Definition of a Verification Method for the MASINA Methodology

Jose Aguilar [1], Juan Vizcarrondo [1], Niriaska Perozo [2]

[1] CEMISID, Dpto. de Computación, Facultad de Ingeniería, Av. Tulio Febres. Universidad de los Andes, Mérida 5101, Venezuela.
aguilar@ula.ve
[2] Unidad de Investigación en Inteligencia Artificial, Dpto. de Sistemas, Decanato de Ciencias y Tecnología. Universidad Centroccidental "Lisandro Alvarado", Barquisimeto 3001-Venezuela.
nperozo@ucla.edu.ve

Abstract

This article proposes a verification method for the MASINA Methodology, which allows the specification of Multi-Agent Systems (MAS). The verification instrument described in this article, divides the MAS verification design into two levels: The *Macro Level* verifies the design of each subsystem that makes up the MAS (breakdown into levels of abstraction), considering the structure and objectives of the proposed MAS; the *Micro Level* validates the design from the point of view of models (cross model), considering the features of each one of the MAS agents at a behavioral, communicational level.
Keywords: Multiagent Systems, Verification Methods, Specification Methodologies.

1 Introduction

Agent technology is being highly used in the last few years, and as a consequence, the industry is becoming interested in adopting this technology in order to develop their products. Nevertheless, in spite of the existence of various methodologies for the development of Multi-Agent Systems (MAS), there is no easy to apply existing formals instruments in order to verify the design of a MAS before it is implemented. Currently, one of the proposals consists of a MAS breakdown into different levels of abstraction, making MAS verification easier by evaluating the system from its minimum expression. But this approach is based on formal logic, which makes it harder to understand and thus, harder to implement [6]. Furthermore, it exclusively aims at verifying whether the system achieves its objectives, and does not take into account criteria such as properties and features that the agents must exhibit.

In this paper, a verification method is proposed, which on one hand will break the MAS down and will validate it without using formal logic at the macro level, verifying its functioning from the point of view of its objectives and its structure; and on the other hand, will validate it at the micro level by verifying the existing relationships between the models designed according to the MASINA methodology, through the "cross model" approach [6, 7]. MASINA is a methodology for MAS specification proposed in [2], which is an extension of the MAS-CommonKADS [4] methodology.

This methodology is based on a set of models that gather the basic and necessary elements to describe a MAS, such as the tasks that are carried out in it, the agents that make it up, the interactions in it, among others. The models are based on attribute-identification and relationships; the resulting product consists of a series of templates that describe the MAS. The verification is an important part in designing a MAS, because it guarantees that the system demands on conduct, properties and features are met. Our MAS design verification proposal with MASINA methodology leads to a formal analysis of the predefined relationships between the designed models.

2 MASINA Methodology

It's an extension of the MAS-CommonKADS [4] methodology, with the purpose of adapting the methodology to the modeling of industrial automation processes, although it can be used in other contexts [2]. It is basically made up of seven models [2]: organization, agent, communication, design, tasks, coordination and intelligence, which describe a MAS. MAS-CommonKADS has been modified to add learning machine features, use of smart techniques to carry out tasks, among other things. In particular, these are the extensions performed:

Agent Model: some attributes were added to this model, such as: *agent components* (if such agent is a MAS) and *reference frame* which is useful for describing the agent, that is, if such agent is based on an existing MAS reference model, such as SCDIA [8]. It specifies, just like MAS-CommonKADS, an agent's features: its reasoning capacities, skills and services, among other things.
Tasks Model: new features were added to this model, one to specify the properties of tasks that require the use of intelligent techniques, and another one to describe the procedure to be followed to execute such tasks. It additionally details the objectives of each task, its breakdown, the ingredients and problem-solving methods to achieve each goal.
Intelligence Model: The intelligence model is made up of a set of elements corresponding to human reasoning. It specifies the knowledge and reasoning methods needed for the smart agents to achieve their objectives. A plan is proposed to integrate concepts such as *experience* (knowledge acquired by the agent along its functioning), *dominion knowledge* (involving the variables used by each agent), *strategic knowledge* (problem-related information, derived from activities performed by other agents), *learning mec*hanism (it describes the experience accumulation process and the process to adapt the reasoning mechanisms and to update strategic knowledge), *task knowledge* (it gives the agent information about the tasks and about who can carry them out), *and reasoning mechanism* (it uses the acquired knowledge to make decisions, for example, to be an inference engine).The modeled intelligence component is perceived as a model that can be inserted into and extracted from each agent's structure, thus adding intelligent capacities to the agent, without compromising the agents' architecture.
Coordination Model: The coordination model lets the user establish the set of strategies that the communities of agents will use to achieve the group objectives. These

