Practical Applications of Agents and MAS: Methods, Techniques and Tools for Open MAS

Javier Bajo, Juan M. Corchado, Vicente Botti and Sascha Ossowski

Abstract—**This paper presents a brief summary of the post-proceedings of the special session on practical applications held in the framework of IWANN 2009. The special session was supported by the THOMAS (TIN2006-14630-C03-03) project and aims at presenting the results obtained in the project, as well as at exchanging experience with other researchers in this field.**

Index Terms— **Multiagent systems, Agent's technology.**

I. Introduction

Research on Agents and Multi-Agent Systems has matured during the last decade and many effective applications of this technology are now deployed. An international forum to present and discuss the latest scientific developments and their effective applications, to assess the impact of the approach, and to facilitate technology transfer, has become a necessity.

The Special Session on Practical Applications of Agents and Multiagent Systems (http://iwann.usal.es/mas), in the framework of the 10th International Work-Conference on Artificial Neural Networks (IWANN 2009) provided a unique opportunity to bring multi-disciplinary experts and practitioners together to exchange their experience in all aspects of Agents and Multi-Agent Systems, especially those concerned with applications, methods, techniques and tools for open multi-agent systems.

The session intended to bring together researchers and developers from industry and academic world to report on the latest scientific and technical advances on the application of multi-agent systems, discuss and debate the major issues, and showcase the latest systems using agent based technology. It is a multidisciplinary discipline that may attract scientist and professionals to IWANN and to provide a different field in which to apply ANN based technology. It promotes a forum for discussion on how agent-based techniques, methods, and tools help system designers to accomplish the mapping between available agent technology and application needs. Other stakeholders should be rewarded with a better understanding of the potential and challenges of the agent-oriented approach.

This special session was supported by the THOMAS research project (TIN2006-14630-C03-03), which aim is to advance and contribute methods, techniques and tools for open multiagent systems, principally in the aspects related to organisational structures. THOMAS is a coordinated project in which the University of Salamanca, the Technical University of Valencia and the University of Rey Juan Carlos cooperate to find new solutions in the field of the multiagent systems. This special session provides a framework to disseminate the results obtained in the project and to exchange knowledge with other researchers in the field of the agent technology.

This volume includes a selection of the best papers presented in the special session, focusing on the physical aspect of the agents and multiagent systems.

Javier Bajo is with Pontifical University of Salamanca.
E-mail: jbajope@upsa.es
Juan M. Corchado is with University of Salamanca.
E-mail: corchado@usal.es
Vicente Botti is with Technical University of Valencia.
E-mail: vbotti@dsic.upv.es
Sasscha Ossowski is with University of Rey Juan Carlos.
E-mail: sascha.ossowski@urj.es

II. THOMAS: Methods, Techniques and Tool for Open Multiagent Systems

The recent years technological evolution in the areas of Computer Technology and Communications (Internet, WWW, e-commerce, wireless connectivity, etc.) has given rise to a new computation paradigm: '*computation as interaction'*. In this new paradigm, computation is something that occurs by means of and through communication among computational entities. Given this approximation, computation becomes an inherently social activity, instead of solitary, leading to new forms of conceiving, designing, developing and handling computational systems. An example of the influence of this point of view is the emerging model of the software as a service, as in the service-oriented architectures. In this model, the applications are no longer monolithic single-user applications, or distributed applications managed by only one organization, but rather societies of computational entities (components) that can be conceived as service providers. These components may not have been designed in a joint way or even by the same development team; they may enter or abandon different societies in different moments and for different reasons; and they may form coalitions or virtual organizations between themselves to attain their current goals.

The technology of agents / multiagent systems (MAS) has some characteristics that show its potential to support this new paradigm of computation as interaction, especially that of Open MAS. To satisfy the requirements of this new paradigm, we should provide the technology with the methods, agents, techniques, tools and infrastructures that support these new computational needs in a strong and efficient way.

Dynamic organizations of agents that self-adjust to obtain advantages from their present context are becoming increasingly important. These organizations might appear in dynamic or emerging societies of agents, such as the ones suggested by the Grid domains, Ambient Intelligence domains, or any other domain in which agents coordinate dynamically to offer compound services. Social factors in the organizations of multi-agent systems are also more and more important to structure the interactions in dynamic and open worlds. Any infrastructure that supports the execution of multiagent applications in these contexts should be strong and efficient. New approaches are necessary to support the infrastructures evolution, and to facilitate their growth and updating in execution time due to the characteristics of these open environments.

Essentially, the THOMAS project comes up with two major lines of research:

Dynamism/Regulation: flexibility to permit the entrance and exit of agents, evolution of the organisational structure, coercive mechanisms, etc.

Heterogeneity: different types of agents with diverse capabilities, different run-time coordination mechanisms (requires semantic descriptions of capabilities or services), different devices (physical resources of the devices), different communication channels (wireline or wireless networks)

In this context, the THOMAS project defines development methods, standards, and platforms for the interoperability of agents that consider these requirements.

III. Special Issue on Practical Applications of Agents and Multiagent Systems Details

This volume presents a selection of the best papers selected from those that were accepted for the 2009 edition. These articles capture the most innovative results and this year's trends: Multi-Agent Systems (MAS) Applications: commerce, health care, industry, internet, etc.; Agent and MAS architectures; Agent development tools; MAS middleware; Agent languages; Engineering issues of MAS; Web services and agents; Agents and grid computing; Real-time multi-agent systems; Agent-based social simulation; Security in MAS; Trust and reputation in MAS; Improving user interfaces and usability with agents; Information recovery with MAS; Knowledge management with MAS; Software Agents in Ubiquitous Computing; Agent technologies for Ambient Intelligence; Software Agents in Industry; Planning and scheduling in MAS; Agent Technologies for Production Systems; Service-Oriented Computing and Agents; Agents for E-learning and education; Mobile computation and mobile Communications. Each paper has been reviewed by three different reviewers, from an international committee composed of 15 members from 7 different countries. From the 17 papers accepted and presented in the special session, 6 were selected for this special issue:

In the first paper, Trigo et al. propose a multi-agent based simulation (MABS) framework to construct an artificial electric power market populated with learning agents The artificial market, named TEMMAS is materialized in an experimental setup involving distinct power generator companies which operate in the market and search for the trading strategies that best exploit their generating units' resources.

Gutiérrez and García-Magariño, in the second paper present a metrics suite for evaluating the communication of the multi-agent systems. These metrics assist designers in detecting undesirable communication patterns in multi-agent systems, in order to improve their cooperative behavior..

In the third paper, Conte et al. present a paradigm for modelling and analysing Home Automation Systems, based on Multi-Agent System theory. A rich and versatile environment for Home Automation System simulation is constructed for investigating the performances of the system.

Rodríguez et al. in the fourth paper illustrate a case study in which the THOMAS architecture is applied in order to obtain a multiagent system (MAS) that can provide recommendations and guidance in a shopping mall. The paper demonstrates that THOMAS is made up of a group of related modules that are well-suited for developing systems in other highly volatile environments similar to a shopping mall.

In the fifth paper, Urra et al. study existing mobile agent platforms by analyzing if they are suitable or not in a mobile environment. They identify some key missing features in the platforms and highlight the requirements and challenges that lie ahead. With this work, they expose existing problems and hope to motivate further research in the area.

In the last paper, Serrano et al. present the development of an infrastructure to study highly complex systems of Ambient Intelligence (AmI) which involve a large number of users. The key ideas about the development of a multi-agent based simulation (MABS) for such purposes, Ubik, are given. The paper also extrapolates effective technologies in the development of multi-agent systems (MAS) to the field of MABS. In particular, the basis for the use of forensic analysis as a method to assist the analysis, understanding and debugging of Ubik in particular and the general MABS are set up.

IV. Special Issue Acknowledgements

We would like to thank all the contributing authors, as well as the members of the Program Committee and the Organizing Committee for their hard and highly valuable work. Their work has helped to contribute to the success of this special session. We also would like to thank the IWANN 2009 for giving us the opportunity of organizing the special session, for their help and support. Thanks for your help, the special session on practical applications of agents and multiagent systems wouldn't exist without your contribution. Finally, The Guest Editors wish to thank Professors Miguel Cazorla and Vicente Matellan (Editors-in-Chief of Journal of Physical Agents) for the publication of this special issue, which notably contributes to improve the quality of the special session and the THOMAS project. We hope the reader will share our joy and find this special issue very useful.

Acknowledgment

This work has been supported by the MEC TIN2006-14630-C03-03 project.

