AOPOA
Organizational Approach for Agent Oriented Programming

Enrique González, Miguel Torres

Grupo de Investigación SIDRe, Pontificia Universidad Javeriana,Bogotá, Colombia

Keywords: Agent-Oriented Programming, Agent Design Methodology, MultiAgent Systems.

Abstract: This paper presents AOPOA, an agent oriented programming methodology based in an organizational approach. The resulting multiagent system is composed by a set of active entities that aim to accomplish a well-defined set of goals. This approach allows to design complex systems by decomposing them into simpler ones. The organizational approach makes it easier to perform an iterative and recursive decomposition based in the concept of goal; and at the same time to identify the interactions between the entities composing the system; at each iteration an organization level is developed. During the analysis phase, tasks and roles are detected. During the design phase, the interactions are characterized and managed by means of cooperation protocols. At the final iteration, the role parameterization is performed, which allows to specify the events and actions associated to each agent. Finally, the deploy of the agent instances is determined allowing redundancy to achieve the requirements of the system.

1 INTRODUCTION

The rational agent concept appeared in AI as a new conceptual and practical approach for designing and building intelligent systems. Rational agents are seen as units acting to attain a set of well-defined goals. In a more general approach, an agent appears as entity encapsulating data, knowledge and behaviour, capable to perform a task in an autonomous and proactive way. Normally, agents do not work alone, but form groups to attain global goals, which can not be achieved by one agent alone; this group of cooperating agents form a MultiAgent System (MAS). MAS have appeared as a new way to build systems that solve complex problems.

Building a MAS implies a different vision of the design process, it must include and take advantage of the agent's intrinsic characteristics. A potential advantage in applying the Agent Oriented Programming (AOP) paradigm is that it facilitates the development of complex applications. This design approach is well-suited to subdivide a complex problem into simpler ones, which are solved by active entities, the agents; besides, it offers the possibility to design a modular solution that allows a more structured and coherent management of global system complexity. However, it is clear that when dividing the system, there are multiple problems to solve to get the agents to work together in a cooperative way in order to fulfil the system's goals. Thus, AOP must provide methods not only to assign responsibilities to agents, but also to identify and manage the interactions between them.

The development of methodologies to perform in an efficient way the analysis and design of an agent based system is a research field still open; even if there are some already proposed alternatives. This paper presents AOPOA, an AOP methodology based on an organizational approach. In this approach, a MAS is perceived as an organization. As in other previous approaches, complexity management is attained through an organizational decomposition of the system in simpler parts. The key point of the approach of AOPOA is that at the same time agents are designed in a structured and progressive way, relationships between them are automatically established and characterized.

González E. and Torres M. (2006).
AOPOA - Organizational Approach for Agent Oriented Programming.
In *Proceedings of the Eighth International Conference on Enterprise Information Systems - SAIC*, pages 75-80
DOI: 10.5220/0002453500750080

2 AOP METHODOLOGIES

The problem of building a system based on an AOP methodologies has already been studied (Wooldridge M. 2000) (Alonso F. 2004). In the development process of AOPOA some existing methodologies were taken into account, and they were analyzed to find the more relevant characteristics that a good methodology must posses. There are a great variety of methodologies for designing Multiagent Systems, some of them are extensions or are based in other design models or methodologies, inheriting its benefits and its failures. Alonso et.al (Alonso F. 2004) proposed a taxonomy to organize such methodologies in three categories: methodologies based upon the Object Oriented paradigm; methodologies based on Agents itself; methodologies based on Knowledge Engineering.

The methodologies based upon the Object Oriented Software paradigm have certain problems such as a generic analysis model, and most of them do not cover the social structure of the system and the environment characteristics. Under these methodologies the agent is a complex object, reducing the level of real abstraction provided for agents. Such methodologies also use some models and views (i.e. UML diagrams) and some of the techniques proposed in Software Engineering common processes.

The methodologies based on Agents are based upon abstraction of social levels such as groups and organizations. These methodologies are strong in the first steps of the specification and design level of the MAS, but as in the previous methodology they lack of a generic analysis model which can be used to assess if any given MAS approximation is appropriate for the given problem. These methodologies also present different levels of abstraction of the MAS such as: the internal structure of the agents, the structure of the interactions among agents, and the social structure of the different groups of agents.

