Analysis of Bullying in Cooperative Multi-agent Systems' Communications

Celia Gutiérrez

Universidad Complutense de Madrid, Spain

Abstract — **Cooperative Multi-agent Systems frameworks do not include modules to test communications yet. The proposed framework incorporates robust analysis tools using IDKAnalysis2.0 to evaluate bullying effect in communications. The present work is based on ICARO-T. This platform follows the Adaptive Multi-agent Systems paradigm. Experimentation with ICARO-T includes two deployments: the equitative and the authoritative. Results confirm the usefulness of the analysis tools when exporting to Cooperative Multi-agent Systems that use different configurations. Besides, ICARO-T is provided with new functionality by a set of tools for communication analysis.**

Keywords— **Analysis, Bullying, Communication, Cooperative Multi-agent Systems, ICARO-T.**

I. Introduction

Cooperative Multi-agent Systems are a type of Multi-agent System (MAS) which main goal concerns the design of complex systems for which the global behavior emerges from the behaviors and interactions of the agents that compose the system. Each agent has an individual goal and a behavior based on a cooperative attitude.

Tools to evaluate communications among agents are relevant to build an efficient architecture in cooperative MAS, in order to fulfill a complete agent-oriented software engineering (AOSE) paradigm. Although this type of tools has been used in non adaptive MAS, like those built under the Ingenias Development Kit (IDK) methodology [1], they have not been applied to a cooperative model.

Communications analysis tools are intended to discover non equal communications among agents, like the ones presents in [2], which identify patterns that appear in a bully scenario:

- bully agents, when they send too many messages.
- mistreated agents, when they received too many messages.
- mistreated-bully agents, when they send and receive too many messages.
- isolated agents, when they neither send nor receive messages.
- regular agents, when they behave correctly because they send and receive messages in a balanced way.

It is desirable to design communications where all agents follow regular patterns. If this situation does not happen, agents may be overloaded and/or overloading with messages, causing the overall system to be suffering of a bullying affect as well. In MAS where agents have different roles depending on their goals, non balanced communication can also appear in a group of agents with the same role. This is the case when there is a bad policy of selection of an agent among several agents of the same type.

Therefore, the bullying effect can appear at three levels:

1) Agent level, when there are agents with non desirable patterns.
2) Type of agent level, when there are types of agents with non desirable patterns.
3) System level, when the system level presents non desirable patterns.

To avoid bullying effect, the designer must handle validation tools which also identify the possible origin of the bullying (e.g., a conversation). With this information, it is possible a re-design of the MAS interactions, towards a MAS a system without bullying. The final consequence of a good design of communications is low response times, and higher Quality of Software (QoS) results, [2], [3].

The framework presented in [4], IDKAnalysis2.0, carries out an analysis of bullying behaviors in communications of MAS built with IDK tool. Its design is based on two modules, so that the second module carries out the validation. This separation males possible the integration of the second module in other MAS architectures. These ones must feed the second module with the logs extracted from the messages exchanged in the conversations among agents. The logs must contain the necessary information to perform the bullying classification.

The contribution of this paper is twofold:

1) To test IDKAnalysis2.0 with a cooperative MAS platform. This will consolidates IDKAnalysis2.0 as an exportable platform for different kinds of MAS, and is a step forward to a new version IDKAnalysis3.0.
2) To complete a cooperative MAS framework with a set of tools that validates the correctness of the design of MAS communications.

This work includes a study of the related work concerning a type of cooperative AMAS in Section 2. Section 3 contains a description of the framework that follows this paradigm,

ICARO-T. Section 4 includes a full description of the whole architecture composed of ICARO-T and part of IDKAnalysis2.0. Section 5 includes the results of the experimentation using the new platform. And finally, Section 6 contains the conclusions and future work.