Towards an Electricity Market Simulator

Paulo Trigo, Paulo Marques, Helder Coelho

Abstract—**This paper describes a multi-agent based simulation (MABS) framework to construct an artificial electric power market populated with learning agents. The artificial market, named TEMMAS (The Electricity Market Multi-Agent Simulator), explores the integration of two design constructs: i) the specification of the environmental physical market properties, and ii) the modeling of the decision-making (deliberative) and reactive agents. TEMMAS is materialized in an experimental setup involving distinct power generator companies which operate in the market and search for the trading strategies that best exploit their generating units' resources. The experimental results show a coherent market behavior that emerges from the overall simulated environment.**

Index Terms—**Electricity market modeling, multi-agent modeling and design, multi-agent based simulation, sequential decision process, adaptation and learning.**

I. INTRODUCTION

The start-up of nation-wide electric markets, along with its recent expansion to intercountry markets, aims at providing competitive electricity service to consumers. The new market-based power industry calls for human decision-making in order to settle the energy assets' trading strategies. The interactions and influences among the market participants are usually described by game theoretic approaches which are based on the determination of equilibrium points to which compare the actual market performance [1], [2]. However, those approaches find it difficult to incorporate the ability of market participants to repeatedly probe markets and adapt their strategies. Usually, the problem of finding the equilibria strategies is relaxed (simplified) both in terms of: i) the human agents' bidding policies, and ii) the technical and economical operation of the power system. As an alternative to the equilibrium approaches, the multi-agent based simulation (MABS) comes forth as being particulary well fitted to analyze dynamic and adaptive systems with complex interactions among constituents [3], [4].

In this paper we describe TEMMAS (The Electricity Market Multi-Agent Simulator), a MABS approach to the electricity market, aimed at simulating the interactions of agents and to study the macro-scale effects of those interactions. TEMMAS agents exhibit bounded rationality, i.e., they make decisions based on local information (partial knowledge) of the system and of other agents while learning and adapting their strategies during a simulation in order to reveal and assist to understand the complex and aggregate system behaviors that emerge from the interactions of the market agents.

Paulo Trigo is with LabMAg, GuIAA; DEETC, ISEL - Instituto Superior de Eng. de Lisboa, Portugal.
E-mail: ptrigo@deetc.isel.ipl.pt

Paulo Marques is with LabMAg, GuIAA; DEETC, ISEL - Instituto Superior de Eng. de Lisboa, Portugal.
E-mail: 28562@alunos.isel.ipl.p

Helder Coelho is with LabMAg; DI, FCUL - Faculdade de Ciências da Universidade de Lisboa, Portugal.
E-mail: hcoelho@di.fc.ul.pt

II. TEMMAS MODELING FRAMEWORK

We describe the structural TEMMAS constituents by means of two concepts: i) the *environmental entity*, which owns a distinct existence in the real environment, e.g. a resource such as an electricity producer, or a decision-making agent such as a market bidder generator company, and ii) the *environmental property*, which is a measurable aspect of the real environment, e.g. the price of a bid or the demand for electricity. Hence, we define the *environmental entity* set, $\mathcal{E}_\mathcal{T} = \{ e_1, \ldots, e_n \}$, and the *environmental property* set, $\mathcal{E}_\mathcal{Y} = \{ p_1, \ldots, p_m \}$. The whole environment is the union of its entities and properties: $\mathcal{E} = \mathcal{E}_\mathcal{T} \cup \mathcal{E}_\mathcal{Y}$.

The environmental entities, $\mathcal{E}_\mathcal{T}$, are often clustered in different classes, or types, thus partitioning $\mathcal{E}_\mathcal{T}$ into a set, $\mathcal{P}_{\mathcal{E}_\mathcal{T}}$, of disjoints subsets, $\mathcal{P}^i_{\mathcal{E}_\mathcal{T}}$, each containing entities that belong to the same class. Formally, $\mathcal{P}_{\mathcal{E}_\mathcal{T}} = \left\{ \mathcal{P}^1_{\mathcal{E}_\mathcal{T}}, \ldots, \mathcal{P}^k_{\mathcal{E}_\mathcal{T}} \right\}$ defines a full partition of $\mathcal{E}_\mathcal{T}$, such that $\mathcal{P}^i_{\mathcal{E}_\mathcal{T}} \subseteq \mathcal{E}_\mathcal{T}$ and $\mathcal{P}_{\mathcal{E}_\mathcal{T}} = \cup_{i=1\ldots k} \mathcal{P}^i_{\mathcal{E}_\mathcal{T}}$ and $\mathcal{P}^i_{\mathcal{E}_\mathcal{T}} \cap \mathcal{P}^j_{\mathcal{E}_\mathcal{T}} = \emptyset \; \forall i \neq j$. The partitioning may be used to distinguish between decision-making agents and available resources, e.g. a company that decides the biding strategy to pursue or a plant that provides the demanded power.

The environmental properties, $\mathcal{E}_\mathcal{Y}$, can also be clustered, in a similar way as for the environmental entities, thus grouping properties that are related. The partitioning may be used to express distinct categories, e.g. economical, electrical, ecological or social aspects. Another, more technical usage, is to separate constant parameters from dynamic state variables.

The factored state space representation. The state of the simulated environment is implicitly defined by the state of all its environmental entities and properties. We follow a factored representation, that describes the state space as a set, $\mathcal{V}$, of discrete state variables [5]. Each state variable, $v_i \in \mathcal{V}$, takes on values in its domain $\mathcal{D}(v_i)$ and the global (i.e., over $\mathcal{E}$) state space, $\mathcal{S} \subseteq \times_{v_i \in \mathcal{V}} \mathcal{D}(v_i)$, is a subset of the Cartesian product of the state variable domains. A state $s \in \mathcal{S}$ is an assignment of values to the set of state variables $\mathcal{V}$. We define $f_\mathcal{C}$, $\mathcal{C} \subseteq \mathcal{V}$, as a projection such that if s is an assignment to $\mathcal{V}$, $f_\mathcal{C}(s)$ is the assignment of s to $\mathcal{C}$; we define a *context* c as an assignment to the subset $\mathcal{C} \subseteq \mathcal{V}$; the initial state variables of each entity and property are defined, respectively, by the functions $init_{\mathcal{E}_\mathcal{T}} : \mathcal{E}_\mathcal{T} \rightarrow \mathcal{C}$ and $init_{\mathcal{E}_\mathcal{Y}} : \mathcal{E}_\mathcal{Y} \rightarrow \mathcal{C}$.

The decision-making approach. Each agent perceives (the market) and acts (sells or buys) and there are two main approaches to develop the reasoning and decision-making capabilities: i) the qualitative mental-state based reasoning, such as the belief-desire-intention (BDI) architecture [6], which is founded on logic theories, and ii) the quantitative, decision-theoretic, evaluation of causal effects, such as the Markov decision process (MDP) support for sequential decision-making in stochastic environments. There are also hybrid approaches that combine the qualitative and quantitative formulations [7], [8].

The qualitative mental-state approaches capture the relation between high level components (e.g. beliefs, desires, intentions) and tend to follow heuristic (or rule-based) decision--making strategies, thus being better fitted to tackle large-scale problems and worst fitted to deal with stochastic environments.

The quantitative decision-theoretic approaches deal with low level components (e.g., primitive actions and immediate rewards) and searches for long-term policies that maximize some utility function, thus being worst fitted to tackle large-scale problems and better fitted to deal with stochastic environments.

The electric power market is a stochastic environment and we currently formulate medium-scale problems that can fit a decision-theoretic agent model. Therefore, TEMMAS adaptive agents (e.g., market bidders) follow a MDP based approach and resort to experience (sampled sequences of states, actions and rewards from simulated interaction) to search for optimal, or near-optimal, policies using reinforcement learning methods such as Q-learning [9] or SARSA [10].

III. TEMMAS DESIGN

Within the current design model of TEMMAS the electricity asset is traded through a spot market (no bilateral agreements), which is operated via a *Pool* institutional power entity. Each generator company, *GenCo*, submits (to *Pool*) how much energy, each of its generating unit, $GenUnit_{GenCo}$, is willing to produce and at what price. Thus, we have: i) the power supply system comprises a set, $\mathcal{E}_{GenCo}$, of generator companies, ii) each generator company, *GenCo*, contains its own set, $\mathcal{E}_{GenUnit_{GenCo}}$, of generating units, iii) each generating unit, $GenUnit_{GenCo}$, of a *GenCo*, has constant marginal costs, and iv) the market operator, *Pool*, trades all the *GenCo*s' submitted energy.

The bidding procedure conforms to the so-called "block bids" approach [11], where a block represents a quantity of energy being bided for a certain price; also, *GenCo*s are not allowed to bid higher than a predefined price ceiling. Thus, the market supply essential measurable aspects are the energy price, quantity and production cost. The consumer side of the market is mainly described by the quantity of demanded energy; we assume that there is no price elasticity of demand (i.e., no demand-side market bidding).

Therefore, we have: $\mathcal{E}_{\mathcal{T}} = \{ Pool \} \cup \mathcal{E}_{GenCo} \cup_{g \in \mathcal{E}_{GenCo}} \mathcal{E}_{GenUnit_g}$ where $\mathcal{E}_{\mathcal{Y}} = \{ quantity, price, productionCost \}$. The $quantity$ refers both to the supply and demand sides of the market. The $price$ referes both to the supply bided values and to the market settled (by *Pool*) value.

The $\mathcal{E}_{GenCo}$ contains the decision-making agents. The *Pool* is a reactive agent that always applies the same predefined auction rules in order to determine the market price and hence the block bids that clear the market. Each $\mathcal{E}_{GenUnit_{GenCo}}$ represents the *GenCo*'s set of available resources.

The resources' specification. Each generating unit, $GenUnit_{GenCo}$, defines its marginal costs and constructs the block bids according to the strategy indicated by its generator company, *GenCo*. Each $GenUnit_{GenCo}$ calculates its marginal costs according to, either the "WithHeatRate" [12]) or the "WithCO_2" [13] formulation.