structures generally imitate coordination attitudes in human groups, such as contracting and auctioning processes, among others, where interaction and exchange of information generate individual actions that when added up will help reach specific goals. In other words, this model describes the mechanisms that help carry out a conversation between a group of agents, for which it is required to define: negotiation and conflict solving protocols, task allocation and planning methods, among others. This model describes the following attributes related to the conversation: *Name* (name of the conversation), *Type of Conversation* (predefined, emerging, among others), *Objective* (objectives satisfied by the conversation), *Agents* (agents participating in the conversation), *Initiator* (agent that initiates the conversation), *Service* (provides services), *Acts of Speech* (acts of speech involved in the conversation), *Description* (description of the conversation in a natural language), *Pre-condition* (conditions that must be present both in the issuer and in the receptor for the conversation to take place successfully) and *Termination Condition* (conditions both in the issuer and in the receptor after the conversation). The attributes of each component of the coordination model help specify the type of planning, the type of communication, and finally, the conflict solving strategies. In this model we specify the *coordination plan* (it indicates the type of coordination used: adaptive or predefined, objective to achieve, procedure used in the adaptive coordination, and default planning used in the predefined coordination), *planning* (type and technique of planning used), *direct communication mechanism* (messaging protocols used in the communication), *indirect communication mechanisms* (stimuli and response-threshold updating strategy in the communications) *metalanguage* (it defines the metalanguage used for communication between agents) and *ontology* (vocabulary used).

Communication Model: It describes each of the direct communications through information exchange, using mechanisms of the messaging type, and each of the indirect communications through the deposit of information in objects (use of shared-memory strategies and stimuli-response methods), that reflect the interactions between agents. In order to represent that, in [2] a communication model is proposed, based on the acts of speech present in the conversation, that is, in the issuer's intention when sending the content of the message. The attributes present in the communication model template, help specify the following in each act of speech involved in a conversation: *Objective* (the objective of the act of speech), *Type* (assertive, directive, compromising, expressive or declarative), *Participating Agents* (agents that participate in the act of speech), *Communication* (an act of speech can be performed directly or indirectly), *Issuer* (message issuer), *Receptor* (message receptor), *Conversation* (name of the conversation where the act of speech takes place), *Service* (name of the associated service), *Exchanged Data* (description of the data exchanged during the act of speech), *Description* (description of the act of speech in a natural language), *Pre-condition* (conditions that must be present both in the issuer and in the receptor for the act of speech to take place successfully) and *Termination Condition* (conditions both in the issuer and in the receptor after the act of speech), *Performative* (verb that describes the intention of the act of speech) and *Means of Communication* (way of communicating).

3 Description of the Proposed Verification Method

Before describing the verification model let's take a look at the evaluation criteria used.

3.1 Evaluation Criteria

a. Coherence Criterion. We define *coherence* as a system's property to behave as a unit, according to an evaluation dimension. We can evaluate this criterion according to different dimensions:

1. **Coverage Criterion.** The activities that lead to achieving the MAS objectives must be performed by at least one agent.
2. **Connectivity Criterion.** The agents may have to interact in such a way that the MAS objectives are reached through cooperatively developed activities, integrated in a global solution.
3. **Capacity Criterion.** It refers to the qualities required to perform the necessary activities in the MAS.

b. Clarity Criterion. Possibility of describing and representing the MAS's conduct, in such a way that an external observer is able to understand it.

c. Robustness Criterion. Degree of degradation of the MAS in the presence of failure or uncertainty. The failure may be due to a task that was not carried out, because of conflict in accessing a resource, etc. We can evaluate this criterion according to different dimensions:

1. **Blockage Recuperation Criterion.** In the case an agent is waiting for a message from another agent, we must make sure that this other agent is not waiting for an activity originated by the first agent, directly or indirectly.
2. **Failed Tasks Recuperation Criterion.** It is desirable that an agent be capable of recuperating if the result of a task does not reach the wanted objective, and other mechanisms to achieve the same objectives are known (try another conversation, another protocol, etc.).
3. **Team or Coalition Formation Criterion.** In case different agents share the same objectives, it is desirable to form a coalition to distribute the workload.
4. **Conflict Recuperation Criterion.** The agents must be capable of recuperating when a conflict of interest arises between them.
5. **Task Redundancy Criterion.** We must avoid having an indispensable agent in the system because it is the only one that knows how to perform a task.

d. Independence Criterion. It happens when an agent has the capacity of having its own behavior and reacting to external stimuli based on its internal state.

e. Effective Communication Criterion. It takes place when certain features inherent to communication between agents are complied with, for which aspects like synchronization and ontology use must be considered, among other things.

3.2 Design Verification Model

To verify the attained design, it is necessary to carry out the corresponding checks at two levels: Macro and Micro.

Macro Level

At this level the design of each subsystem from the point of view of the MAS is verified. For this purpose, the MAS is seen from the different levels of abstraction that make it up, since it is a MAS [6]. From the highest level, a MAS can be seen as a single component S with interfaces, noted as L_0, where the information and internal processes in it are not specified (the communications and internal processes are hidden). Upon descending to the next level of abstraction L_1, component S can be seen as a collection of agents, which exchange information with each other and with their surroundings, besides exercising control over their tasks. Likewise, an agent can be broken down into different agents at a further abstraction level L_i.

The verification tests to connect an abstraction level L_i with the next level L_{i+1} are carried out the following way: each time a component S is broken down, the new components S_{i+1} must be consistent with the previous definition of S_i. The components of level L_{i+1} are responsible of the objectives of component L_i. The components of level L_{i+1} must be consistent with the coordination model and keep the same external interactions as component L_i.

Table 1. Relations to be verified between the defined levels of abstraction

Nº.	RELATION	CRITERION
1	Satisfies-objective relation, (Opt.), one to many, between Component of Level L_{i+1} and Component of Level L_i. A component of Level L_{i+1} satisfies one or many objectives of level L_i.	Coverage (Coherence)
2	Made-up-of relation, (Opt.), one to many, between Component of Level L_i and Component of Level L_{i+1}. A component of level L_i is made up of one or many components of level L_{i+1}.	Coverage (Coherence)
3	Related-to relation, (Opt.), one to many, between Component of Level L_{i+1} and Component of Level L_i. A component of L_{i+1} interrelates to one or many components of level L_i and level L_{i+1}.	Connectivity
4	Performs-task relation, (Opt.), many to many, between Component of Level L_{i+1} and Component of Level L_i. The components of level L_{i+1} are responsible of the tasks in level L_i.	Coverage (Coherence)
5	Initiated-by relation, (Opt.), one to many, between task of Component of level L_{i+1} and Component of level L_i or L_{i+1}. A component of level L_{i+1} performs tasks initiated at the request of one or many components of level L_i or L_{i+1}.	Independence
6	Involves relation, (Opt.), one to many, between task of Component of level L_{i+1} and Component of level L_i or L_{i+1}. A component of level L_{i+1} performs a task together with one or many components of level L_i and L_{i+1}.	Team or Coalition Formation (Robustness)

Micro Level

At this level the design is validated from the models point of view. In this article the design verification is explained through the comparison of each model with the rest of the models, known as “Cross-Model” [7]. We are going to represent only some of the tables of our “Cross-Model”, for the case of the *Agents Model* with the rest of the models (relationship among agent model and the rest of the models). We must make the same thing for the rest of the models (task model versus rest of models, and so forth).