II. RELATED WORK

The design of adaptive complex systems based on cooperative MAS and emergence has been studied since the 90s. In this vein, there is a relevant theory, called Adaptive Multi-Agent Systems (AMAS) [5], [6]. The main concept of this theory is to gives local criteria to design agents in order to make possible the emergence of an organization in the system and produce its global function in consequence. The characteristic of adaptation of the system makes possible this function to change and is produced by self-organization of the agents. The cooperative attitude is the main purpose of this self-organization since it guides, locally, the agents in its decision making. The designer task is to define: the agents, the environment (if needed) and the means for interaction; the organization is emerging. This approach contrasts with others where the designer has to specify the organization, like those based on the AGR (Agent, Groups, Roles) model [7], or INGENIAS [8].

The contributions to AMAS paradigm include:

- A methodology to design software for emergent behaviours, called ADELFE [9], which includes operational needs/requirements analysis, analysis and design phases, available on www.irit.fr/ADELFE. It is also a pioneer in considering as well the environment as helping to identify the agents.
- A formalization the AMAS theory, by the use extended automata products [10].
- Experimentations using different scenarios:, STAFF, ABROSE, FORSIC, ANTS, ARCADIA, and so on.

From the above-mentioned state-of the-art, at the time being, there is not a methodology to analyze and debug agent communications for AMAS.

III. ICARO-T FRAMEWORK

ICARO-T is a representative platform of this cooperative paradigm [11]. ICARO-T is a software infrastructure designed for the development of applications with agent organizations. ICARO-T provides "agent patterns" from which instances of "application agents" can be generated and executed on the nodes of a processor network. Applications are modeled as organizations composed of agents and resources. Table I shows the main differences between ICARO-T and other agents platforms.

ICARO-T paradigm offers the possibility to model MAS cooperation in two ways:

1) A team model following AMAS theory, where task responsibility is assigned on the most suitable cost evaluation to achieve the goal. This assignment is agreed between team nodes by exchanging messages containing these estimations.

2) A hierarchical model where a coordinator assigns the task team members estimations and then assigns the task to the most suitable subordinate based on the cost evaluations sent to him.

This distinction makes possible the analysis of the three levels of communication detailed in Section 1 using the hierarchical model, and only two levels (the agent and system level) using the team model.

TABLE I
COMPARISON WITH OTHER AGENT PLATFORMS

Feature	ICARO-T	Other Agent Platforms
Representation	Classes in Java	Logic clauses
Communication	Messages/events	Clauses / percepts/ FIPA messages
Paradigm based on	States/Goal processing	BDI Model
Codification	Java patterns	Logic model in Java
Distribution	Transparent	Possible, based on JADE
Support tools	Java tools	Ad-hoc

IV. SOLUTION ARCHITECTURE

The ICARO-T extended architecture contemplates two modules, following the IDKAnalysis2.0 [4], like Fig. 1 shows.

Fig. 1. Block diagram of the IDKAnalysis2.0, extracted from [5].

In our case, the first module is ICARO-T architecture, and the second module is the customization of the evaluation module of IDKAnalysis2.0. Each time a message is received by an agent, an event log is generated in the log event file. The generation of these logs can be extended to other type of events, with more fields, but in this case these logs just contain the basic information for the bullying analysis.

- Name of the agent that receives the message.
- Name of the agent that sends the message.

For both fields, the agent name includes the type of agent it belongs to, so that the bullying effect can also be analyzed at the type of agent level.

There are three parameters for the response time:

- Number of iterations that a task must be executed to calculate the response time.
- The initial task that must be executed to start the response time counting.
- The final task that must be executed to end the response time counting.

For both models (team and hierarchical) these parameters have not been customized because the experimentation case study already provides measurement of the efficiency performance, like the section 5 describes.

The customization of the second module has been done in two ways, each one for each of the two models.