The "WithHeatRate" formulation estimates the marginal cost, *MC*, by combining the variable operations and maintenance costs, *vO&M*, the number of heat rate intervals, $nPat$, each interval's capacity, cap_i and the corresponding heat rate value, hr_i, and the price of the fuel, $fPrice$, being used; the marginal cost for a given $i \in [1, nPat]$ interval is given by,

$$MC_{i+1} = vO\&M + \frac{(cap_{i+1} \times hr_{i+1}) - (cap_i \times hr_i)}{blockCap_{i+1}} \times fPrice \tag{1}$$

where each block's capacity is given by: $blockCap_{i+1} = cap_{i+1} - cap_i$.

The "WithCO_2" marginal cost, *MC*, combines the variable operations and maintenance costs, *vO&M*, the price of the fuel, $fPrice$, the CO_2 cost, CO_2cost, and the unit's productivity, η, through the expression,

$$MC = \frac{fPrice}{\eta} \times K + CO_2cost + vO\&M \tag{2}$$

where K is a fuel-dependent constant factor, and CO_2cost is given by,

$$CO_2cost = CO_2price \times \frac{CO_2emit}{\eta} \times K \tag{3}$$

where CO_2emit is the CO_2 fuel's emissions. Here all blocks have the same capacity; given a unit's maximum capacity, $maxCap$, and a number of blocks, $nBlocks$, to sell, each block's capacity is given by: $blockCap = \frac{maxCap}{nBlocks}$.

The decision-making strategies. Each generator company defines the bidding strategy for each of its generating units. We designed two types of strategies: a) the *basic-adjustment*, that chooses among a set of basic rigid options, and b) the *heuristic-adjustment*, that selects and follows a predefined well-known heuristic. There are several *basic-adjustment* strategies already defined in TEMMAS. Here we outline seven of those strategies, $sttg_i$ where $i \in \{ 1, \ldots, 7 \}$, available for a *GenCo* to apply: i) $sttg_1$, bid according to the marginal production cost of each $GenUnit_{GenCo}$ (follow heat rate curves, e.g., cf. tables II and III), ii) $sttg_2$, make a "small" increment in the prices of all the previous-day's block bids, iii) $sttg_3$, similar to $sttg_2$, but makes a "large" increment, iv) $sttg_4$, make a "small" decrement in the prices of all the previous-day's block bids, v) $sttg_5$, similar to $sttg_4$, but makes a "large" decrement, vi) $sttg_6$, hold the prices of all previous-day's block bids, vii) $sttg_7$ set the price to zero. There are two *heuristic-adjustment* defined strategies: a) the "Fixed Increment Price Probing" (FIPP) that uses a percentage to increment the price of last day's transacted energy blocks and to decrement the non-transacted blocks, and b) "Physical Withholding based on System Reserve" (PWSR) that reduces the block's capacity, as to decrement the next day's estimated system reserve (difference between total capacity and total demand), and then bids the remaining energy at the maximum market price.

The agents' decision process. The above strategies correspond to the *GenCo* agent's primary actions. The *GenCo* has a set, $\mathcal{E}_{GenUnit_{GenCo}}$, of generating units and, at each decision-epoch, it decides the strategy to apply to each generating unit, thus choosing a vector of strategies, $\overrightarrow{sttg}$, where the i^{th} vector's component refers to the $GenUnit^{i}_{GenCo}$ generating unit; thus, its action space is given by: $\mathcal{A} = \times_{i=1}^{\left|\mathcal{E}_{GenUnit_{GenCo}}\right|} \{\, sttg_1, \ldots, sttg_7 \,\}_i \cup \{\, \text{FIPP}, \text{PWSR} \,\}$. The *GenCo*'s perceived market share, *mShare*, is used to characterize the agent internal memory so its state space is given by $mShare \in [\, 0..100 \,]$. Each *GenCo* is a MDP decision-making agent such that the decision process *period* represents a daily market. At each decision-epoch each agent computes its daily profit (that is regarded as an internal reward function) and the *Pool* agent receives all the *GenCo*s's block bids for the 24 daily hours and settles the hourly market price by matching offers in a classic supply and demand equilibrium price (we assume a hourly constant demand).

TEMMAS *architecture and construction.* The TEMMAS agents along with the major inter-agent communication paths are represented in the bottom region of Figure 1; the top region represents the user interface that enables to specify the each of the resources' and agents' configurable parameters. The implementation of the TEMMAS architecture followed the INGENIAS [14] methodology and used its supporting development platform. Figure 2 presents the general "agent's perspective", where the tasks and the goals are clustered into individual and social perspectives. Figure 3 gives additional detail on the construction of tasks and goals using INGENIAS.

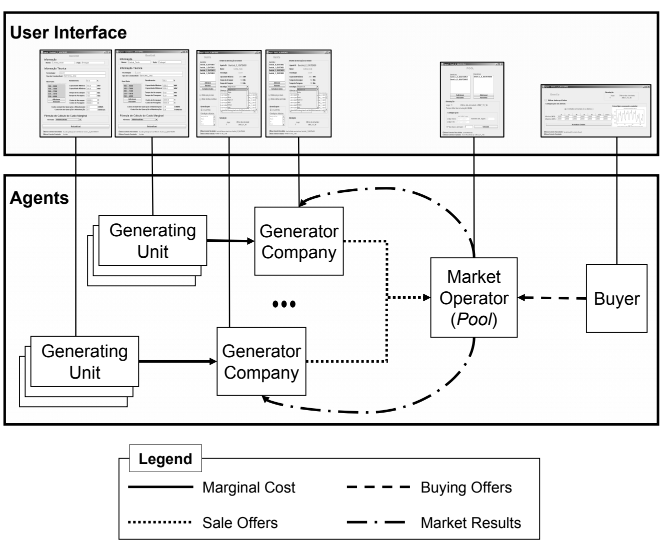


Fig. 1. The TEMMAS architecture and the configurable parameters.

IV. TEMMAS ILLUSTRATIVE SETUP

We used TEMMAS to build an electric market simulation model. Our experiments have two main purposes: i) illustrate the TEMMAS functionality, and ii) analyze the agents'

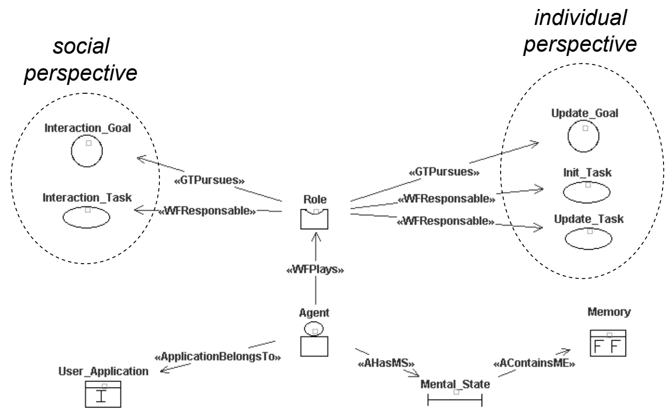


Fig. 2. TEMMAS agent's view using INGENIAS framework.

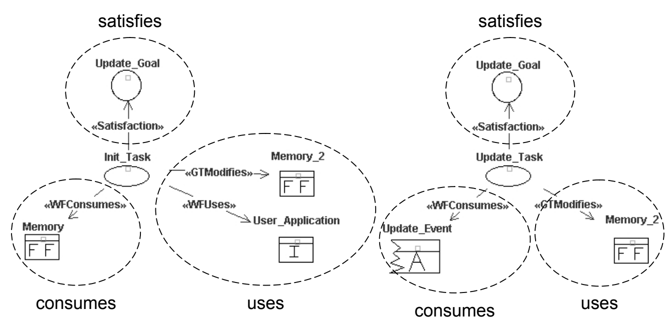


Fig. 3. TEMMAS tasks and goals specification using INGENIAS framework.

resulting behavior, e.g. the learnt bidding policies, in light of the market specific dynamics; cf. [15] for our extended experimental setup.

We considered three types of generating units: i) one base load coal plant, CO, ii) one combined cycle plant, CC, to cover intermediate load, and iii) one gas turbine, GT, peaking unit. Table I shows the essential properties of each plant type and tables II and III shows the heat rate curves used to define the bidding blocks. The marginal cost was computed using expression (1); the bidding block's quantity is the capacity increment, e.g. for CO, the 11.9 marginal cost bidding block's quantity is $350 - 250 = 100$ MW (cf. Table II, CO, top lines 2 and 1).

TABLE I
PROPERTIES OF GENERATING UNITS; THE UNITS' TYPES ARE COAL (CO), COMBINED CYCLE (CC) AND GAS TURBINE (GT); THE O&M INDICATES "OPERATION AND MAINTENANCE" COST.

Property	*unit*	Type of generating unit		
		CO	**CC**	**GT**
Fuel	—	Coal (BIT)	Nat. Gas	Nat. Gas
Capacity	MW	500	250	125
Fuel price	€/MMBtu	1.5	5	5
Variable O&M	€/MWh	1.75	2.8	8

We designed a simple experimental scenario and Table IV shows the *GenCo*'s name and its production capacity according to the respective *GenUnit*s (cf. Table I). The "*active*" suffix (cf. Table IV, *name* column) means that *GenCo* searches for its *GenUnit*s best bidding strategies; i.e. "*active*" is a policy

TABLE II
CO AND CC UNIT'S CAPACITY BLOCK (MW) AND HEAT RATE (BTU/KWH) AND THE CORRESPONDING MARGINAL COST (€/MWH).

