevertheless, there is still no way to represent the environment within AUML and no control relationships between agents in the agent class diagram.

2.3 MAS-CommonKADS

This methodology is an extension of the CommonKADS methodology [9] for agent systems using techniques of Object Oriented methodologies. It is based on the development of seven models: agent model, task model, knowledge model, organization model, coordination model, communication model and design model. These models are constructed in three stages: conceptualization, analysis and design. The first stage helps to understand the work of the future system by developing use cases and Message Sequence Charts (MSC). The analysis stage of the method is the production of all models except the design model, which will be produced at a later stage.

The agent model is constituted by agent classes. Each agent class is described by three parts. The first part is the role of the agent. The second part is the mental state and internal attributes of the agent, such as its beliefs, goals or plans. The third part is the external attributes of the agent such as the sensors and effectors of the agent. For the soccer game, the agent class for defender agents is presented as follows (see Figure 2):

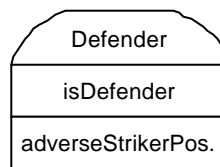


Fig. 2. MAS-CommonKADS central defender agent class

The role of the agent is defender. It must be aware of it (part of its beliefs) and must know the position of adverse strikers (through external attributes) to counter any attack.

According to the dimensions identified from organization theory, MAS-CommonKADS does not deal with the environment and the control relationship between agents. If the defender has to change its role to a midfielder for tactical reason, there is no way in MAS-CommonKADS to model that information. Also there is no way to represent the environment in which a MAS evolves, and therefore, no provision is made to formally represent agent behaviour in relation to the environment,

allowing possible failure situations to subsequently occur as a result of this under specification.

Looking at agent oriented methodologies [3, 8, 11], we find that they do not take into account the environment and its impact on the organization structure of the MAS. There is no formal way to include the description of the environment into the conceptualization of a MAS using these three methods. Since it has an influence on the structure of the MAS, we introduce, in the next section, the environment diagram that can be added to MAS modelling methodologies. The environment could affect the roles or the tasks played by the agents, the communication between two agents, the control of one agent on another or the collaboration between agents. From this, we propose an agent diagram where the agent is identified by its name, the roles it has to play and where control, communication and collaboration relationships between agents are also represented. We will also propose to add a control relation between agents, in the agent diagram, since the environment diagram may impact on it.

3. The Environment Diagram

A Multi-Agent System (MAS) is designed to achieve specific goals. A MAS is placed in an environment with which it interacts. The MAS must decide what to do and develop a strategy in order to achieve its assigned goals. For this, the MAS must have a representation or a model of the environment in which it evolves. We now ask the question: *how can the environment be modelled?*

A part of this answer is given by McCarthy in its situation calculus [5]. For McCarthy, the environment (or world) is composed of situations. A situation is the complete state of the world at an instant of time: a snapshot of the world at some instant. Each situation can be followed by an event that leads to a new situation. The situation calculus allows to infer something about future situations. Let us define $s' = \text{result}(e, s)$ where s' is a situation that can be reached from situation s when event e occurs. It is specifically designed for representing dynamically changing worlds.

Moreover, the SODA [9] and ADELFI [2] methodologies consider the environment in their process to design MAS; however, no formal representation is defined to model the environment. This is what we propose in this work. Based on McCarthy theory, and since agent oriented methodologies as presented in section 2 lack in this regard, we propose to add to these methodologies an environment diagram that models different situations of the environment and their transitions. We define the environment as a set of situations and transitions between these situations. Each situation is described by a certain number of *critical* parameters. These situations are considered to be important in the design of a MAS since they could for example affect its organization structure or prevent it from achieving its goals. The transitions between situations are defined as events. In the soccer example, many critical parameters can be identified such as: the remaining time, the score difference or the team deployment tactic. The change of the value of one of these parameters could affect the roles played by the agents and so the organization structure of the MAS. So it is important to identify

these critical parameters, and describe various situations of the environment that should have an impact on the MAS behaviour.