A. Customization of the Evaluation Module for the Team Model

These are the parameters and values customized for this model:
- Role that is suspected to be the Bully in the conversations: *RobotMasterIA*
- Role of that is suspected to be the Mistreated in the conversations: *RobotMasterIA*
- Threshold for the bullying metrics: 1.0

In this case the role is the same for the Bully and the Mistreated, because there is no distinction among the agents, and only the measurements for the agent and system levels are necessary. The name of the agents begins with *RobotMasterIA*

B. Customization for the Evaluation Module for the Hierarchical Model

These are the parameters and values customized for this model:
- Role that is suspected to be the Bully in the conversations*: robotCoordinator*
- Role of that is suspected to be the Mistreated in the conversations: *robotSubordinated*
- Threshold for the bullying metrics: 1.0

In this case, the role to be analyzed as Bully only contains one instance, and corresponds to the coordinator agent; the role suspected to be Mistreated belongs to the subordinate agent type. In this case the three levels of the measurements are going to be taken into account, to detect if the coordinator agent has a bias towards a concrete subordinate agent.

V. EXPERIMENTATION RESULTS

The hardware of the experimentation has been a machine with 2 GHz and 2GB RAM, using 32-bit Windows 7 Professional.

The first stage of the experimentation has been the model used in [12]. The work is part of the research effort undergone in the ROSACE (Robots and Embedded Self-Adaptive Communicating Systems) project available at http://www.irit.fr/Rosace,737. In a simulated fire forest scenario, there are Autonomous Adaptive Vehicles (AAV), who coordinate among themselves for helping potential victims. Agents of the MAS act as the AAV, performing as RobotMaster for the team model, and Robot Coordinator or RobotSubordinate for the hierarchical model.

The communication in the hierarchical model is sent from the coordinator towards the subordinates with two purposes:
1) To request to estimate their cost for achieving the goal.
2) To accept/refuse proposals for assuming the goal.

The communication in the team model is sent among the components, to exchange cost estimations, and decide which member is the best situated to help the victim. Evaluation to assign to a concrete robot the rescue of a concrete victim, is made considering the time needed for helping the victim. In our case, it is useful to test that the communication of both models is not overloaded with too many messages to evaluate the cost of rescue a victim.

A. Experimental parameters

Experimentation has been running using the parameters described in the following subsections:

1) The type of model

As mentioned before, there are two models, the hierarchical model and the team model.

2) The team size and the number of victims

Notation $nRmV$ represents the configuration for n robots and m victims. In the present experimentation the running configurations have been:
- 4R6V-4 robots and 6 victims
- 4R16V-4 robots and 16 victims
- 5R16V-5 robots and 16 victims

Fig. 2 shows the distribution of victims and robots (i.e. the AAV's) of a scenario sample for 4 robots and 6 victims configuration.
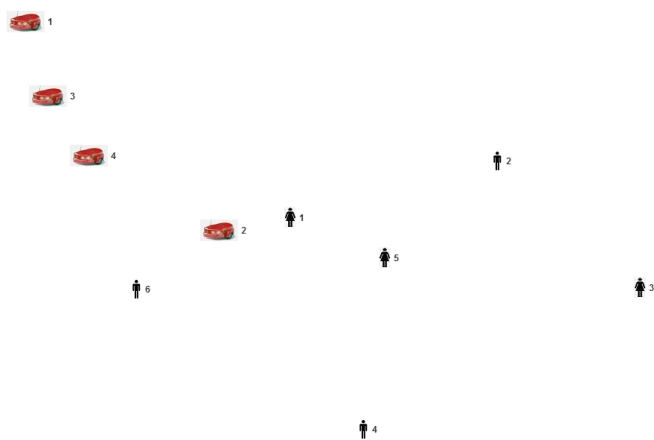

Fig. 2. AAV scenario for 4 robots and 6 victims.

3) The frequency of message

This applies to the time to request the rescue in order to assess the response of the agents when they y are undergoing stressing requests. We have chosen 2 representative values: 80 and 1000 milliseconds

The combination of these three aspects summarizes a total of 12 experimental configurations, as Table II shows.