CO generating unit			**CC generating unit**		
Cap.	Heat rate	Marg. cost	Cap.	Heat rate	Marg. cost
250	12000	—	100	9000	—
350	10500	11.9	150	7800	29.8
400	10080	12.5	200	7200	29.8
450	9770	12.7	225	7010	30.3
500	9550	13.1	250	6880	31.4

TABLE III
GT UNIT'S CAPACITY BLOCK (MW) AND HEAT RATE (BTU/KWH) AND THE CORRESPONDING MARGINAL COST (€/MWH) .

GT generating unit		
Cap.	Heat rate	Marg. cost
50	14000	—
100	10600	44.0
110	10330	46.2
120	10150	48.9
125	10100	52.5

learning agent.

We considered a constant, 2000 MW, hourly demand for electricity. Figure 4 shows the market share evolution while *GenCo_minor&active* learns to play in the market with *GenCo_major*, which is a larger company with a fixed strategy: "bid each block 5€ higher than its marginal cost". We see that *GenCo_minor&active* gets around 18% (75 – 57) of market from *GenCo_major*. To earn that market the *GenCo_minor&active* learnt to lower its prices in order to exploit the "5€ space" offered by *GenCo_major* fixed strategy.

TABLE IV
THE EXPERIMENT'S *GenCo*S AND *GenUnit*S.

GenCo		
name	Prod. Capac.	***GenUnit*s**
GenCo_major	2000	2×CO & 4×CC
GenCo_minor&active	875	3×CC & 1×GT

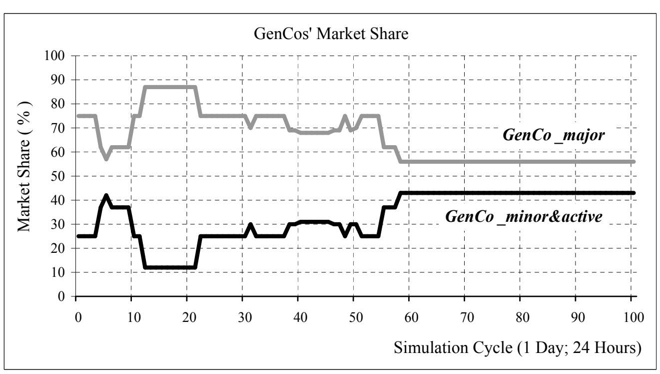


Fig. 4. Market share evolution induced by *GenCo_minor&active*.

V. CONCLUSIONS AND FUTURE WORK

This paper describes our preliminary work in the construction of the TEMMAS agent-based electricity market simulator. Our contribution includes a comprehensive formulation of the simulated electric power market environment along with the inhabiting decision-making and learning agents. Our initial results reveal an emerging and coherent market behavior, thus inciting us to extend TEMMAS with additional bidding strategies and to incorporate specific market rules, such as congestion management and pricing regulation mechanisms.

REFERENCES

[1] Berry, C., Hobbs, B., Meroney, W., O'Neill, R., Jr, W.S.: Understanding how market power can arise in network competition: a game theoretic approach. Utilities Policy **8**(3) (September 1999) 139–158

[2] Gabriel, S., Zhuang, J., Kiet, S.: A Nash-Cournot model for the north american natural gas market. In: Proceedings of the 6th IAEE European Conference: Modelling in Energy Economics and Policy. (2–3 September 2004)

[3] Schuster, S., Gilbert, N.: Simulating online business models. In: Proceedings of the 5th Workshop on Agent-Based Simulation (ABS-04). (May 3–5 2004) 55–61

[4] Helleboogh, A., Vizzari, G., Uhrmacher, A., Michel, F.: Modeling dynamic environments in multi-agent simulation. JAAMAS **14**(1) (2007) 87–116

[5] Boutilier, C., Dearden, R., Goldszmidt, M.: Exploiting structure in policy construction. In: Proceedings of the IJCAI-95. (1995) 1104–1111

[6] Rao, A., Georgeff, M.: BDI agents: From theory to practice. In: Proceedings of the First International Conference on Multiagent Systems, S (1995) 312–319

[7] Simari, G., Parsons, S.: On the relationship between MDPs and the BDI architecture. In: Proceedings of the AAMAS-06. (May 8–12 2006) 1041–1048

[8] Trigo, P., Coelho, H.: Decision making with hybrid models: the case of collective and individual motivations. International Journal of Reasoning Based Intelligent Systems (IJRIS); Inderscience Publishers (2009)

[9] Watkins, C., Dayan, P.: Q-learning. Mach. Learning **8** (1992) 279–292

[10] Sutton, R., Barto, A.: Reinforcement Learning: An Introduction. MIT P. (1998)

[11] : OMIP - The Iberian Electricity Market Operator. online: 'http://www.omip.pt'

[12] Botterud, A., Thimmapuram, P., Yamakado, M.: Simulating GenCo bidding strategies in electricity markets with an agent-based model. In: Proceedings of the 7th Annual IAEE European Energy Conference (IAEE-05). (August 28–30 2005)

[13] Sousa, J., Lagarto, J.: How market players aadjusted their strategic behaviour taking into account the CO2 emission costs - an application to the spanish electricity market. In: Proceedings of the 4th International Conference on the European Electricity Market (EEM-07), Cracow, Poland (May 23–27 2007)

[14] Gómez-Sanz, J., Fuentes-Fernández, R., Pavón, J., García-Magariño, I.: INGENIAS development kit: a visual multi-agent system development environment (BEST ACADEMIC DEMO OF AAMAS'08). In: Proceedings of the Seventh AAMAS, Estoril, Portugal (May 12-16 2008) 1675–1676

[15] Trigo, P., Marques, P.: The electricity market as a multi-agent system. In: Proceedings of the 5th International Conference on the European Electricity Market. IEEE Xplore, Lisbon, Portugal (May 28–30 2008)

A Metrics Suite for the Communication of Multi-agent Systems

C. Gutiérrez and I. García-Magariño

Abstract— **Multi-agent systems are composed of autonomous entities that cooperate forming emergent behaviors. The communication between agents is sometimes unpredictable, and the resulting communication patterns can become crucial in their performance. This paper presents a metrics suite for evaluating the communication of the multi-agent systems. These metrics assist designers in detecting undesirable communication patterns in multi-agent systems, in order to improve their cooperative behavior. At the same time, we provide a demonstration that a well balanced communication is related to high levels of Quality of Service measured by response times.**

***Index Terms*—Agent-oriented software engineering, balance, metrics, mobile agents and multi-agent systems.**

I. Introduction

In the execution of multi-agent systems (MASs), several resources (such as CPU time and network traffic) are consumed, and consequently they must be properly designed. One of the key aspects of designing MASs is the manner agents communicate. The aim of this work is to provide a metrics suite to assist developers in designing the communication of their MASs, and these metrics are based on the load balancing. For instance, a group of agents can always choose the same agent, and the consequences are that an agent is overloaded with the continuous requests while others are not receiving any request. This fact causes the overall time response to be higher, and the overall consequence is a Quality of Service reduction.

In order to prevent MASs from using these communication patterns, this work presents a suite of metrics for measuring communication in MASs and detecting the origin of the unbalanced communications. The metrics measures the balance of communication in an active manner (sending messages) and in a passive manner (receiving messages).

As a proof of concept, the metrics suite is applied to MASs that are created with the INGENIAS [1] methodology, which asserts the importance of communication in MAS. Its support tool, INGENIAS Development Kit (IDK) [2], allows building plug-ins to make the specifications work, and therefore, this fact facilitates the creation of a framework for measuring the proposed metrics. In the application of these metrics to several IDK developments, one can observe that the measurement values of these metrics are strongly related with the Quality of Service (QoS). This paper presents one of these cases, in which the metrics measures the communication of a MAS for crisis-management, and indicates which is the origin of the low QoS of this system.

Using the same case study, this work shows that the effect of the unbalanced communication becomes greater when the number of agents increases, obtaining worse values for the metrics and the response times (therefore also for the QoS).

Celia Gutierrez and Ivan García-Magariño are with the Universidad Complutense de Madrid.
E-mail: cegutier@fdi.ucm.es; ivan_gmg@fdi.ucm.es

II. Multi-agent Systems and Their Communication

A MAS [3] is a system composed of multiple interacting intelligent agents. Multi-agent systems can be used to solve problems which are difficult or impossible for an individual agent or monolithic system to solve.

Designers of MASs describe models of agent interactions in order to guide the communication of MASs. For instance, Jiao et al. [4] propose to define a group of organization models for describing MASs, in which interaction patterns play a central role. The Agent Unified Modeling Language (AUML) [5] is the result of the cooperation of Foundation of Intelligent Physical Agents (FIPA) and the Object Management Group (OMG), and this modeling language includes formalism for defining the interaction protocols of multi-agent systems. In addition, the INGENIAS methodology also considers modeling elements to explicitly define the interactions, such as: the interaction protocols, the interaction units, and the precedence order of these interaction units.