Each situation within the environment is described by a set of valued parameters. The set of parameters describing a situation s_i is P_i . Each parameter p_i of P_i has a single¹ value v_i . There could exist two situations s_i and s_j where $P_i \subseteq P_j$. If also, the value v_j (of an element p_j) is equal the value v_i of the element p_j in P_i ($P_i \subseteq P_j$), we will say that s_i is a generalization of s_j , and s_j is a specialization of s_i . We conclude that if a situation s_k leads to situation s_j , so it leads to situation s_i ; and if the situation s_i leads to a situation s_k , then the situation s_j leads to the situation s_k .

Formally, the environment structure S is defined as $S = \langle I, W, E, R, F \rangle$ where I is the set of initial situations, W is the set of all situations ($I \subseteq W$), E is the set of all possible events, R is a relation that represents transitions between set of situations with $R \subseteq \wp(W) \times E \times \wp(W)$ where $\wp(W)$ is the power set of W . R is a triplet (S_i, e, S_j) where S_i and S_j are in $\wp(W)$ and e is in E . The triplet (S_i, e, S_j) means that from any element of S_i , if the event e occurs, the system will subsequently be in situation S_j . All the elements of S_i , respectively S_j , are related to each other by the relation of generalization-specialization as defined above. The set F is the set of final situations that can be reached ($F \subseteq W$). There are no possible situations after the elements of F , i.e., no triplet of R has the structure (f, e, f') for whatever e in E and f' in W . The situations and their transitions define the environment diagram [6] which is a state (situation) transition diagram. It is a graphical representation of R such that it is a graph where nodes represent elements in W and where an arc e between two situations s_i and s_j in W exists if there exists two sets $S_1 \in \wp(W)$ and $S_2 \in \wp(W)$ where $s_i \in S_1$, $s_j \in S_2$ and $\langle S_1, e, S_2 \rangle \in R$.

In the soccer example, we have identified three parameters (for simplicity reasons) which, together, define situations that may impact the roles of agents. These parameters are: the Score Difference (SD), the Remaining Time (RT) and the Deployment Tactic (TC). If $SD = +v$, for $v \in \mathbb{N}^*$, it means that the team is leading by v goals. If $SD = -v$, then it is loosing by v goals. The RT parameter takes values of the form: $op\ t$, where $t \in \mathbb{N}^*$, and $op \in \{=, <, >, =\}$. Finally, TC is defined as $= n_1 - n_2 - n_3$, such that $n_1 + n_2 + n_3 =$ number of players on the field (for the same team), where n_1 is the number of defenders, n_2 is number of midfielders and n_3 is the number of strikers.

An environment diagram such as what we propose, i.e., a situation transition diagram, would help to formally represent the characteristics of the environment. In each situation, we give a name that identifies the situation, and values to the different parameters characterizing it. For the soccer example, we restrict our study to only eight situations (see Figure 3). Among these situations, there are special ones s_0 , s_1 , s_3 , s_5 and s_6 . s_0 is the initial situation identified by a bold rectangle. It is the first situation that could occur and represents the environment when the game starts. s_1 , s_3 , s_5 and s_6 are final situations; there are no possible situations after them.

Each situation is described by the three parameters SD, RT and TC. For example the situation s_0 represents a situation where the score difference ($SD = 0$) is equal to 0,

¹ We can extend the formalism for multi-valued parameter

where there remain 90 minutes of playtime (RT = 90), and where the adopted tactic is 4-4-2 (TC = 4-4-2).

According to the definition of the environment $\langle I, W, E, R, F \rangle$, the set $I = \{s_0\}$, $W = \{s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7\}$, $E = \{\text{box_goal}, \text{scores_goal}, \text{time_} < 25, \text{injured_player}, \text{recovered_player}\}$, $R = \{(\{s_0\}, \text{box_goal}, \{s_1\}), (\{s_0\}, \text{scores_goal}, \{s_2\}), (\{s_0\}, \text{time_} < 25, \{s_4\}), (\{s_2\}, \text{box_goal}, \{s_5\}), (\{s_4\}, \text{scores_goal}, \{s_3\}), (\{s_4\}, \text{injured_player}, \{s_7\}), (\{s_7\}, \text{recovered_player}, \{s_6\})\}$ and $F = \{s_1, s_3, s_5, s_6\}$.

Once the situations are identified, it is interesting to see their impact on the structure of the MAS, more specifically on the roles played by the individual agents and their relationships. For this purpose, we need to use an agent diagram [6] to represent agents, and to describe their roles and relationships. This agent diagram is drawn early on when designing a MAS. This kind of agent diagram is not defined in any of the mentioned methodologies: Gaia, AUML and MAS-CommonKADS. This is what we present in the next section.