Task Model: The agent model provides a frame to detail the agents’ capacities to carry out their tasks. The following relations must be verified:

Table 2. Relations between the Agent Model and the Task Model

Nº.	RELATION	CRITERION
1	Performed-by relation, (Opt.), many to many, between task and agent. The tasks are assigned to one or many agents.	Coverage (coherence)
2	Supplies-ingredient relation, (Opt.), one to many, between agent and ingredients. The task ingredients (Inputs) are provided by agents.	Connectivity (coherence)
3	Receives-ingredient relation, (Opt.), one to many, between agent and ingredients. The task ingredients (Outputs) are received by the agents.	Connectivity (coherence)
4	Requires-capacity relation, (Opt.), many to many, between task and agent capacity. The necessary capacities to carry out a task are specified in detail in the agent capacity entity.	Capacity (coherence)
5	Satisfies-objective relation, (Opt.), one to many, between task and agent. A task can satisfy the objective of one or many agents.	Coverage (coherence)
6	Recuperates relation (Opt.), one to one, between task and agent. An agent recuperates and tries to achieve through other means the unaccomplished objective of a task.	Failed Task Recu-peration (Robustness)
7	Requires-independence relation, (Opt.), one to many, between task and agent. A task may require for one or many agents to have their own behavior, in other words, to have the capacity to decide when a task should be carried out.	Independence

Communication Model: Once the communication model has been developed, it can re-feed the agents model in order to find out whether it has all the elements that will enable it to make the necessary communications.

Table 3. Relations between the Agents Model and the Communication Model

Nº.	RELATION	CRITERION
1	Involves relation, (Opt.), one to many, between act of speech and agent. The acts of speech are executed by one or many agents.	Coverage (coherence)
2	Initiated-by relation, (Opt.), one to many, between act of speech and agents. The acts of speech are initiated by one or many agents.	Connectivity (coherence)
3	Knows-ontology relation, (Opt.), many to many, between agent and communication. Each agent must know the ontology used in the	Effective Communica-tion

	communications where it participates.	

Coordination Model: The agent model must be developed before the coordination model, providing an initial set of agents with certain abilities and objectives, that will help define the conversations required in the system.

Table 4. Relations between the Agents Model and the Coordination Model

Nº.	RELATION	CRITERION
1	Satisfies-objective relation, (Opt.), one to many, between conversation and agent. A conversation must satisfy the objective of one or many agents.	Coverage (Coherence)
2	Involves relation, (Opt.), one to many, between conversation and agent. The conversations are executed by one or many agents.	Connectivity (Coherence)
3	Requires-capacity relation, (Opt.), one to many, between interaction and agent. The capacities shown by an agent in a conversation are specified in detail in the agent capacity entity.	Capacity (Coherence)
4	Requires-synchronization relation, (Opt.), one to many, between conversation and agent. A conversation needs synchronization between the agents involved in it.	Effective Communication
5	Provides-service relation, (Opt.), One to one, between agent and conversation. Through a conversation, an agent provides services to other agents.	Connectivity (Coherence)
6	Initiated-by relation, (Opt.), one to many, between conversation and agent. The conversations are initiated by one or many agents.	Connectivity (Coherence)

Intelligence Model: The intelligence model is used to describe the agents' intelligent capacities (learning, reasoning capacity, among others).

Table 5. Relations between the Agents Model and the Intelligence Model

Nº.	RELATION	CRITERION
1	Requires-reasoning-capacity relation, (Opt.), one to one, between agent and decision-making.	Capacity (coherence)

4 Study Case: Design Verification of a Community Manager System for an Operating System

As a study case we will consider the design of the Web Operating System (WOS) described in [1], since the WOS was designed in its entirety as a MAS. We can break it down according to the levels of abstraction mentioned in macro level. Additionally, at the micro level, we will validate the Community Manager Subsystem's design (CMS) [1, 3], which is one of the subsystems of WOS. At the highest level, the WOS can be seen as a single component S with interfaces, noted as L_0, where the information and internal processes in it are not specified. At the next level of abstraction L_1, component S can be seen as a collection of agents or sub-components S_i (Subsystems) that reach the objectives of the WOS, and sub-sequentially, each subcomponent S_i or subsystem can be seen as a collection of agents that achieve the objectives of that subsystem (see figure 1). The WOS subsystems are: **RMS** (Resource Manager System), **LRMS** (Local Repositories Manager System), **RRMS** (Remote Repositories Manager System), **WOMS** (Web Object Manager System) and **CMS** (Community Manager System). As an example of the next level, the CMS agents are: Community Administrator (CA) and Search Coordinator (SC).