Following this line of research, the presented work measures the amount and type of communication activity in several scopes. According to our study, there can be five main patterns of agents regarding the communication activity: overloaders, overloaded, isolated, overloaded-overloader, and regular. The first four types should not exist in a balanced system and the last one is the ideal agent:

1. *Overloaders*: Agents that overload the communication by sending too many messages.
2. *Overloaded*: Agents that are overloaded by receiving too many messages.
3. *Isolated*: Agents that neither send nor receive any message.
4. *Overloaded-overloader*: Agents that are overloaded but also overload other ones.
5. *Regular*: Agents whose behavior is ideal because they send and receive an adequate amount of messages according to the communication activity of the system.

The existence of the first four types is related to a low quality of QoS of the system. For instance, when an agent is occupied in paying attention to a great amount of requests (an overloaded agent), its response time delays and, consequently, the overall response time of the system also delays. In this example, the response time is used as an indicator of the QoS. The next section presents the necessary metrics for detecting the aforementioned types of agents.

III. METRICS SUITE FOR MAS COMMUNICATION

In order to determine when an agent is overloader or overloaded, this paper provides the definition of some metrics in two different scopes: the global system and the group of agents playing the same role. Agents will be classified within each scope, e.g. an agent will be classified into any of the mentioned classes using the value of metrics. It is important to distinguish both scopes because an agent can have a pattern (e.g. overloader) regarding the rest of the agents of a scope (e.g. global scope), and have another pattern (e.g. regular) regarding the agents of the other scope (e.g role scope). For instance, the reason can be that its activity is neutralized by the other agents' activity within a particular scope.

A. *Scope of the Global System*

The scope of the global system is the performance of all agents in the system. Table I provides the metric name with each calculation (1), (2), and limits (3), (4). All the metrics presented in this paper use the following notation:

- A1..An are agents.
- S is the set of agents in the system.
- n(X) is the cardinality of X set.
- $r(A_x)$ is the number of the A_x agent's received messages.
- $s(A_x)$ is the number of the A_x agent's sent messages.
- R is the subset of agents playing the same role.

The following rules interpret the measurement with certain thresholds:

1. Agents who do not receive or send any messages will be classified as isolated.
2. Agents whose BS is much higher than the threshold will be classified as overloaders.
3. Agents whose MS is much higher than the threshold will be classified as overloaded.
4. Agents whose BS and MS are both much higher than the threshold will be classified as overloaded-overloader.
5. Agents whose BS and MS are close to the threshold will be classified as regular, because their communication is balanced.

B. *Scope of Agents Playing the Same Role*

The performance of each agent can also be measured in the scope of the agents playing the same role. Table II provides the metric names with each calculation (5), (6), and limits (7), (8):

The agents are classified with the following rules, considering certain thresholds:

1. Agents who do not receive or send any messages will be classified as isolated.
2. Agents whose BR is much higher than the threshold will be classified as overloaders.
3. Agents whose MR is much higher than the threshold will be classified as overloaded.
4. Agents whose BR and MR are both much higher than the threshold will be classified as overloaded-overloader.
5. Agents whose BR and MR are not much higher than the threshold will be classified as regular, because their communication is balanced.

IV. CASE STUDY: A CRISIS-MANAGEMENT MAS

A MAS with overloaded communication effect has been tested to identify communication patterns. The MAS implements a crisis-management case [6] in which, a

TABLE I.
METRICS FOR AN AGENT PERFORMANCE IN THE SYSTEM SCOPE

Metric name	Description	Limits
$BS(A_j)$	OverloaderSystem Metric measures the amount of an agent's sent messages comparing it to the amount of sent messages in the system, with the following definition: $$BS(A_j)=\frac{s(A_j)}{\left(\sum_{A_j \in S} s(A_j)\right)/n(S)} \qquad (1)$$	$BS(A_j) \in [0, n(S)]$ $Min(BS(A_j))=0$ in case A_j does not send any message. $Max(BS(A_j))=n(S)$ if all messages are sent by the same agent A_j : $$BS(A_j)=X/(X/n(S)) \qquad (3)$$
$MS(A_j)$	OverloadedSystem Metric measures the amount of an agent's received messages comparing it to the amount of received messages in the system, with the following definition: $$MS(A_j)=\frac{r(A_j)}{\left(\sum_{A_j \in S} r(A_j)\right)/n(S)} \qquad (2)$$	$MS(A_j) \in [0, n(S)]$ $Min(MS(A_j)) = 0$ in case A_j does not send any message. $Max(MS(A_j)) = n(S)$ if all messages are sent by the same agent A_j : $$MS(A_j)=X/(X/n(S)) \qquad (4)$$

TABLE II.
METRICS FOR AN AGENT PERFORMANCE IN THE ROLE SCOPE

Metric name	Description	Limits
$BR(A_j)$	OverloaderRole Metric measures the amount of an agent's sent messages comparing it to the amount of sent messages by the agents playing the same role, with the following definition: $$BR(A_j)=\frac{s(A_j)}{(\sum_{A_j \in R} s(A_j))/n(R)} \quad (5)$$	$BR(A_j) \in [0, n(R)]$ $Min(BR(A_j)) = 0$ in case A_j does not send any message. $Max(BR(A_j)) = n(R)$ if all messages are sent by the same agent A_j: $$BR(A_j)=X/(X/n(R)) \quad (7)$$
$MR(A_j)$	OverloadedRole Metric measures the amount of an agent's received messages comparing it to the amount of received messages by the agents playing the same role, with the following definition: $$MR(A_j)=\frac{r(A_j)}{(\sum_{A_j \in R} r(A_j))/n(R)} \quad (6)$$	$MR(A_j) \in [0,n(R)]$ $Min(MR(A_j)) = 0$ in case A_j does not send any message. $Max(MR(A_j)) = n(R)$ if all messages are sent by the same agent A_j: $$MR(A_j)=X/(X/n(R)) \quad (8)$$

poisonous material has been released into a city. Since the centralized medicals services are not enough, the citizens with medical capabilities can offer help to the affected people in nearby areas; at the same time citizens are warned of the affected locations, in order to avoid them. All the experiments presented in this work can be reproduced with both the *IDK* coordinator agents send messages to the same network agent due to the ingenuous selection in the communication process, which always selects the first agent of the yellow pages of the MAS. In the second version, this problem has been fixed by randomly selecting a network agent. In particular, this work only alters the mechanism of selecting the network agent for

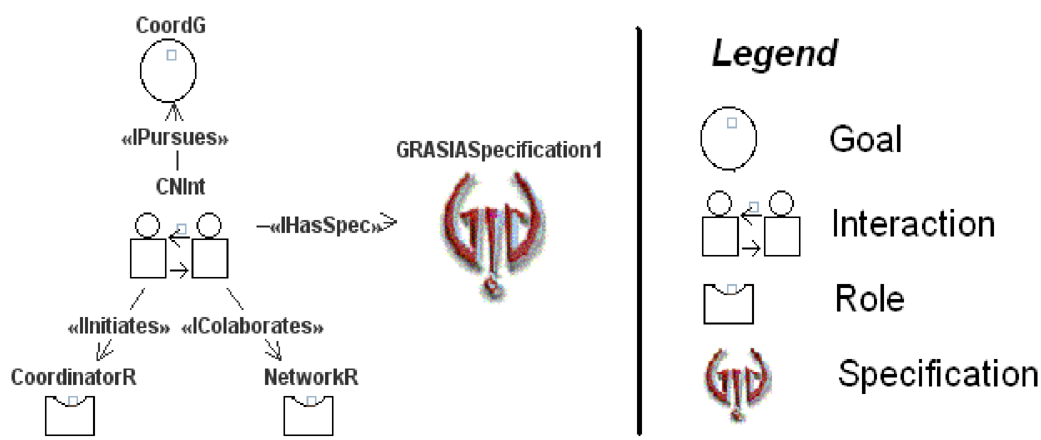


Fig. 1. The Coordinator-network Interaction, with the INGENIAS notation. There are two roles represented by the CoordinatorR and the NetworkR roles. The interaction is initiated by a coordinator agent, and is sequenced by the network agent.

2.8, which contains the crisis-management MAS and is available from *Grasia* Web (http://grasia.fdi.ucm.es in "*Software*" section and "*Additional Material for papers*" subsection).

In the experiments, there are coordinator agents (beginning with *CoordA* prefix) and network agents (beginning with *NetworkA* prefix), which exchange messages to coordinate help and notify the risk areas.

The experimentation considers two versions of the system: a crisis simulation with overloaded communication and another with balanced communication. In the first version, the crisis management system may be blocked because the triggering the *Coordinator-network* interaction (*CNInt* in Fig. 1).

Furthermore, the experimentation uses two deployments that respectively instantiate four and ten agents of each type. Fig. 2 and Fig. 3 respectively show examples of executions of these deployments.

From the experience of this case study, in both versions, a threshold of unit order (3.00) provides an accurate way to distinguish the regular and irregular patterns, in the global and role scopes, because the values are of unit order.

