AGRE: Integrating Environments with Organizations

Jacques Ferber, Fabien Michel, José-Antonio Baez-Barranco

AGRE: Integrating Environments with Organizations

J. Ferber, F. Michel, and J. Baez

LIRMM, CNRS 161 rue Ada
34392 Montpellier Cedex 5, France
{ferber, fmichel, baez}@lirmm.fr

Abstract. This paper presents an extension of the AGR (Agent-Group-Role) organizational model, called AGRE (AGR + Environment), which includes physical (or simply geometrical) environments. This extension is based on the concept of a space which can be seen either as a physical area or as a social group, and on a clear distinction between an agent and its mode, i.e. the way it appears and interacts into a space with other agents. A notation which encompasses both social and physical environments is given.

1 Introduction

Recently a particular interest has been given to the use of organizational concepts within multiagent systems (MAS) where the concepts of 'organizations, 'groups, 'communities, 'roles', 'functions', etc. play an important role [1, 2, 3, 4, 5, 6].

The use of organizations provides a new way for describing the structures and the interactions that take place in MASs. The organizational level, the way organizations are described is responsible for the description of the structural and dynamical aspects of organizations. It stands for an abstract representation of concrete organizations, i.e. as a specification of the structural and dynamical aspects of a MAS. The organizational level describes the expected relationships and patterns of activity which should occur at the agent level and therefore it defines the constraints and potentialities that constitute the horizon in which agents behave.

1.1 Organization Centered General Principles

The principles for designing true organizational centered multiagent systems as explained in [7] are the following:

Principle 1: The organizational level describes the "what" and not the "how". The organizational level imposes a structure into the pattern of agents activities, but does not describe how agents behave. In other terms, the organizational level does not contain any "code" which could be executed by agents, but provides specifications, using some kind of norms or laws, of the limits and expectations that are placed on the agents behavior.

D. Weyns et al. (Eds.): E4MAS 2004, LNAI 3374, pp. 48–56, 2005.

Principle 2: No agent description and therefore no mental issues are provided at the organizational level. The organizational level should not say anything about the way agents would interpret this level. Thus, reactive agents as well as intentional agents may act in an organization. In other words, ant colonies are as much organizations as human corporations. Moreover, seen from a certain distance, or using an intentional stance it is impossible to say if the ants or the humans are intentional or reactive. Thus, the organizational level should get rid of any mental issues such as beliefs, desires, intentions, goals, etc. and provide only descriptions of *expected* behaviors.

Principle 3: An organization provides a way for partitioning a system, each partition (or group) constitutes a context of interaction for agents. Thus, a group is an organizational unit in which all members are able to interact freely. Agents belonging to a group may talk to one another, using the same language. Moreover, groups establish boundaries. Whereas the structure of a group A may be known by all agents belonging to A, it is hidden to all agents that do not belong to A. Thus, groups are opaque to each other and do not assume a general standardization of agent interaction and architecture.

These principles are not without consequences:

1. An organization may be seen as a kind of dynamic framework where agents are components. Entering a group/playing a role may be seen as a plug-in process where a component is integrated into a framework.
2. Designing systems at the organizational level may leave implementation issues, such as the choice of building the right agent to play a specific role, left opened.
3. It is possible to realize true "Open Systems" [8] where agents architecture is left unspecified.
4. It is possible to integrate multiple aspects of a system and make them interact together, considering each group as a "black boxes" which represents a specific perspective of a system: what happens in a group cannot be seen from agents that do not belong to that group.

However, the general concept of environment which is one of the main concepts of a MAS [9] is not taken into account with these principles. If we consider an environment as *the conditions under which an entity exists* [10], these principles provide only support for social environments [11, 7]. They do not say anything about physical (or even simply geometrical) environments, and entities which are not agents (e.g. documents, objects to be grasped, etc.) are not considered. Several attempts have been made to integrate environments with AGR (Agent-Group-Role). In [11] a model has been proposed. But this model has not really been analyzed in detail and nothing has been said about the way to practically integrate environments with groups. Thus, it is still necessary to extend the AGR model to take physical environments into account, without losing the expressiveness and simplicity of this model.

1.2 Content of the Paper

Section 2 will summarize the main concepts of AGR and of the UML meta-model one can use to implement AGR in various platforms. Section 3 will present the AGRE (Agent-Group-Role-Environment) model which is an extension of the AGR model and which allows for the design of social and physical environments in an integrated way. We will see that both groups and areas (parts of the physical environments) may be considered as specializations of more general spaces in which agents are embedded through what we call 'modes'. This presentation will include the basic concepts and the notation one can use to describe both social and physical environments. Section 4 will draw conclusions and present some perspectives.