TABLE II
EXPERIMENTAL CONFIGURATIONS

Type of model	Team and Victims Configuration	Interval time
Hierarchical	4R6V	80
Hierarchical	4R6V	1000
Hierarchical	4R16V	80
Hierarchical	4R16V	1000
Hierarchical	5R16V	80
Hierarchical	5R16V	1000
Team	4R6V	80

Team	4R6V	1000
Team	4R16V	80
Team	4R16V	1000
Team	5R16V	80
Team	5R16V	1000

Fig. 2 Scenario for the victims and robots

Experimental results for all configurations appear in the next subsections.

B. Bullying results for the hierarchical model

Bullying results for agents using the mentioned configurations are contained in Table III and IV. The first one shows the values for the metrics at the agent level. $BS(A_j)$ and $MS(A_j)$ values are used to classified the bullying effect of agents related to the agents of the system; in a similar way $BR(A_j)$ and $MR(A_j)$ are related to the agents playing the same role; and $CA(A_j)$ related to ratio between messages received and sent by the agent.

TABLE III
AGENT BULLYING MEASURES FOR THE HIERARCHICAL MODEL

Configuration	Interval time	Agent name	Metric				
			$BS(A_j)$	$MS(A_j)$	$BR(A_j)$	$MR(A_j)$	$CA(A_j)$
4R6V	80	RobotCoordinator	5.0	0.0	1.0	0.0	0.0
		RobotSubordinated_1	0.0	1.16	0.0	0.93	1.0
		RobotSubordinated_2	0.0	1.5	0.0	1.2	1.0
		RobotSubordinated_3	0.0	1.16	0.0	0.93	1.0
		RobotSubordinated_4	0.0	1.16	0.0	0.93	1.0
	1000	RobotCoordinator	5.0	0.0	1.0	0.0	0.0
		RobotSubordinated_1	0.0	1.16	0.0	0.93	1.0
		RobotSubordinated_2	0.0	1.5	0.0	1.2	1.0
		RobotSubordinated_3	0.0	1.16	0.0	0.93	1.0
		RobotSubordinated_4	0.0	1.16	0.0	0.93	1.0
4R16V	80	RobotCoordinator	5.0	0.0	1.0	0.0	0.0
		RobotSubordinated_1	0.0	1.18	0.0	0.95	1.0
		RobotSubordinated_2	0.0	1.25	0.0	1.0	1.0
		RobotSubordinated_3	0.0	1.31	0.0	1.05	1.0
		RobotSubordinated_4	0.0	1.25	0.0	1.0	1.0
	1000	RobotCoordinator	5.0	0.0	1.0	0.0	0.0
		RobotSubordinated_1	0.0	1.3	0.0	1.1	1.0
		RobotSubordinated_2	0.0	1.12	0.0	0.9	1.0
		RobotSubordinated_3	0.0	1.25	0.0	1.0	1.0
		RobotSubordinated_4	0.0	1.25	0.0	1.0	1.0
5R16V	80	RobotCoordinator	6.0	0.0	1.0	0.0	0.0
		RobotSubordinated_1	0.0	1.12	0.0	0.93	1.0
		RobotSubordinated_2	0.0	1.25	0.0	1.04	1.0
		RobotSubordinated_3	0.0	1.25	0.0	1.04	1.0
		RobotSubordinated_4	0.0	1.18	0.0	0.98	1.0
		RobotSubordinated_5	0.0	1.18	0.0	0.98	1.0
	1000	RobotCoordinator	6.0	0.0	1.0	0.0	0.0
		RobotSubordinated_1	0.0	1.18	0.0	0.98	1.0
		RobotSubordinated_2	0.0	1.25	0.0	1.04	1.0
		RobotSubordinated_3	0.0	1.18	0.0	0.98	1.0
		RobotSubordinated_4	0.0	1.18	0.0	0.98	1.0
		RobotSubordinated_5	0.0	1.18	0.0	0.98	1.0

Using these values as inputs for the classification rules, like in [2], with standard threshold of 1.00, Table IV shows the

class labels for the agents. These results are expected from the type of communication exchanged between RobotCoordinator and RobotSubordinated agents. The first ones must be Bully because they only send messages to the other agents. The second ones must be always be Mistreated because they only receive messages.