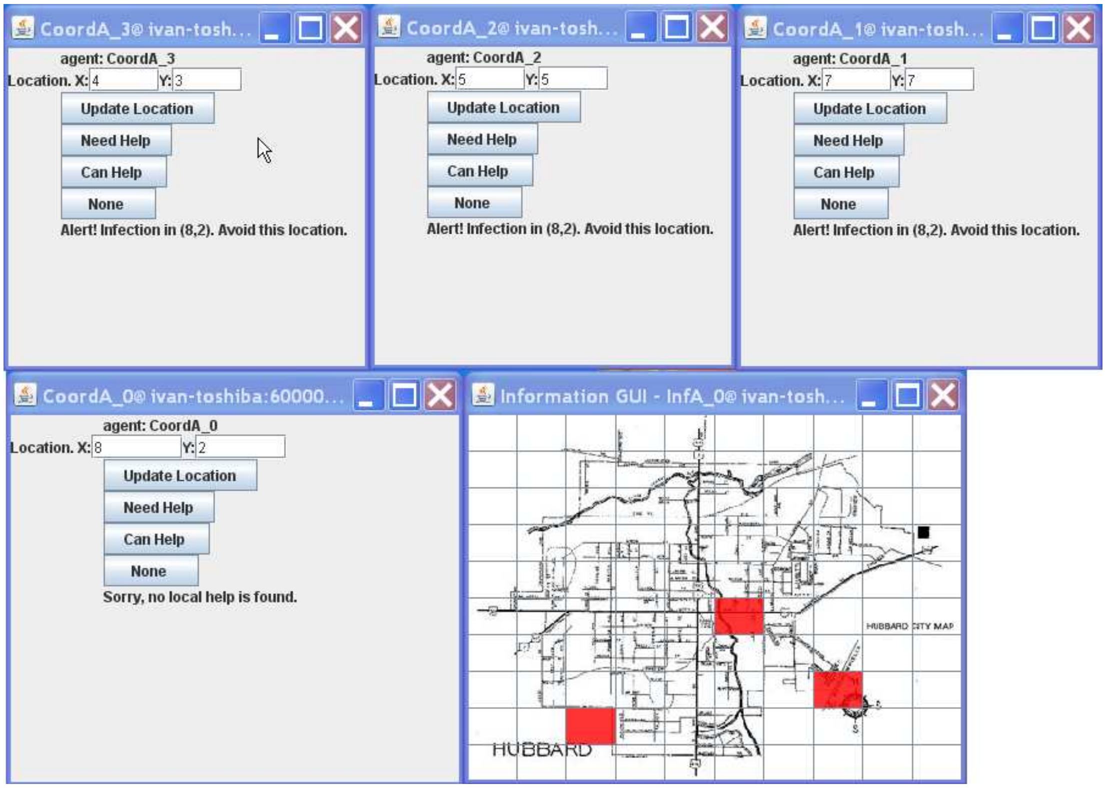


Fig. 2. Execution of the Crisis-management MAS with four agents. In this execution three agents are infected, whose positions are indicated in the map as infected areas. Nobody could help the last infected user (CoordA_0), since the unique user with medical capabilities (CoordA_3) is occupied helping another infected user (CoordA_2).

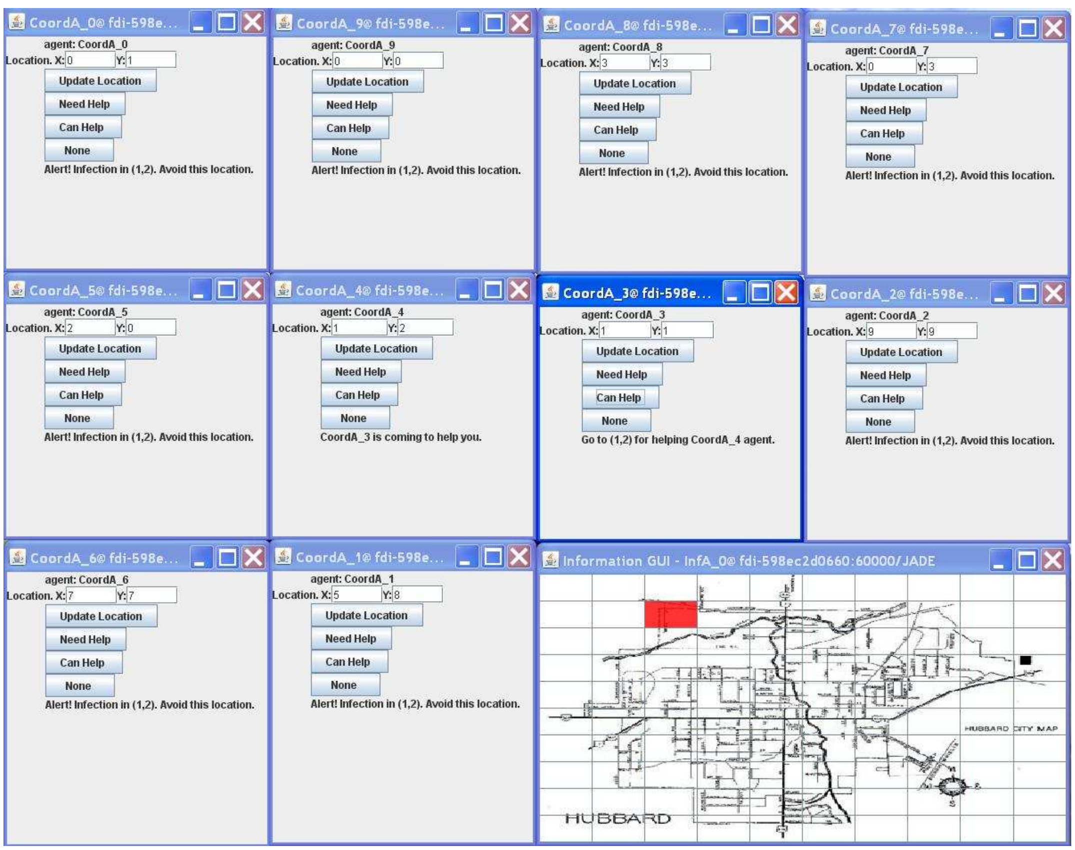


Fig. 3. Execution of the Crisis-management MAS with ten agents. In this execution example, CoordA_3 user offered his help, and CoordA_4 asked for help. Therefore, the MAS coordinates these two user, as one can see in the messages of the graphical user interfaces.

The first experimentation belongs to the metrics and patterns that appear in the deployment of four agents with an ingenuous selection communication; the second one belongs to the metrics and patterns that appear in the deployment of four agents of each type with a random selection communication; the third one contains the results of the response times for both versions of the deployment with four agents; and the fourth experimentation shows all the results for both versions using the deployment of ten agents.

A. Experimental Results in the Crisis System with Ingenuous Communication

Using the metrics from previous sections, concretely $BS(A_j)$, $MS(A_j)$, $BR(A_j)$, and $MR(A_j)$, the results in Table III have been obtained:

TABLE III.
VALUES OF THE METRICS FOR 4 AGENTS AND THE INGENUOUS SELECTION

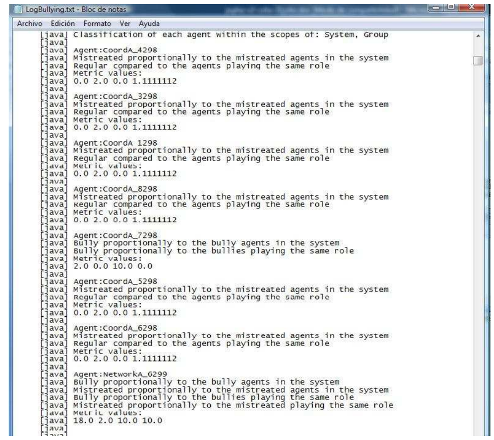
Agent ID	$BS(A_j)$	$MS(A_j)$	$BR(A_j)$	$MR(A_j)$
CoordA_034	0.61	1.85	1.00	1.33
CoordA_134	0.61	1.23	1.00	0.89
CoordA_234	0.61	1.23	1.00	0.89
CoordA_334	0.61	1.23	1.00	0.89
NetworkA_0	0.00	0.00	0.00	0.00
NetworkA_1	5.50	2.46	4.00	4.00
NetworkA_2	0.00	0.00	0.00	0.00
NetworkA_3	0.00	0.00	0.00	0.00

Classification results within both scopes have been collected from this execution as in Table IV:

TABLE IV.
CLASSIFICATION RESULTS FOR 4 AGENTS AND THE INGENUOUS SELECTION

Agent ID	Global system	Agents playing the same role
Coord_034	Regular	Regular
Coord_134	Regular	Regular
Coord_234	Regular	Regular
Coord_334	Regular	Regular
Network_0	Isolated	Isolated
Network_1	Overloader	Overloaded-overloader
Network_2	Isolated	Isolated
Network_3	Isolated	Isolated