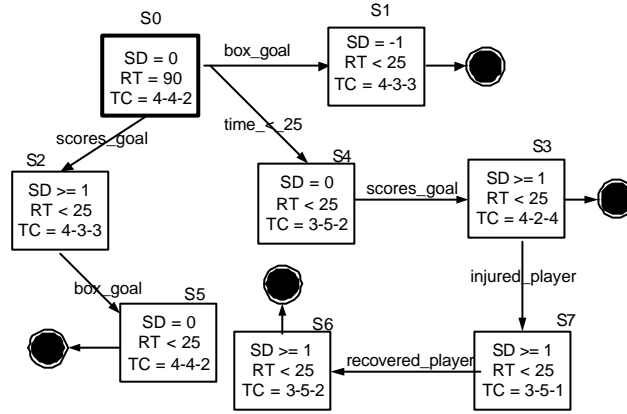


Fig. 3. The environment diagram of a soccer game.

4. The agent Diagram

The environment diagram is composed of situations that could have an impact on the organization structure of the MAS according to the roles played by the agents and their relationships (control and communication). Some of these situations may lead the MAS to a failure situation. So, we propose to define a diagram that would show the impact of any situation of the environment on the organization structure of the MAS; this is the agent diagram.

In the agent diagram, each identified agent is specified by its name and roles. Since control and communication relations² are at the centre of a MAS architecture, it is vital for the modeler to represent them. Their explicit representation favours knowledge

² For now, we will consider collaboration as a kind of communication

elicitation (more semantics is added to the agent diagram); their formal representation favours automatic validation of the system. Furthermore, the maintenance of the system will be facilitated by such an explicit representation of the system, and so will the automatic generation of the code. Each agent diagram (or subpart of it) is attached to a situation, the one identified by the label *Situation name* in the agent diagram. To see, how the agent diagram is related to the environment, we now continue with the soccer example. In the initial state s_0 , the team is organized according to a 4-4-2 deployment tactic: four defenders, four midfielders and two strikers. Any player can communicate with any other player. Of course the volume of communication is more abundant between players who share the same role. The agent diagram associated with situation s_0 is represented in Figure 4.

In Figure 4, we have omitted to represent communication and control relationships between roles where the communication is not abundant. For example, we suppose that since agent A8 controls the midfielders team mates, he is the only responsible agent for this task, and so there is no control no communication relation necessary between A7, A6 and A9.

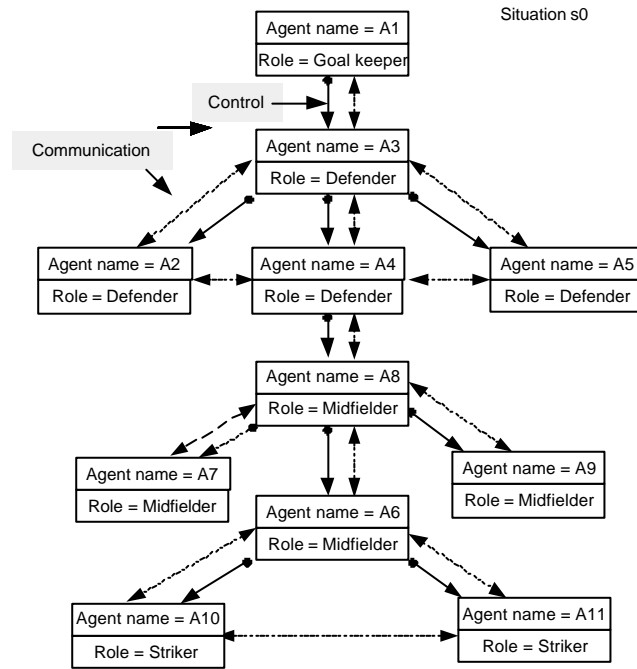


Fig. 4. The agent diagram D_1 associated with situation s_0 .