Finally, the methodologies based on Knowledge Engineering are characterized by the identification, acquisition and modeling of the knowledge used by the agents in the MAS. The most representative methodologies of this category are extensions of the CommonKADS (Schreiber G. 1999) methodology for developing Knowledge Based Systems. For instance, MASCommonKADS also appropriates some object oriented design and analysis techniques. In next paragraph some of the most relevant MAS methodologies are presented.

Tropos is an agent oriented methodology (Penserini, L. 2004), is based upon two basic concepts: the notion of an agent who uses plans in order to fulfill goals and the covering of the early and late requirements analysis. Prometheus (Padgham, L. 2002) is detailed and complete, and covers all the steps since the requirement analysis process until the MAS implementation using the JACK framework (Howden N. 2001). Odell presents a methodology based on the object oriented software paradigm, and it uses as a basis the UML diagrams for the MAS representation (Odell J. 2004); using Metamodels in order to describe the MAS, and its elements. GAIA (Wooldridge M. 2000) is another object oriented based methodology, which uses the initial concept of organization and sub organization, modeling of the environment of the MAS, role modeling and interaction model among roles. GAIA does not present particular techniques for implementation or for requirements elicitation. MASE is a MAS development methodology (DeLoach S. 2000), which goes from initial specification until the actual implementation of the MAS. The process of capturing goals produces a goal hierarchy that is used to also identify Use Cases; which are then used to generate sequence diagrams among roles.

To summarize, the fundamental characteristics in the development of a methodology are: to identify the goals that must be attained in order to solve the problem; to assigning them to roles, which will perform the necessary tasks to achieve the objectives; and then, to establish the social aspects, such as interaction and cooperation mechanisms required to get the desired social behaviour; finally, the assignation of the roles to the agents which will conform the system. In order to achieve this process, there are different alternatives, as has been shown.

3 MULTIAGENT SYSTEMS

An agent can be defined as an entity that perceives its surrounding environment through sensors, and which also responds or acts in that environment through effectors (Rusell N. 2003). Agents respond to events coming from the environment or from other agents; agents select the most adequate action that leads the agent to achieve its own goals.

A MultiAgent System (MAS) is a set of organized agents; they interact in a cooperative way to reach in a collective way the global goals of the system (Ferber J. 1999). A MAS can be viewed as a organization of agents in which interactions are the origin and product of the system's persistence and dynamics through time. Cooperation is an important issue in agent interactions, and it is composed by three characteristic elements: collaboration, coordination and conflict resolution. Collaboration is required when the agent's abilities and resources are not enough for the agent to accomplish goals. Coordination is related to the order in which the system's tasks must be performed. When resource conflicts arise, they must be solved; usually agents have to negotiate or to apply rules that will impose certain social restrictions. Finally, agent communication is the support for cooperation. In practice, any cooperation technique can be modelled by an interaction protocol, which defines an ordered set of communication acts between the implicated agents.

In a MAS, a role makes reference to an abstract entity, whose function is to achieve a set of goals; in other words, a role defines a set of tasks. The accomplishment of such a task depends on the abilities, resources and bindings between the entity, its environment and other roles. The objective then is to build an efficient system, in which the abilities and resources of a role are as different as possible from others, in order to avoid redundancy of abilities and reduce resource conflicts.

An organization is an array of relationships among individuals; that can be perceived as a single unit or system. The organizational structure is defined in an abstract way by a set of roles, which can be assumed by instantiated agents, and a set of relationships between such roles. Each organization can be perceived as a set of organizations; each of them can be decomposed in a recursive way into lower-level organizations.

4 GOAL DECOMPOSITION

In the AOPOA methodology, there are two main processes: analysis and design; each one of them is applied iteratively until it is determined that it is no longer necessary to decompose the goals and roles already identified. Based on these organizational perspective, the system is viewed as an organization that can be recursively divided. In each iteration, the analysis of an organizational level is performed. In this way, a MAS can be modelled as an organizational tree, where the root node represents the whole system, child nodes represent progressive role decompositions, and leaf nodes represent final roles that can be instantiated as the agents of the functional system. The iterative process of recursive decomposition ends when it is considered that the complexity of all the final roles is low enough. Once this point is reached, a final iteration is performed, to make the detailed design of the agents that will represent the existing roles in the final system.