In the classification, none of the network agents follows a regular pattern; therefore, there is an unbalanced communication in both scopes. Since a network agent is overloaded while the other network agents are isolated, the problem is the selection mechanism of the network agents in the coordinator-network interaction. The problem is fixed straightforward by changing this mechanism of selection to a more balanced strategy, such as random-selection. In Fig. 4 there is a sample of the metric measurement.



```
LogBullying.txt - Bloc de notas
Archivo  Edición  Formato  Ver  Ayuda
[java] Classification of each agent within the scopes of: System, Group
[java]
[java] Agent:CoordA_4298
[java] Mistreated proportionally to the mistreated agents in the system
[java] Regular compared to the agents playing the same role
[java] Metric values:
[java] 0.0 2.0 0.0 1.1111112
[java]
[java] Agent:CoordA_3298
[java] Mistreated proportionally to the mistreated agents in the system
[java] Regular compared to the agents playing the same role
[java] Metric values:
[java] 0.0 2.0 0.0 1.1111112
[java]
[java] Agent:CoordA 1298
[java] Mistreated proportionally to the mistreated agents in the system
[java] Regular compared to the agents playing the same role
[java] Metric values:
[java] 0.0 2.0 0.0 1.1111112
[java]
[java] Agent:CoordA_8298
[java] Mistreated proportionally to the mistreated agents in the system
[java] Regular compared to the agents playing the same role
[java] Metric values:
[java] 0.0 2.0 0.0 1.1111112
[java]
[java] Agent:CoordA_7298
[java] Bully proportionally to the bully agents in the system
[java] Bully proportionally to the bullies playing the same role
[java] Metric values:
[java] 2.0 0.0 10.0 0.0
[java]
[java] Agent:CoordA_5298
[java] Mistreated proportionally to the mistreated agents in the system
[java] Regular compared to the agents playing the same role
[java] Metric values:
[java] 0.0 2.0 0.0 1.1111112
[java]
[java] Agent:CoordA_6298
[java] Mistreated proportionally to the mistreated agents in the system
[java] Regular compared to the agents playing the same role
[java] Metric values:
[java] 0.0 2.0 0.0 1.1111112
[java]
[java] Agent:NetworkA_6299
[java] Bully proportionally to the bully agents in the system
[java] Mistreated proportionally to the mistreated agents in the system
[java] Bully proportionally to the bullies playing the same role
[java] Mistreated proportionally to the mistreated playing the same role
[java] Metric values:
[java] 18.0 2.0 10.0 10.0
[java]
```

Fig. 4. Extraction of the execution of the overloading measurement system for the crisis management case study with ingenuous selection, for ten agents and a threshold of 1.00. In this phase, for each agent there are the classification patterns for both scopes, and finally the values for the metrics. Notice that in this prototype the patterns have been labeled with different names: bully is used for overloader, and mistreated for overloaded.

B. Experimental Results in the Crisis System with Random-selection Communication

This architecture has the same number of agents than the previous one. As the origin of the unbalanced communication is the mechanism of selection of the coordinator agent, this mechanism of selection is changed. Thus, in this version, coordinator agents randomly interact with any agent of the network role. In this system, the values of the metrics are presented in Table V, and the results of applying the classification rules are collected in Table VI. The values of metrics are closer to 1 and all agents of the system are classified as regular; hence, all network agents have taken part in the system in a proper proportion.

TABLE V.
VALUES OF THE METRICS FOR 4 AGENTS AND THE RANDOM SELECTION

Agent ID	$BS(A_j)$	$MS(A_j)$	$BR(A_j)$	$MR(A_j)$
CoordA_034	0.55	1.38	1.00	0.95
CoordA_134	0.55	1.65	1.00	1.14
CoordA_234	0.55	1.38	1.00	0.95
CoordA_334	0.55	1.38	1.00	0.95
NetworkA_0	1.65	0.55	1.14	1.00
NetworkA_1	0.83	0.55	0.57	1.00
NetworkA_2	1.65	0.55	1.14	1.00
NetworkA_3	1.65	0.55	1.14	1.00

TABLE VI.
CLASSIFICATION RESULTAOF 4 AGENTS AND THE RANDOM SELECTION

Agent ID	Global system	Agents playing the same role
Coord_034	Regular	Regular
…	…	…
Network_0	Regular	Regular
…	…	…

C. Response Times in the Crisis System with Ingenuous Communication and with Random-selection Communication

Table VII shows the response times for each agent and the system in both variants. The system response time is the average of the agents' response times. CoordA_034 agent has no response time because it is the one who offers the service requested. As suspected, response times are lower in the second variant, because the communication is well balanced and therefore, it improves the response time and quality of service. Therefore, the presented metrics are strongly related with the QoS of the system. In Fig. 5 there is a sample of the response time measurement execution for the ingenuous selection version.

TABLE VII.
AGENT AND SYSTEM RESPONSE TIMES IN MILLISECONDS FOR 4 AGENTS

Agent ID/System	Response time (Ingenuous Communication)	Response time (Random-selection Communication)
CoordA_034	N/A	N/A
CoordA_134	2524	1973
CoordA_234	2987	1287
CoordA_334	698	2526
System	2069	1928

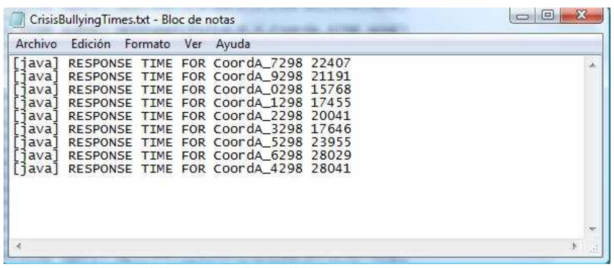


Fig. 5. Extraction of the execution of the overloading measurement system to the crisis management case study with ingenuous selection and ten agents. In this phase, for each agent there is its response time.

D. *Experimental Results for both Versions with ten Agents*

Both versions of Crisis System have been tested with ten agents of each type instead of four. The aim of this experiment is to know how an increase of the number of agents affects to the values of the metrics, the classification of agents, and the response times. Tables VIII, IX, and X show the results for the version with ingenuous selection method, and Tables XI, XII, and XIII contain the results for the random selection method. The thresholds have been the same as in the deployment with four agents of each type:

TABLE VIII.
VALUES OF THE METRICS FOR 10 AGENTS AND THE INGENUOUS SELECTION

Agent ID	$BS(A_j)$	$MS(A_j)$	$BR(A_j)$	$MR(A_j)$
CoordA_034	0.20	1.80	1.00	1.00
CoordA_134	0.20	1.80	1.00	1.00
CoordA_234	0.20	1.80	1.00	1.00
CoordA_334	0.20	1.80	1.00	1.00
CoordA_434	0.20	1.80	1.00	1.00
CoordA_534	0.20	1.80	1.00	1.00
CoordA_634	0.20	1.80	1.00	1.00
CoordA_734	0.20	1.80	1.00	1.00
CoordA_834	0.20	1.80	1.00	1.00
CoordA_934	0.20	1.80	1.00	1.00
NetworkA_0	0.00	0.00	0.00	0.00
NetworkA_1	0.00	0.00	0.00	0.00
NetworkA_2	0.00	0.00	0.00	0.00
NetworkA_3	0.00	0.00	0.00	0.00
NetworkA_4	18.00	2.00	10.00	10.00
NetworkA_5	0.00	0.00	0.00	0.00
NetworkA_6	0.00	0.00	0.00	0.00
NetworkA_7	0.00	0.00	0.00	0.00
NetworkA_8	0.00	0.00	0.00	0.00
NetworkA_9	0.00	0.00	0.00	0.00

TABLE IX.
CLASSIFICATION RESULTS OF 10 AGENTS AND THE INGENUOUS SELECTION

Agent ID	Global system	Agents playing the same role
CoordA_034	Regular	Regular
CoordA_134	Regular	Regular
CoordA_234	Regular	Regular
CoordA_334	Regular	Regular
CoordA_434	Regular	Regular
CoordA_534	Regular	Regular
CoordA_634	Regular	Regular
CoordA_734	Regular	Regular
CoordA_834	Regular	Regular
CoordA_934	Regular	Regular
NetworkA_0	Isolated	Isolated
NetworkA_1	Isolated	Isolated
NetworkA_2	Isolated	Isolated
NetworkA_3	Isolated	Isolated
NetworkA_4	Overloader	Overloader-overloaded
NetworkA_5	Isolated	Isolated
NetworkA_6	Isolated	Isolated
NetworkA_7	Isolated	Isolated
NetworkA_8	Isolated	Isolated
NetworkA_9	Isolated	Isolated

TABLE X.
AGENT AND SYSTEM RESPONSE TIMES IN MILLISECONDS FOR 10 AGENTS AND THE INGENUOUS SELECTION

Agent ID/System	Response time
CoordA_034	N/A
CoordA_134	4219
CoordA_234	12242
CoordA_334	18641
CoordA_434	10701
CoordA_534	19079
CoordA_634	12481
CoordA_734	12502
CoordA_834	16869
CoordA_934	18277
System	13890

TABLE XI.
VALUES OF THE METRICS FOR 10 AGENTS AND THE RANDOM SELECTION

Agent ID	$BS(A_j)$	$MS(A_j)$	$BR(A_j)$	$MR(A_j)$
CoordA_034	0.22	1.77	1.11	0.99
CoordA_134	0.22	1.77	1.11	0.99
CoordA_234	0.00	2.00	0.00	1.11
CoordA_334	0.22	1.77	1.11	0.99
CoordA_434	0.22	1.77	1.11	0.99
CoordA_534	0.22	1.77	1.11	0.99
CoordA_634	0.22	1.77	1.11	0.99
CoordA_734	0.22	1.77	1.11	0.99
CoordA_834	0.22	1.77	1.11	0.99
CoordA_934	0.22	1.77	1.11	0.99
NetworkA_0	2.00	0.22	1.11	1.11
NetworkA_1	0.00	0.00	0.00	0.00
NetworkA_2	2.00	0.22	1.11	1.11
NetworkA_3	0.00	0.00	0.00	0.00
NetworkA_4	4.00	0.44	2.22	2.22
NetworkA_5	2.00	0.22	1.11	1.11
NetworkA_6	2.00	0.22	1.11	1.11
NetworkA_7	2.00	0.22	1.11	1.11
NetworkA_8	2.00	0.22	1.11	1.11
NetworkA_9	2.00	0.22	1.11	1.11