In situation s_2 , the team is organized along a 4-3-3 tactic. The A6 player changes its role to become a striker. The volume of communication between him and the other strikers will increase and decrease between him and other middle-fielders. Figure 5

represents how the MAS structure changes with situation s_2 where the role of A6 has changed to become a striker (the changes are in bold). Of course one could imagine that the drawing of Figure 5 could be done incrementally from that of Figure 4 in a MAS case tool.

By comparing the agent diagrams associated with sequentially occurring situations, the information on how the architecture of the MAS will be affected by these changes is depicted. The agent diagram structure is defined as $D_a = \{A, C, R_a\}$ where A is the set of agent instances, C is the set of relationships that can be defined between agents, and R_a is a relation that determines the relationships between agents; $R_a \subseteq A \times C \times A$. Each agent diagram is related to a particular situation in the environment diagram. We define D as a set of various agent diagrams D_a , so $D = \{D_a\}$. We define a relation $R_{sa} \subseteq W \times D$ that relates a situation W in the environment diagram to some agent diagram. In the soccer game example, according to Figure 5, (s_2, D_2) belongs to R_{sa} .

The agent diagram D_2 is defined as $D_2 = \{A, C, R\}$ where $A = \{A4, A6, A7, A8, A9, A10, A11\}$, $C = \{\text{control, communication}\}$ and $R_a = \{(A4, \text{control}, A8), (A4, \text{communication}, A8), (A8, \text{communication}, A4), (A8, \text{control}, A7), (A8, \text{communication}, A7), (A7, \text{communication}, A8), (A8, \text{control}, A9), (A8, \text{communication}, A9), (A9, \text{communication}, A8), (A8, \text{control}, A10), (A8, \text{communication}, A10), (A10, \text{communication}, A8), (A8, \text{control}, A6), (A8, \text{communication}, A6), (A6, \text{communication}, A8), (A8, \text{control}, A11), (A8, \text{communication}, A11), (A11, \text{communication}, A8), (A6, \text{communication}, A10), (A10, \text{communication}, A6), (A6, \text{communication}, A11), (A11, \text{communication}, A6), (A10, \text{communication}, A11), (A11, \text{communication}, A10)\}$.

5. Applications

The environment and agent diagrams are used to design multi-agent systems where the environment is dynamic (evolves over time) and where the organization structure of the MAS evolves over time where the roles of agents and their relations change according to the environment evolution. Typical domain applications could be computer games, military applications or Internet applications. In what follows, we give an example of a military application.

5.1 The military application

We present the example of a group of five F-16 fighters piloted by automated software agents, that have to attack and destroy a nuclear plant. This is a highly unpredictable environment since many troublesome situations may occur such as an interception attempt from enemy fighters or from the enemy's air defense system. For each situation, the fighters must reorganize themselves, as they may have to regroup in smaller squadrons to engage the enemy. The MAS guiding the attack must quickly react to any new situation and try to reach its target as much as possible, or retreat. There is much information that must be shared at all times between the fighters of the squad-

ron. It is important to identify this information, and describe various states of the environment that should have an impact on the MAS behavior. The useful information to the fighters in their various tasks are, among others:

- The position of the target: each agent must know how far the plant is and where it is located according to the position of the group.
- Position within the group: each fighter has a position within the squadron, that depends on the actual formation.
- The position of enemy fighters: knowing this position, the MAS can (re)organize itself to avoid an attack and to defend its unity. Counter-attack could also be organized.

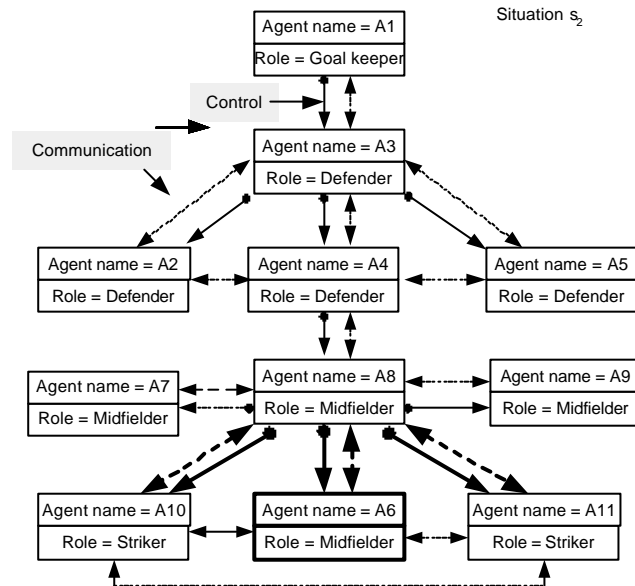


Fig. 5. The agent diagram D_2 associated with situation s_2 : the role of A6 has changed and so have the relationships with the neighbouring agents.