2 AGR: A Basic Model of Organization Centered MAS

In order to show how these principles may be actualized in a computational model, we have presented the Agent-Group-Role model, or AGR for short, also known as the Aalaadin model [4] for historical reasons, which complies with the organization centered general principles that we have proposed in the previous section. The AGR model is based on three primitive concepts: Agent, Group and Role that are structurally connected and cannot be defined by other primitives. They satisfy a set of axioms that unite these concepts.

- **Agent:** an agent is an active, communicating entity playing *roles* within *groups*. An agent may hold multiple roles, and may be a member of several groups. An important characteristic of the AGR model, in accordance with the principle 2 above, is that the architecture of an agent is left unspecified, and that no cognitive abilities are assumed. Thus, an agent may be as reactive as an ant, or as clever as a human.
- **Group**: a group is a set of agents sharing some common characteristic. A group is used as a context for a pattern of activities, and is used for partitioning organizations. Following principle 3, two agents may communicate if and only if they belong to the same group, but an agent may belong to several groups. This feature will allow the definition of organizational structures.
- **Role:** the role is the abstract representation of a functional position of an agent in a group. An agent must play a role in a group, but an agent may play several roles. Roles are local to groups, and a role must be requested by an agent. A role may be played by several agents.

The AGR meta-model is represented in Fig. 1 in UML.

A **group type** (or **group structure**), defined at the organizational level, describes a particular type of **group**, how a group is constituted, what are its roles, its communication language, and the possible norms that apply to this type of group. A group is thus a kind of instance of a group type. A **role type** is part of the description of a group structure and describes the expected behavior of an **agent** playing that **role**. Role types may be described as in Gaia [12] by attributes such as its cardinality (how many agents may play that role). It is also

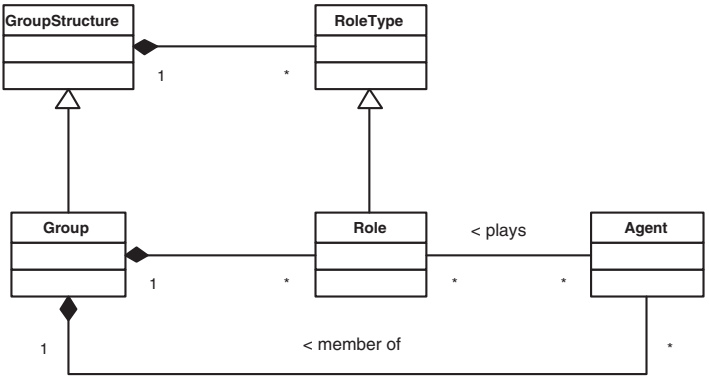

Fig. 1. The UML meta-model of AGR

possible to describe *interaction protocols* and *structural constraints* between roles (not viewed in the figure, but presented in [7]). A structural constraint describes a relationship between roles that are defined at the organizational level and are imposed to all agents.

A role, which is part of a group, is an instance of a role type defined for an agent. We can see the role as a representative of an agent or as a kind of social body that an agent plays when it is a member of a group, the interface by which an agent is able to communicate and more generally to perform actions in a group.

Several notations may be used to represent organizations. In [7] we have proposed a set of diagrams to represent both static and dynamic aspects of organizations.

3 AGRE: Integrating Environments to AGR

First, we will give a general overview of AGRE, and then we will present the principles on which this model is based.

3.1 Description of AGRE

In this section we provide an extension of the AGR model to take into account both physical and social environments. This extension, called AGRE, for Agent-Group-Role-Environment, is based on the idea that agents are situated in domains, that we call **spaces**. A space may be physical (i.e. geometrical) or social. Geometrical spaces will be called **areas**, and social spaces represent AGR **groups**. There may be other kinds of spaces but we will not discuss them here.

Agents are situated in spaces and are able to perform actions in these spaces through **modes**. A mode should be seen as the manifestation of an agent in a specific domain, as its way of existence and appearance in a space. A mode describes the agent's location and the way it perceives and acts within a space. A mode in an area is called a **body**, and a mode in a group is called a **role**.