TABLE IV
AGENT BULLYING CLASS FOR THE HIERARCHICAL MODEL

Configuration	Interval time	Agent name	Class
4R6V	80	RobotCoordinator	Bully
		RobotSubordinated_1	Mistreated
		RobotSubordinated_2	Mistreated
		RobotSubordinated_3	Mistreated
		RobotSubordinated_4	Mistreated
	1000	RobotCoordinator	Bully
		…	…
5R16V	1000	RobotSubordinated_5	Mistreated

Table V reveals the values for the metrics and class at the RoborSubordinated role level, showing a regular behavior in all configurations. This means that the communication is similar in the agents of this type. RobotCoordinator has not been measured as there is only one agent, and there are not more agents to compare with it.

TABLE V
ROBOTSUBORDINATED ROLE BULLYING MEASURES AND CLASS FOR THE HIERACHICAL MODEL

Configuration	Interval time	Metric		Class
		$BR(R)$	$MR(R)$	
4R6V	80	0.0	1.01	Regular
4R6V	1000	0.0	1.00	Regular
	…	…		
5R16V	1000	0.0	1.00	Regular

Table VI shows the results of the metrics and class at the system level, showing the expected results. In this type of MAS, where there is an agent that sends all messages, and the rest of agents only receive, the expected behavior is of bullying effect, in this case Bully.

TABLE IV
SYSTEM BULLYING MEASURES AND CLASS FOR THE HIERARCHICAL MODEL

Configuration	Interval time	Metric			Class
		$BS(S)$	$MS(S)$	$CS(S)$	
4R6V	80	5.0	1.26	1.56	Bully
	1000	5.0	1.26	1.56	Bully
4R16V	80	5.0	1.25	1.56	Bully
	1000	5.0	1.25	1.56	Bully
5R16V	80	6.0	1.20	1.80	Bully
	1000	6.0	1.20	1.80	Bully

In conclusion, the communication of this MAS has been designed correctly, with a bullying behavior at an agent level, that has been influenced to the system level. Moreover, the communications for the RobotSubordinated agents have been

sent in an equal manner among the agents of this type, showing regular patterns in consequence.

C. Bullying results for the team model

Bullying measures for agents of this model have been collected in Table VII. In this case, as all agents belong to the same type, the measures related to the role have not been included, as they do not reveal anything important.

TABLE VII
AGENT BULLYING MEASURES FOR THE TEAM MODEL

Configuration	Interval time	Agent name	Metric		
			$BS(A_j)$	$MS(A_j)$	$CA(A_j)$
4R6V	80	RobotMasterIA_1	1.13	1.08	0.48
		RobotMasterIA_2	0.91	0.91	0.50
		RobotMasterIA_3	1.13	1.11	0.49
		RobotMasterIA_4	0.80	0.88	0.52
	1000	RobotMasterIA_1	0.93	0.96	0.50
		RobotMasterIA_2	1.01	0.98	0.49
		RobotMasterIA_3	1.13	1.15	0.50
		RobotMasterIA_4	0.91	0.89	0.49
4R16V	80	RobotMasterIA_1	1.03	1.61	0.60
		RobotMasterIA_2	1.15	0.00	0.00
		RobotMasterIA_3	0.83	1.18	0.58
		RobotMasterIA_4	0.96	1.19	0.55
	1000	RobotMasterIA_1	0.88	1.00	0.53
		RobotMasterIA_2	0.98	1.02	0.50
		RobotMasterIA_3	1.06	0.95	0.47
		RobotMasterIA_4	1.05	1.00	0.48
5R16V	80	RobotMasterIA_1	1.26	1.18	0.48
		RobotMasterIA_2	0.88	0.97	0.52
		RobotMasterIA_3	0.89	0.93	0.51
		RobotMasterIA_4	0.86	0.98	0.53
		RobotMasterIA_5	1.09	0.92	0.45
	1000	RobotMasterIA_1	1.01	1.11	0.52
		RobotMasterIA_2	0.92	0.94	0.50
		RobotMasterIA_3	1.02	0.91	0.47
		RobotMasterIA_4	1.11	1.06	0.48
		RobotMasterIA_5	0.91	0.96	0.51