To see how the agent diagram is related to the environment diagram, we now continue with our example. The squadron is composed of a head pilot that gives orders to the other fighters, of an auxiliary head pilot that can replace the head pilot in case of problems, and of three additional fighters. We find three major roles: head pilot, auxiliary head pilot and fighter. Two agents have two roles: head (or auxiliary head) squadron, and fighter. Three agents only act as fighters. All agents must probe the environment to make sure that they can achieve their goal. On the second and third states of the environment diagram of Figure 6, we can observe that some (critical) parameter values change as a result of some events. In each case, the MAS must be reorganized to overcome the two related situations. In the second state, the auxiliary head pilot replaces the head pilot. This induces the designation of a new auxiliary head. For the

third state, the squadron must be split into two groups. One group will attack the enemy fighters while the other, if help is not needed by their teammates, will proceed to their target. Each environment state, being described by high impact parameters, plays a direct role in the (re)organization of the MAS. The agents play some roles in each state and we need some mechanism to model the change in agent roles. For this, we assume that we have a complete agent diagram of the MAS (not presented here for lack of space) as described in relation to the initial state of the environment diagram.

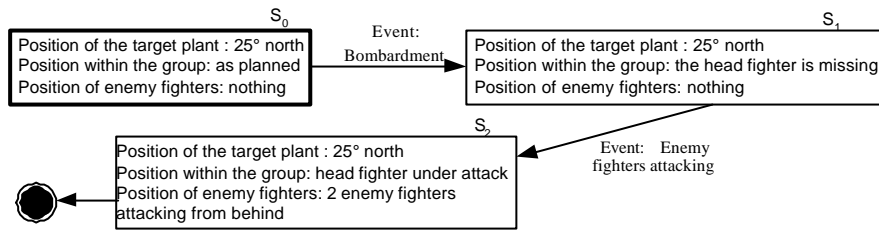


Fig. 6. Environment Diagram of the airplane attack

For example, a part of the agent diagram for the first environment state of our example is given in Figure 7.

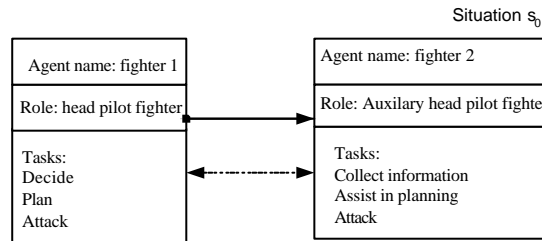


Fig. 7. Part of the agent diagram at state s_0

A part of the agent diagram for the second state is shown in Figure 8.

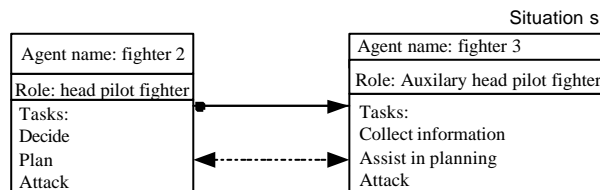


Fig. 8. Part of the agent diagram after head pilot is shot down.

To summarize, a MAS has goals to achieve and acts in an environment with which it is in perpetual interaction. The behavior of the system is affected by the evolution of the environment over time. We propose to model each state of the environment as a

set of critical parameters identified in the design phase. A changing in value of one of these parameters will lead to a new state of the environment. These parameters are critical on the correct behavior of the system. To represent the evolution of the environment over time, we propose an environment diagram which is a state transition diagram. Furthermore, the evolution of the environment should have an effect on the behavior of the system. So, we propose to attach to each environmental state, a sub-part of the agent diagram that shows the impact of this state on the organization of the MAS.