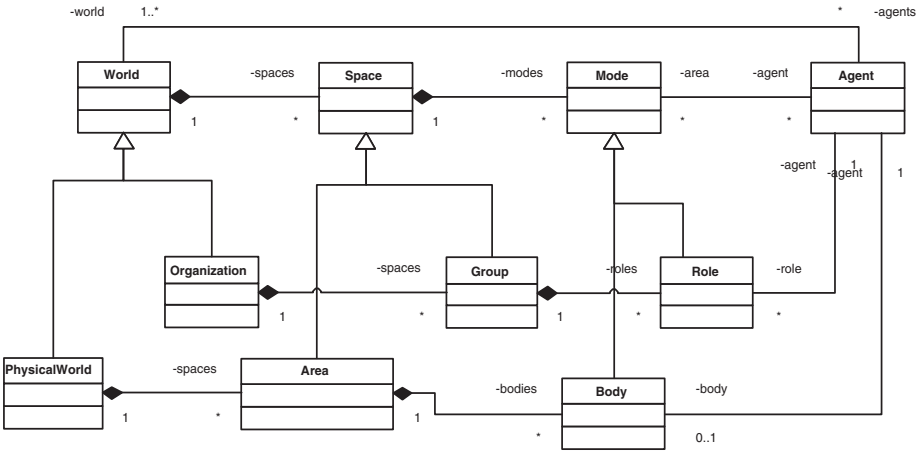

Fig. 2. The UML meta-model of AGRE at the concrete level

Spaces are regrouped in **worlds**. A world is simply a collection of spaces of the same kind. For the moment we will only consider two type of worlds: organizations which represents social environments and are composed by sets of groups, and physical worlds which represent physical environments and are made of areas. It could be possible to consider other worlds: worlds for displaying agents in a specific manner, worlds made of places for describing agent mobility, etc. In this paper we will consider only two types of worlds: physical worlds made of areas, and organizations made of groups.

Figure 2 shows the simplified UML diagram which represents the relations between world, spaces, areas, groups, modes, bodies and roles at the concrete level. For each concept at the concrete level (e.g. group, area, role, body) there is a related abstract concept at the organizational level (e.g. group structure, area structure, role description, mode description), except for the concept of agent which does not have any corresponding concept at the organizational level.

The aggregation relation that links space to mode, are overridden by the same relation that links area to body and group to role. In the same idea, the aggregation which relate world to space is overridden by an aggregation which link organization to group on one hand, and physical world and area on the other hand.

A world proposes the required primitives that are necessary for an agent to enter a space and get its mode. Because the mode is the only way through which an agent can act in a space, it is necessary for an agent to have the ability to enter a space. This is done through the world which gives the necessary primitives that an agent needs for entering a space and acquire a mode. An agent may live in several worlds at once. Worlds are used as starting points for agents to enter groups and areas. When an agent is created, it must register to a world. For instance, a social agent, i.e. an agent that plays roles in groups, has

to register first to an organization. Let us suppose that the agent is registered to an organization `o`, then, to enter a group, the agent may use the primitive[1]:

```
Role r = o.requestRole(GroupName, RoleType, RoleName, a);
```

which gives it the ability to request the entrance to a group for playing the role of the type `RoleType` with the authorization `a`. If this is possible, the agent will get a role through which it will have the possibility to act within this group. All the skills associated to this role will be available to it, and naturally it will be able to send messages to agents within this group, using a primitive of the following kind:

```
r.sendMessage(RoleName,Message);
```

which expresses a request for the role to send a message to the agents having the role `RoleName`. One can see that agents are only referenced by their `RoleName` in a group. There is no way to send messages to the "real" agent, because formerly there are no agents in a group: only roles by which agents are connected to groups. To get an idea of the concepts involved, a role instantiated in a group is like a registered login name in an internet e-commerce site. An agent may act only through its login name which constitutes its mode. Of course there are primitives by which an agent may acquire information about the different agents related to a specific `RoleType` such as the following:

```
List<RoleName> l = r.getAgentsWithRoleType(RoleType);
```

which will return a list of all the local `RoleName` (i.e. all the logins) of all the agents that play a specific `RoleType` in the group where the agent has the role `r`.

The same idea applies to physical worlds and areas. To enter an area, an agent must register to a physical world `p` which contains this area and use a primitive such as

```
Body b = p.requestBody(AreaName, BodyType, Location, a);
```

which gives it the ability to enter an area using a specific `BodyType` at the location Location, with the authorization `a`. A reference to a body is returned. Then, according to the capabilities of the body, the agent may move, grasp things, etc. with commands like the following:

```
b.move(30,10);
```

Figure 3 shows the graphical notation used to represent items in the AGRE model, which is an extension of the "cheeseboard" notation proposed in [7].

The following figure shows a snapshot of two agents playing roles in groups and having bodies in areas. One can see that agent `A1` plays two different roles in `G2` and one role in `G1`, while agent `A2` plays only one role in `G1`. Both agents have a body in area `A1`. These bodies are instances of the body type `B`.