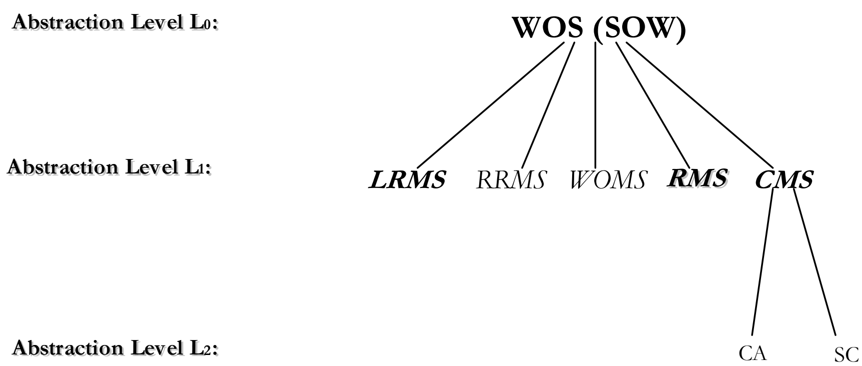

Fig. 1. Levels of Abstraction and WOS breakdown

4.1 Analysis of the Results at the Macro Level Verification

It was proven that at the macro level, that the design of the CMS proposed in [1, 3] coherently satisfies the objectives set in the WOS (see table 1). Likewise, it was also proven that the interactions of the CMS subsystem with the rest of the WOS subsystems are performed in a coordinated way. In regards to the CMS breakdown, we can note that it is made up of 2 agents: *the Search Coordinator Agent and the Community Administrator Agent*, which help satisfy the objectives set for the CMS (see table 1). Additionally, the CMS relations described in table 1 are kept by the CMS agents, and the tasks that have to be carried out by the CMS are distributed between the two agents.

4.2 Results Analysis of the Verification of the CMS design at the Micro Level

Let us take a look at the results observed for each cross model performed for the CMS:

Agents Model vs. Task Model: It was verified that all the CMS tasks are assigned to a CMS agent; all the ingredients required or provided by the tasks are supplied or received by agents defined in the WOS; all the CMS agents satisfy their objectives by performing the tasks defined in the tasks model; the capacities required by the tasks are provided by the agents that perform them (defined in the agents model), and lastly, it was indicated which tasks need independence and robustness on the part of the agents that carry them out (table 2).

Agents Model vs. Communication Model: the ontology used, the agents involved, and who initiated each act of speech of the designed communication model coincide with the defined agents and the known ontology for each CMS agent (table 3).

Agents Model vs. Coordination Model: it was validated that the involved agents, the objectives to be satisfied, the required capacities, the services provided and the synchronizations required by the CMS coordination model correspond to the agents, objectives, capacities and services defines in the CMS agent model (table 4).

Agents Model vs. Intelligence Model: it was verified that the reasoning capacities set in the intelligence model for the CMS agents, correspond to the reasoning capacities defined for the CMS agent model (table 5).

Task Model vs. Communication Model: it was proven that the CMS tasks are executed through service requests to the CMS agents, and for this purpose, the CMS agents and, the WOS agents in general, interact through the speech acts reflected in the CMS Communication model.

Task Model vs. Coordination Model: it was verified that the entry and exit ingredients required in the CMS task model, correspond to the ingredients exchanged by the agents in the CMS coordination model.

Task Model vs. Intelligence Model: the reasoning capacities required by the tasks performed by the CMS agents, are considered in the intelligence model of them.

Coordination Model vs. Communication Model: every speech act required in the CMS coordination model is defined in the CMS communication model.

Coordination Model vs. Intelligence Model: It was verified that the reasoning capacities required by the conversations in the CMS coordination model are included in the intelligence model of the CMS agents.

Communication Model vs. Intelligence Model: the reasoning capacities required by the speech acts are considered in the intelligence model of the CMS agents.

5 Conclusions

A methodology that has been proposed to specify MAS is the MASINA methodology [2]. MASINA is an extension of the MAS-CommonKADS methodology [4] to take into account MAS specification needs, such as task planning specification, intelligence notion, exchange of information with agent-oriented communication languages, code mobility or motivation of system components. But the efficient design of a MAS through a given methodology cannot be complete without some form of design verification. In this article a MAS design verification method is proposed according to MASINA, one that is highly effective and exhaustive when validating objectives, properties, structures, communication and interaction of each agent in the MAS. It also evaluates certain MAS criteria such as coherence, independence and robustness, among others. For this purpose the MAS architecture verification is performed from different levels of abstraction (macro level) and through MASINA cross model (micro level). Through the defined verification process we can detect whether there are any unused elements, and whether there are any contradictions or ambiguities in the design.