Conclusion and Future Work

Based on organization theories, we have identified five key dimensions that must be considered when modelling some organization, either human or software-based: the environment, the nature of the tasks to be performed, the communication, control and collaboration links between the different entities (agents) of the system. Through an analysis of the three most popular MAS modelling methodologies: Gaia, AUML and MAS-CommonKADS, we could pinpoint that all of them do not provide diagrams for the explicit modelling of the environment in which the MAS will evolve. If this environment is dynamic, the MAS may have some instable behaviour.

Therefore, we propose two diagrams to add to agent oriented modelling methodologies: the *environment diagram* and the *agent diagram*. The environment diagram is a state transition diagram, and connects to agent diagrams, representing the impact of each state on the organization structure of the MAS. The agent diagram states how the MAS is organized according to the roles and tasks accomplished by agents. Our two diagrams are independent from any of the methodologies and can be added to any of them as extensions.

As future work, many issues are still opened for investigation. Since the environment is unpredictable, we are not able to predict which state will occur next other than by using a probabilistic approach in the representation of the situation transition diagram. So, we can evaluate, with a certain probability, whether the MAS will well function or not. It is also important to represent the beliefs of the agents within the agent model pertaining to the information given in each environmental situation, since the information the agent has is important to evaluate whether the MAS will well function. For now, our main objective is to provide a simple graphical model to the designer so that the environment and its impact on the MAS can easily be captured, and that it could be translated into a system of logic whose model checking procedure would provide assistance to the designer by guiding the modelling process and detect situations that could MAS to a failure to achieve its goals. Cost reduction and reliability issues of MAS development (specifically when dealing with dynamic environments) are two of the main concerns driving our research efforts.

References

1. Bauer., B. UML Class Diagrams Revisited in the Context of Agent-Based Systems. In the Second International Workshop on Agent-Oriented Software Engineering (AOSE-2001), Montreal, Canada, May 28- June 01. 2001. pp 1-8.
- 2., Bernon., C., Gleizes., G., Peyruqueou., S., Picard., G. ADELFI, a Methodology for Adaptive Multi-Agent Systems Engineering. Workshop Notes of the Third International Workshop Engineering Societies in the agents world, 16-17 septembre 2002, madrid, spain, pp. 21-34.
3. Carlsen., S., and Gjersvik., R. Organizational Metaphors as Lenses for Analyzing Workflow Technology. In proceedings of the international ACM SIG GROUP conference on Supporting Group Work: the Integration Challenge, Phoenix, AZ USA, November 16-19, 1997
4. Iglesias., C. A., and Garijo., M., and Gonzales., J. C., and Velasco., R. Analysis and Design of Multi-Agent Systems using MAS-CommonKADS. In proceedings of the AAAI'97 Workshop on agent Theories, Architectures and Languages, Providence, USA, July, 1997.
5. McCarthy., J. Situation Calculus With Concurrent Events and Narrative. <http://www-formal.stanford.edu/jmc/narrative/narrative.html>, 2000.
6. Mellouli., S., Mineau., G., and Pascot., D. The Integrated Modeling of Multi-agent Systems and their Environment. In Proceedings of the First International Conference on Autonomous Agents and Multi-Agent Systems 2002 (AAMAS 2002), 15-19 July 2002, Bologna, Italy. pp 507-508.
7. Morgan., G. Images of Organization. SAGE Publication. 1997. ISBN: 0-7619-0634-7.
8. Odell., J., Van Dyke Parunak., H., and Bauer., Bernhard. Extending UML for Agents. Proceedings of the Agent-Oriented Information Systems Workshop at the 17th National Conference on Artificial Intelligence, Gerd Wagner, Yves Lesperance and Eric Yu eds., Austin, Tx, pp 3-17, AOIS Workshop at AAAI 2000.
9. Omicini., A. SODA : Societies and Infrastructures in the Analysis and Design of Agent-based Systems. Agent Oriented Software Engineering Workshop, 10 June 2000, Limerick, Ireland, pp 185-193.
10. Van Dyke Parunak., H., Odell., James. Representing Social Structures in UML. Proceeding of the Second International Workshop on Agent-Oriented Software Engineering (AOSE-2001) at the 5th International Conference on Autonomous Agents, pp 17-24, Montreal, Canada, May, 2001.
11. Wooldridge., M., and Jennings., N. R., and Kinny., D. The Gaia Methodology for Agent-Oriented Analysis and Design. Journal of Autonomous Agents and Multi-Agent Systems 3 (3).