[1] We use a Java-like notation to give an idea of how such a model may be practically realized. But obviously, this could be expressed in any language.

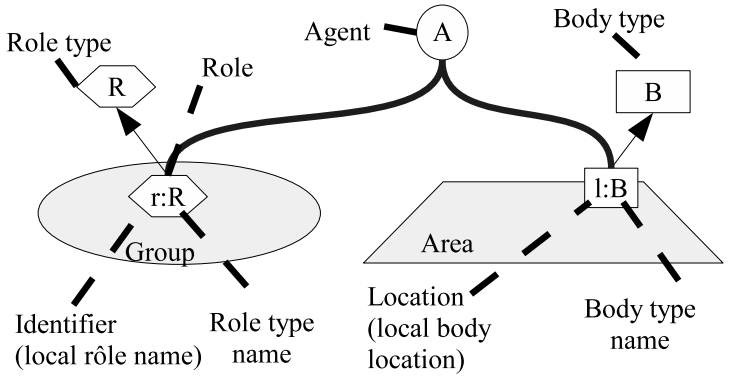

Fig. 3. Notation used to represent an instance diagram of agents, areas and groups

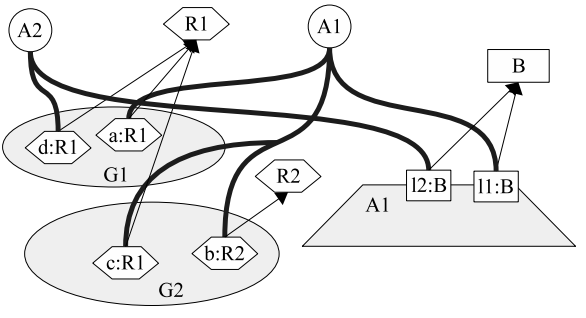

Fig. 4. A simple example with two agents playing roles in groups and having bodies in areas

3.2 Principles of AGRE

In this section, we will summarize the principles applying to the AGRE model that we have introduced in previous sections.

Principle 1: a multiagent world is constituted of agents (individuals) that may perceive and act in spaces and manifest their existence through their mode. This statement, when it is reduced to social world may be expressed as the following:

> **Principle 1a:** An organization is a kind of world in which spaces are groups and in which agents perceive and act through their roles.
>
> **Principle 1b:** A physical world is a kind of world in which spaces are areas and in which agents perceive and act through their bodies.

Principle 2: an agent may belong simultaneously to a social world and to a physical world. The number of roles an agent may play is not restricted, but the number of bodies an agent may possess is constrained by obvious conditions that an agent can act in a world through only one body.

Principle 3: An agent may possess several modes of different kinds. The constraints about the number of modes an agent possesses depend of the world in which it has been registered.

> **Principle 3a:** an agent may have several roles in a group and may belong to several groups.
>
> **Principle 3b:** an agent may possess only one body for a given world. This principle expresses the fact that an agent may not live in two different places at the same time. However, this constraint may be relaxed, when two areas overlap.

Let us note by `s: m.op(a1,..,an)` the action of an agent with mode `m` executing the operator `op` with args `a1,..,an` in space `s`. Let us also note the following assertion:

- `mode(x, m, s)[role(x,r,g)]` : the agent `x` has a mode `m` in space `s` [has a role `r` in group `g`]
- `type(m, M) [type(r, R)]` : the mode `m` is of type `M` [the role `r` is of type `R`]
- `op(o(a1,..,an), M)` : the operator `o(a1,..,an)` is defined in mode[role] type `M`

Principle 4: The mode is the way for an agent to act in a space. An agent may act in a space if one of its mode in the space gives it the power to do so. Thus, an agent `a` may perform an action `u` in a space `s`, if there exist a mode `m` of `a` in `s` such that the type of `m` (its ModeType) allows for `u`.

$$\mathtt{s} : \mathtt{m.o}(\mathtt{a1}, .., \mathtt{an}) \Rightarrow \exists \mathtt{x} : \mathtt{Agent}, \mathtt{mode}(\mathtt{x}, \mathtt{m}, \mathtt{s}) \wedge \mathtt{type}(\mathtt{m}, \mathtt{M}) \wedge \mathtt{op}(\mathtt{o}(\mathtt{a1}, .., \mathtt{an}), \mathtt{M})$$