Table VIII shows the classification results at the agent level (using the same rules and thresholds as in subsection 5.A) showing that all patterns are regular, except one, which corresponds to RobotMasterIA_2 in the 4R16V configuration for an interval of 80 milliseconds. As this value put more stress than 1000 millisecond, a desyncrhonization delay is produced, due to interruption during the decision of assigning a task to a certain agent. This can cause an imbalanced communication in certain agents, such us this case. Besides that, some results for the configurations using 80 milliseconds are slightly worse that the ones using 1000 millisecond, as the values for the latter show, which are usually closer to 1.00 than the former ones.

In a similar way that in subsection 5.A, results for the team model have been obtained showing regular patterns for the system. The configurations using 80 milliseconds show a slightly worse performance, as explained before. These results have been collected in Table IX.

TABLE VIII
AGENT CLASSIFICATION FOR THE TEAM MODEL

Configuration	Interval time	Agent name	Class label
4R6V	80	RobotMasterIA_1	Regular
		…	…
4R16V	80	RobotMasterIA_1	Regular
		RobotMasterIA_2	Bully
		RobotMasterIA_3	Regular
		…	…

TABLE IX
SYSTEM CLASSIFICATION FOR THE TEAM MODEL

Configuration	Interval time	Metric			Class label
		$BS(S)$	$MS(S)$	$CS(S)$	
4R6V	80	1.02	1.00	1.01	Regular
	1000	1.00	1.00	1.00	…
4R16V	80	1.01	1.36	1.07	
	1000	1.00	1.00	1.00	
5R16V	80	1.02	1.00	1.01	
	1000	1.00	1.00	1.00	Regular

These results are also expected, revealing non bullying communications and an adequate design, in consequence. The value of MS(S) for 4R16V, 80 milliseconds shows a slight deviation from the others, closer to the unit. This is consequence of the deviation in the RobotMasterIA_2 in the agent bullying measures (see Table VII).

VI. CONCLUSIONS AND FUTURE WORK

This work presents a complete framework for a cooperative MAS platform that includes robust tools for communication analysis. These tools may be applied to this type of MAS to discover bullying patterns in communication among agents. ICARO-T platform support this paradigm, concretely the AMAS. The resulted patterns reveal there is a robust design in communication architecture. In the experimentation phase, two different deployments built with ICARO-T have been tested. Although the evaluation at the type level cannot be applied for the team deployment, the two other levels have been tested. In this sense, this research shows that the framework is flexible enough to be integrated with MAS different to IDK, which has the three levels of functionality, and validation. New findings in the metric values show better results for some configuration over the others. The designer can choose the use of the best configurations in executions in terms of equal communications. Therefore, the new framework becomes useful, relevant and consistent to validate communications.

This research opens other perspectives for ICARO-T in particular, and AMAS in general:

- For ICARO-T, future case studies communication can be validated with these tools. Besides, it is planned to validate these case studies with another set of tools, mentioned in section 1, [3].
- For AMAS in general, the possibility to use this framework using analysis tools is open. Besides, there are other analysis tools than can be applied for other purposes, as in

[13] and [14]. As these tools use logs with different kind of information, an ontotology may be parsed to get this information. This can be used by a new version of IDKAnalysis2.0, named IDKAnalysis3.0.