The key concept for identifying a role is the association of an autonomous entity to a set of goals. In AOPOA, the notion of goal is the foundation, which allows the decomposition process of a complex role into simpler ones. In fact, this sub-role generation process implies two activities:

- decomposition of the associated goals of a role into simpler ones.
- grouping this new sub-goals to create new less complex roles.

The process begins by modelling the MAS as a single first level unique role. The first role's goals and environment interactions are derived directly from the requirements that where identified for the entire system. This general system's goals must be divided into sub-goals, which can be assigned to less complex derived roles.

Another AOPOA key concept is the use of the notion of interaction. Two agents interact when they join efforts and abilities to reach a goal; or when they have to synchronize its activities; or when agents share resources during the execution of a task. Every interaction is represented by an abstract relationship, between the involved roles, called cooperation bind. During the iterative decomposition phase, the identification of interactions by goals is performed directly. Every time a complex goal is divided, if some of the resulting roles include a sub-goal of the former one, these roles will probably need to interact in order reach the complex goal. Also through the decomposition, the identification of the resources required by the new derived roles is performed; when a shared resource is detected, the implicated roles will probably need to interact.

As can be observed, different cooperation binds automatically arise because of the potential interactions that can appear between roles.

During the design phase, every cooperation bind is analysed; depending in the type of interaction, an adequate cooperation technique can be chosen. As was previously explained, the implementation of a cooperation technique is made through an interaction protocol. In summary, from the bind identification, it is possible to determine, in a structured way, the adequate sequence of interchanged messages between agents.

An important characteristic of cooperation binds is that through the role decomposition, the associated binds of a role are inherited by the resulting sub-roles. In fact, the binds existing between roles in an organizational level should be assumed by the sub-roles in the next organizational level. Figure 1 illustrates the previous concepts; The roles are represented as circles, and binds among them are arrows. In iteration i there are two identified roles. In iteration i+1 the sub-roles are included inside the roles identified in the previous level. In iteration i only two binds where identified, and in iteration i+1 those binds remain and are extended to some of the new sub-roles. New interactions among roles in the same organizational level can also appear, for instance the bind between roles 11 and 12.

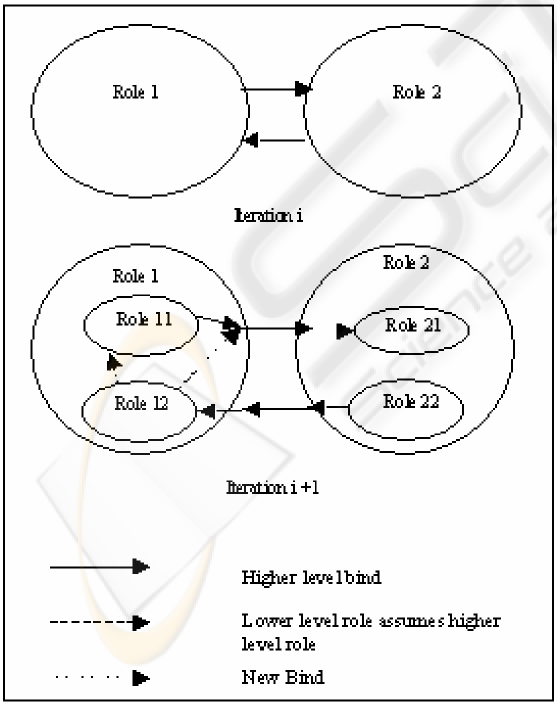

Figure 1: Inheritance of cooperation binds.

5 AOPOA PHASES

Requirements are the basis upon which the system's general goals are identified. Once the problem is well understood, it is examined in order to decide if an agent-oriented approach is well-suited for the problem at hand.

5.1 Analyse Phase

In the first iteration, the general objectives of the system are analysed and divided. In each iteration, tasks are defined for each goal to reach; a task is characterized by the set of resources and abilities required to attain the goal. Based on these tasks, a grouping process is performed to determine the ideal set of required roles; each group of tasks is assimilated to a new role. Notice that only the tasks derived from a role are grouped to obtain its corresponding sub-roles; as a consequence, the grouping procedure is applied to small sets of tasks.

The grouping process aims to qualify and group tasks applying three different criteria:

- No opposite roles are assigned to the same role.
- There are few resource conflicts among the new generated roles.
- The presence of abilities' redundancy should be reduced among roles.