References

1. Aguilar, J., Perozo, N., Ferrer, E., Vizcarrondo, J. *Arquitectura de un Sistema Operativo Web Basado en Sistemas Multiagentes.* Revista Colombiana de Computación. Vol. 5 (2004) 7-29.
2. Aguilar, J., Rivas F., Hidrobo F., Cerrada M. *Análisis y Diseño de Sistemas Multiagente Usando la Metodología MASINA.* Reporte Técnico – Universidad de los Andes (CEMISID), January 2004.
3. Perozo, N; Aguilar, J. *Diseño de un Sistema Manejador de Comunidades para un Sistema Operativo Web*. Proc. V Jornadas de investigación del Decanato de Ciencias y Tecnología - UCLA, Venezuela (2004) 110-120.
4. Iglesias, C. *Definición de una Metodología para el desarrollo de Sistemas Multiagentes.* Tesis Doctoral, Departamento de Ingeniería de Sistemas Telemáticos, Universidad Politécnica de Madrid (1998).
5. Gómez, J. *Metodologías para el Desarrollo de Sistemas Multi-Agente.* Revista Iberoamericana de Inteligencia Artificial, Vol 18 (2003) 51-63.
6. Joeri, E., Catholijn, J., Jan T. *Compositional Verification of Multi-agent systems in Temporal Multi-Epistemic Logic.* Journal of Logic, Language and Information, Vol. 11 (2002) 195 – 225.
7. Weiss, G. *Multiagent Systems. Modern Approach to Distributed Artificial Intelligence*. MIT Press, Cambridge, 1999.
8. Aguilar, J., Rivas F., Hidrobo F., Cerrada M. *Design of an Industrial Automation Architecture based on Multi-agents Systems.* Proceeding of the 16th IFAC World Congress, Praga, July 2005.

Jose Aguilar Castro

He is graduated as *Systems Engineer* in 1987 in the University of the Andes (ULA) Mérida-Venezuela, obtained a *Master in Informatics* in 1991 at University Paul Sabatier Toulouse-France, and the *Doctorate in Computer sciences* in 1995 in the University Rene Descartes Paris-France. In addition, he made a *Postdoctorate* in the Department of Sciences of the Computer at University of Houston between 1999 and 2000. He is Professor in the Department of Computation at ULA. In this moment, he is the Director and is Member of the Research Staff at the Training Center in Microelectronics and Distributed Systems (CEMISID) at ULA. Besides, he is the President of the Foundation for the Development of Science and Technology of the State Mérida, Venezuela. Nowadays, he has already published more than 150 articles in magazines, books and acts of international congresses, in the areas of Parallel and Distributed Systems, Intelligent Computation and Systems of Control, and has supervised and coordinated several theses of engineers, masters and doctorates.

Niriaska Perozo Guédez

She is graduated as *Informatics Engineer* in 1997 in the University "Lisandro Alvarado" (UCLA) Barquisimeto-Venezuela, obtained a *Master in Computer sciences* in 2004 in the University of the Andes (ULA) Mérida-Venezuela, and she is making her doctoral studies in the Training Center in Microelectronics and Distributed Systems (CEMISID) at ULA. She is Professor in the Department of Systems at UCLA, and Member of the Research Staff at the Artificial Intelligence Research Unit of that University. Nowadays, she has published some articles in the areas of Distributed Systems, Operating systems and Artificial Intelligent.

Juan Vizcarrondo Rojas

He is graduated as *Systems engineer* in 2000 in the University of the Andes (ULA) Mérida-Venezuela, obtained a *Master in Computer sciences* in 2004 in the University of the Andes (ULA) Mérida-Venezuela. He is Professor in the Academy of Free Software in the Foundation for the Development of Science and Technology of the State Mérida, Venezuela. Besides, he is invited researcher in the Training Center in Microelectronics and Distributed Systems (CEMISID) at ULA. Nowadays, he has published some articles in the areas of Distributed Systems, Operating systems and Artificial Intelligent.