REFERENCES

[1] J. J. Gomez-Sanz, R. Fuentes-Fernández, J. Pavón, and I. García-Magariño, "INGENIAS Development Kit: a visual multi-agent system development environment," in *Proc. 7th Int. Conf. Autonomous Agents and Multiagent Systems,* 2008, pp. 1675-1676.

[2] C. Gutiérrez and I. García-Magariño, "Revealing bullying patterns in multi-agent systems," *Journal of Systems and Software*, vol. 84, no. 9, pp. 1563-1575, 2011.

[3] C. Gutiérrez, I. García-Magariño, and R. Fuentes-Fernández, "Detection of undesirable communication patterns in multi-agent systems," *Engineering Applications of Artificial Intelligence,* vol. 24, no. 1, pp. 103-116, 2011.

[4] C. Gutiérrez, "An Analysis Architecture for Communications in Multi-agent Systems", *International Journal of Artificial Intelligence and Interactive Multimedia*, vol. 2, no 1, pp. 65-72, 2013.

[5] V. Camps, M-P. Gleizes, P. Glize, "Une theorie des phénomens globaux fondée sur des interactions locals", *in Actes des Sixieme journées francophones IAD&SMA*, 1998.

[6] J-P. Georgé, M-P. Gleizes, P. Glize, "Conception de systems adaptatifs a fonctionnalité émergente: la théorie Amas", *Revue d'Intelligence Artificielle*, RSTI série RIA, vol. 17, no. 5, pp. 591-626, 2004

[7] J. Ferber, O. Gutknecht, "Aalaadin: a meta-model for the analysis and design of organizations in multi-agent systems", *in Proc. 3rd Int. Conf. On Multi-Agent Systems*, 1998, pp.128-135.

[8] J. Pavón and J.J. Gomez-Sanz, "Agent Oriented Software Engineering with INGENIAS," in *Proc. 3rd Int. Central and Eastern European Conf. on Multi-Agent Systems*, vol. 2691, Prague, pp. 394-403, 2003.

[9] G. Picard and M-P. Gleizes, "The ADELFE Methodology," *Designing Adaptive Cooperative Multi-Agent Systems*, chapter 8, pp. 157-176, Kluwer Publishing, 2004.

[10] D. Capera, J. Fanchon, J-P. Georgé, V. Camps, "A Generic Model Based on Automata for Multi-Agent Systems", in *Proc. 3rd European Workshop on Multi-Agent Systems*, 2005, pp. 79-90.

[11] ICARO-T. http://icaro.morfeo-project.org/lng/en/

[12] J-M. Gascueña, F. Garijo, A. Fernández-Caballero, M-P. Gleizes, P. Glize, "Implementation and Assessment of Robot Team Cooperation Models Using Deliberative Control Components", *Advances in Artificial Intelligence IBERAMIA 2012,* Lecture Notes in Computer Science series, vol. 7637, 2012, pp 412-421.

[13] C. Gutierrez, I. García-Magariño, E. Serrano, J-A. Botía, "Robust design of multi-agent system interactions: a testing approach based on pattern matching", *Engineering Applications of Artificial Intelligence* (submitted in 2013).

[14] C. Gutiérrez, I. García-Magariño, "Detection of Overworked Agents in INGENIAS", *Advances in Practical Applications of Agents and Multiagent Systems 2010*, Advances in Soft Computing, Advances in Soft Computing series, vol. 70, 2010, pp. 113-118.

C. Gutierrez was born in Bilbao, Spain, in 1969. She received the B.Eng. and Ph.D. degrees in Computer Science from the University of Deusto in 1992, and from the University of the Basque Country in 2000, respectively. She has worked for Labein technological research center, in Bilbao. She has also worked for the Basque Government and Indra, as a software Engineer. After doing teaching and research at some private universities, her current position is Assistant Professor at the Complutense University in Madrid, Spain, with the Grasia research group. She has published in relevant journals and has also direct financially supported projects. Her current interests focus on Data Mining, Multiagent Systems and Accessibility.