> **Principle 4a:** an agent may communicate only if it plays a role in a group. This communication is performed through its role, and the receiver is necessarily another role within the same group.
>
> $$\mathtt{s} : \mathtt{r1.send}(\mathtt{r2}, \mathtt{msg}) \Rightarrow \exists \mathtt{x}, \mathtt{y} : \mathtt{Agent}, \mathtt{role}(\mathtt{x}, \mathtt{r1}, \mathtt{g}) \wedge \mathtt{role}(\mathtt{y}, \mathtt{r2}, \mathtt{g})$$

4 Conclusion

We have proposed a simple model, AGRE, which is an extension of AGR and which integrates smoothly physical and social environments. This extension respects the main principles of organizational centered multiagent systems. Both groups and areas may be seen as specializations of spaces in which agents may act through modes. Because roles and bodies are modes, it is possible to consider social and physical embedding as a general manner for an agent to manifest itself in a world. We have proposed some notations for this model and a set of principles which describe the basic elements for understanding AGRE.

These concepts may be used for practical implementations. The MadKit platform [13] that we have designed is built around the AGR model. Since its first release, hundreds of users (thousands of downloads) have been able to use these

organizational concepts (presented in a less rigorous way than here) to build applications in various areas. We plan to extend the MadKit platform to integrate this new AGRE model.

References

1. Jennings, N.R.: On agent-based software engineering. Artificial Intelligence **117** (2000) 277–296
2. Zambonelli, F., Van Dyke Parunak, H.: From design to intention: signs of a revolution. In: Proceedings of the first international joint conference on Autonomous agents and multiagent systems, ACM Press (2002) 455–456
3. Demazeau, Y., Costa, A.R.: Populations and organizations in open multi-agent systems. In: 1st National Symposium on Parallel and Distributed AI (PDAI96). (1996)
4. Ferber, J., Gutknecht, O.: A meta-model for the analysis and design of organizations in multi-agent systems. In: Proceedings of the 3rd International Conference on Multi Agent Systems, IEEE Computer Society (1998) 128–135
5. Odell, J., Parunak, H.V.D., Breuckner, S., Fleischer, M.: Temporal aspects of dynamic role assignment. In Giorgini, P., Muller, J.P., Odell, J., eds.: Agent-Oriented Software Engineering IV: 4th International Workshop, Aose 2003. Lecture notes in computer science LNCS, Melbourne, Australia, Springer Verlag (2003) 47–59
6. Gasser, L.: Perspectives on organizations in multi-agent systems. In Luck, M., Mak, V., tpnkov, O., Trappl, R., eds.: Multi-Agent Systems and Applications : 9th ECCAI Advanced Course ACAI 2001 and Agent Link's 3rd European Agent Systems Summer School, EASSS 2001. Volume 2086 of LNAI., Prague, Czech Republic, Springer-Verlag Heidelberg (2001) 1–16
7. Ferber, J., Gutknecht, O., Michel, F.: From agents to organizations: an organizational view of multi-agent systems. In Giorgini, P., Muller, J.P., Odell, J., eds.: Agent-Oriented Software Engineering IV: 4th International Workshop, Aose 2003. Lecture notes in computer science LNCS, Melbourne, Australia, Springer Verlag (2003) 185–202
8. Hewitt, C.: Offices are open systems. ACM Trans. Inf. Syst. **4** (1986) 271–287
9. Weyns, D., Parunak, H.V.D., Michel, F., eds.: The First International Workshop on Environments for Multiagent Systems E4MAS. In Weyns, D., Parunak, H.V.D., Michel, F., eds.: Environments for Mutiagent Systems. Volume 3477 of LNAI., Springer (this volume, 2005)
10. Weyns, D., Parunak, H.V.D., Michel, F., Holvoet, T., Ferber, J.: Environments for multiagent systems: State-of-the-art and research challenges. In Weyns, D., Parunak, H.V.D., Michel, F., eds.: Environments for Mutiagent Systems. Volume 3477 of LNAI., Springer (this volume, 2005)
11. Parunak, H.V.D., Odell, J.: Representing social structures in uml. In: Agent-Oriented Software Engineering II. Volume 2222 of Lecture notes in computer science LNCS., Berlin, Springer (2002) 1–16
12. Zambonelli, F., Jennings, N.R., Wooldridge, M.: Developing multiagent systems: The Gaia methodology. ACM Transactions on Software Engineering and Methodology (TOSEM) **12** (2003) 317–370
13. Gutknecht, O., Ferber, J., Michel, F.: Integrating tools and infrastructures for generic multi-agent systems. In: Proceedings of the fifth international conference on Autonomous agents, AA 2001, ACM Press (2001) 441–448