The procedure to find the evaluate a generated grouping is as follows:

1. Assign tasks in a way that a set of groups *NG* can be produced. Each group of tasks represents a possible candidate role to create. *NG* is the total number of groups identified.

2. Evaluate the criteria for each group of tasks. The applied criteria for a group *g* are: *Aog*, *Ahg* and *Arg*; which respectively measure the criteria for goals, abilities and resources. In order to perform this evaluation equation 1 is applied.

$$A_{jg} = \frac{NI_{jg}}{N_{jg}} \quad \text{(Eq. 1)}$$

Ajg is the classification value of the indicator for a criteria *j* for the evaluated group of tasks *g*; *j* can have the values: *h*, *o* and *r*; *g* is the identifier of the group that is being evaluated, and its value ranges from 1 to *NG*.

NIjg is the number of intersections of type *j*; if *j=o* the amount of goals of the different type in group *g*, if $j=h$ the quantity of redundant abilities that are present in other groups; if $j=r$ the required shared resources that are present in other groups of tasks.

Njg is the total number of elements of type j inside group g, i.e. objectives, abilities and resources inside the group of tasks included in g.

3. Calculate an average Pj for each criteria j by using equation 2.

$$P_j = \frac{1}{NG} * \sum_{g=1}^{G} A_{jg} \quad \text{(Eq. 2)}$$

4. Calculate a weighted sum of Po, Ph, and Pr.

An optimisation technique is used to find the grouping with minimum evaluation value. This process is not to long, as only few tasks are considered at the same time. Finally, in order to stop the decomposition procedure, an evaluation of the complexity associated to each new role is performed.

5.2 Design Phase

For each iteration, once the roles have been identified during the analysis phase, the design phase starts, and cooperative binds among roles are identified. Once binds are identified, they are characterized as interaction situations were the coordination, collaboration and conflict resolution techniques are present. For each identified interaction the most suitable method must be applied. It is open for the developer to use already available methods and protocols or its own ones. Finally, the selected protocols are translated into communication binds.

5.3 Final Iteration

Once the analysis of complexity of the roles shows that there is no need for any new division of roles, then the final iteration of the AOPOA methodology is applied. In this last iteration, the set of resulting leave roles in the last level of the branches in the organizational tree is parameterized, thus obtaining the agent prototypes. A prototype of an agent has a specification of goals, sensorial inputs, effector outputs on the environment, and a definition of the actions that the agent can perform. Once the agents are specified, the events that trigger the actions performed by the agent must be analysed, also the mental abilities to execute these actions are specified. Events can be perceptions from the external elements in the environment or messages received associated to the interaction communication protocols. For each incoming event, the actions that must be taken must be determined.

In Figure 2 the complete process is depicted, the stages of the AOPOA methodology are shown. Circles in the diagram represent artefacts generated during this process.

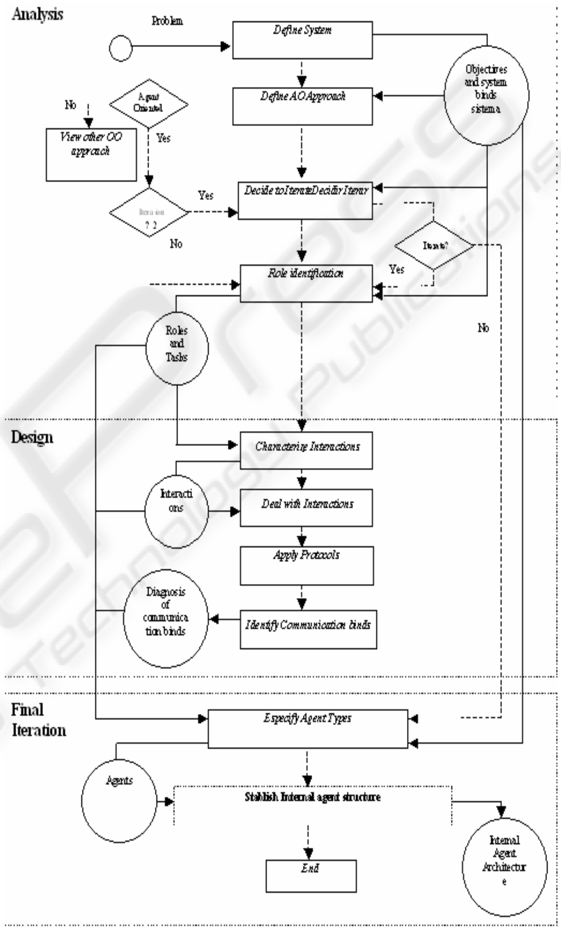

Figure 2: Phases of the AOPOA methodology.

6 DISCUSSION

The identified advantages of using AOPOA are:

- The organizational approach helps to deal with the system complexity and also to divide the problem in smaller ones with well-defined relationships.

- The grouping procedure aims to find a good combination of task groups, in order to optimize the role generation process.

· The process is complete, since considers all the required aspects to develop a good MAS; starting with requirements elicitation until defining the communication protocols and agent instances.

The selected case of study to test the AOPOA methodology, was a based on the construction of a restaurant simulation. The case of study also provided a way to test the methodology's organizational, intra-agent and intra-organizational scalability. Organizational scalability implies that a system can be designed to be part of a greater system, i.e. a restaurant can be part of a food chain. Intra-agent scalability, means that new objectives can be added to an already existent agent role. Finally, intra-organizational scalability allows to aggregate new roles to different organization's levels, over an already existent system. A detailed explanation of the case of study is out of the scope of this paper.

In order to implement the case of study, the BESA agent framework was used (González E. 2003). The AOPOA model transformation into a BESA implementation was direct and fast. The AOPOA model allows a rapid and robust event and action implementation of the MAS. A detailed presentation of the restaurant simulator design using AOPOA and implementation using BESA can be found in the work of Ahogado and Reinemer (Ahogado D. 2003).

Actual work to extend the AOPOA methodology include:

The use of dynamic roles, as a mechanism for agents to perform different roles accordingly to its own objectives and situation.

Agents mobility, applied for dynamic agents who can migrate through different machines in a distributed system.

Taking into account the obtained results, it can be concluded that AOPOA is a good choice for constructing complex agent based systems. In fact, the obtained advantages are derived from the cooperative rational agent concepts used, allowing a higher semantic level of the system and its conforming entities.

REFERENCES

Ferber J., 1999. "Multiagent Systems: An Introduction to Distributed Artificial Intelligence", Addison – Wesley Longman. 1ra. Ed., 1999.

Rusell N., 2003. "Artificial Itelligence: A Modern Approach", 2nd Edition, Pearson Education.

Ahogado D., Reinemer A.M., 2003. "Programación Orientada a Agentes: Metodologías de Desarrollo de Software", Proyecto de Grado Carrera de Ingeniería de Sistemas, Universidad Javeriana Bogotá.

González E., Bustacara C., Avila J., 2003. "BESA: Behavior- oriented, Event-driven Social-based Agent framework.". PDPTA'03, Las Vegas-USA, CSREA Press, vol. 3, Junio 2003, pp 1033-1039.

Alonso, F., et.al., 2004. Sonia: A methodology for natural agent development. ESAW'2004 - 5th Intl.Workshop on Engineering in the AgentsWorld

Wooldridge M., Jennings N., Kinny d., The GAIA Methodology for Agent-Oriented Analysis and Design, Autonomous Agents and Multi-Agent Systems, vol. 3, pp. 285-312, Kluwer Academic Publishers, 2000.

DeLoach, S. A. and Wood, M. F. 2000. Multiagent systems engineering: the analysis phase. Technical Report AFIT/EN-TR-00-02, Air Force Institute of Technology.

Odell, J., Nodine, M. H., and Levy, R. 2004. A metamodel for agents, roles, and groups. In AOSE, pages 78–92.

Howden, N., Rönnquist, R., Hodgson, A. and Lucas, A. 2001. JACK Intelligent Agents – Summary of an Agent Infrastructure. The agent oriented software group (www.agt-software.com.au).

Padgham, L. and Winikoff, M. 2002. Prometheus: A methodology for developing intelligent agents. In Proceedings of the Third International Workshop on AgentOriented Software Engineering, Bologna - Italy. AMMAS.

Penserini, L.; Kolp, M.; Spalazzi, L.; Panti, M. 2004. Socially-based design meets agent capabilities. Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'04).

Schreiber, G., Akkermans, H., Anjewierden, A., de Hoog, R., Shadbolt, N., Velde, W. V. D., and Wielinga, B. 1999. Knowlede Engineering and Management: The CommonKADS Methodology. MIT Press.