TABLE XII
CLASSIFICATION RESULTS OF 10 AGENTS AND THE RANDOM SELECTION

Agent ID	Global system	Agents playing the same role
CoordA_034	Regular	Regular
CoordA_134	Regular	Regular

CoordA_234	Regular	Regular
CoordA_334	Regular	Regular
CoordA_434	Regular	Regular
CoordA_534	Regular	Regular
CoordA_634	Regular	Regular
CoordA_734	Regular	Regular
CoordA_834	Regular	Regular
CoordA_934	Regular	Regular
NetworkA_0	Regular	Regular
NetworkA_1	Isolated	Isolated
NetworkA_2	Regular	Regular
NetworkA_3	Isolated	Isolated
NetworkA_4	Overloader	Regular
NetworkA_5	Regular	Regular
NetworkA_6	Regular	Regular
NetworkA_7	Regular	Regular
NetworkA_8	Regular	Regular
NetworkA_9	Regular	Regular

TABLE XIII.
AGENT AND SYSTEM RESPONSE TIMES IN MILLISECONDS FOR 10 AGENTS AND THE RANDOM SELECTION

Agent ID/System	Response time
CoordA_034	N/A
CoordA_134	3628
CoordA_234	5944
CoordA_334	6509
CoordA_434	4455
CoordA_534	5482
CoordA_634	4081
CoordA_734	5063
CoordA_834	2193
CoordA_934	10831
System	5354

In the ingenuous version, there is a network agent (NetworkA_4) which is always selected. As suspected, the values of most of the metrics for that agent are higher than the threshold, whereas the rest of the network agents do not participate in any communication. The consequence of this is to have undesirable patterns for the ten network agents, and high response times.

These effects almost disappear in the random selection version, which has the following characteristics:

- There are only two isolated network agents, and the rest become regular.
- NetworkA_4 becomes only overloader for the system scope: the overloaded-overloaded effect has disappeared for the role scope, and the overloaded effect has diminished in the system scope.

Nevertheless, the most relevant feature of this experimentation is that the overloading effect becomes worse than in the previous deployment. This can be shown in the values of the metrics for the overloader agent (NetworkA_4) in Table VIII, which are higher than the overloader agent's results (NetworkA_1) in Table III. This effect is worse in the system response time of Table X, which is much higher than the system response time of Table XIII.

V. RELATED WORK

Load balancing has been pursued in MASs in several ways and applied in multiple scopes. [7] explains a multi-agent reinforcement learning process in MAS load balancing. When choosing a resource, the agent must optimize both the resource usage in the system and fairness by using purely local information. [8] mentions that mobile agents are a paradigm used to balance load in dynamic environments because this type of agent can decide when and where to move, depending on the system conditions.

Though effective, the focus of these works is different from ours: their aim is to build a balanced MAS, whereas the aim of this work is to detect unbalances in the execution of MAS communication.

Although load balancing has been studied extensively in MAS context, the design of analysis tools of debugging data of MAS executions is only preliminary. In [9], a task of clustering has been made to detect of agents with similar communication activity. In [10], clustering algorithms and complex data visualization techniques have been applied to the logs of interactions resulting from the execution of a MAS on the JADE platform.

In contrast, our work does not perform a clustering but a classification task, identifying the agents that cause an unbalanced communication. Other differences are that in the mentioned works they do not use metrics, and their work is not related to the QoS.

[11] has a formalism, called Enhanced Dooley Graphs, to explicate relationships within agent conversations and provide aid to new designs. This work combines the graph representation of the agents that are involved in a conversation and the states agents pass through. In contrast, our work has quantitatively evaluated the communication with metrics and classification rules; and these metrics are empirically proven to be strongly related with QoS. Moreover, the advantage of our approach is that assigning these patterns to agents, created with the IDK, allows the straightforward identification of the interactions that are causing the unbalanced communication. With this information, the designer has just to change the selection mechanism of the affected interaction.

The Quality of Service (QoS) of systems is considered several times in the literature. In many applications there are designs of systems that cover the requirements with a high QoS degree but the solutions are dependable on the scope. In [12], a highly satisfactory QoS adaptive multimedia application is provided by the use of machine-learning techniques that adapts the set of settings to the network capacity. In contrast, our approach improves the QoS by detecting undesirable patterns in the execution and allowing the designer to suppress them by altering the design and implementation of MAS. There are works related to the use of MAS to achieve high QoS levels, like [13], where the system was designed to support resource allocation in cellular data services in such a way that it meets both customer satisfaction and cost effectiveness ends. The solution here was to design agents within three modules built into the scheme: the knowledge source, the blackboard system, and the control engine. The cooperation among the agents provides the correct allocation policy. On the contrary, our approach improves the QoS by means of measuring and improving the communications policy.

VI. Conclusions and Future work

The presented metrics suite measures the communication in MASs, and their measurement values are strongly related with the QoS. The metrics and the presented classification rules assist designers in creating balanced communications in MASs and, consequently, efficient MASs. In this case, the overloading effect has been due to a wrong agent selection mechanism and has been solved by changing it.

According to the scalability, our experiments advocate that the unbalanced patterns of communications have worse effects when MASs are larger. Thus, the presented metrics can become more valuable for huge MASs. Moreover, the proposed metrics can be applied to open MASs, to facilitate the detection of malicious or problematic agents that considerably decrease the performance of the whole open-system.

In the future, the relationship between the metrics and QoS can be corroborated in other MAS methodologies and tools. One of our short-term goals is to corroborate this theory, applying this tool to other MAS with different communication methods than the crisis-management case-study.

Moreover, other metrics are planned to be defined. Firstly, metrics can be defined for the agent scope, that is to say, measures that indicate if the proportion of sent or received messages is too high with respect to the overall amount of sent and received messages by an agent. Secondly, some metrics can be defined to detect overloading behaviors in systems and roles. This may be achieved by defining metrics that extract information from the system and role scopes respectively.

Finally, a long-term goal is to create a framework that uses the presented metrics to dynamically evaluate running MASs. Regarding the results of the metrics, the framework will dynamically change some communication patterns (e.g. modifying the selection mechanism). In this manner, the framework will dynamically improve the QoS of MASs when some inefficient communication patterns are detected.

Acknowledgments

We acknowledge support from the project *Agent-based Modelling and Simulation of Complex Social Systems* (SiCoSSys), supported by Spanish Council for Science and Innovation, with grant TIN2008-06464-C03-01

References

[1] J. Pavón, and J.J. Gómez-Sanz, "An agent oriented software engineering with Ingenias", in *Proc. of the 3rd Intl. Central and Eastern European Conf. on Multi-Agent Systems*, LNCS, vol. 2691, Heidelberg: Springer, 2003, pp. 394-403.

[2] J.J. Gómez-Sanz, R. Fuentes-Fernández, J. Pavón, and I. García-Magariño, "Ingenias development kit: a visual multi-agent system development environment", in *Proc. of the 7th Intl. Conf. on Autonomous Agents and Multi-Agent Systems*, 2008, pp. 12-16.

[3] M. Wooldridge, *An Introduction to MultiAgent Systems*, John Wiley & Sons Ltd., 2002.

[4] W. Jiao, J. Debenham, and B. Henderson-Sellers, "Organizational models and interaction patterns for use in the analysis and design of multi-agent systems", *Intl. J. of Web Intelligence and Agent Systems*, vol. 3, no. 2, 2005, pp. 67-83.

[5] B. Bauer, J.P. Muller, and J. Odell, "Agent UML: A formalism for specifying multiagent interaction", *Agent-Oriented Software Engineering*, LNCS, vol. 1957, Heidelberg: Springer, 2001, pp. 109-120.

[6] I. García-Magariño, C. Gutiérrez, and R. Fuentes-Fernández, "The INGENIAS Development Kit: A Practical Application for Crisis-Management", *Bio-Inspired Systems: Computational and Ambient Intelligence*, LNCS, vol. 5517, Heidelberg: Springer, 2009, pp. 537-544.

[7] S. Andrea, Y. Shoham, and M. Tennenholtz, "Adaptive load balancing: a study in multi-agent learning", in *J. of Artificial Intelligence Research,* vol. 2, 1995, pp. 475-500.

[8] J. Gomoluch, and M. Schroeder, "Information agents on the move: a survey on load-balancing with mobile agents", *Software Focus*, vol. 2, 2001, pp. 31-36.

[9] J.A. Botia, J.M. Hernansaez, and A.F. Gomez-Skarmeta, "On the application of clustering techniques to support debugging large-scale multi-agent systems", *Programming Multi-Agent Systems*, LNAI, vol. 4411, Heidelberg: Springer, 2007, pp. 217-227.

[10] J.A. Botia, J.J. Gomez-Sanz, and J. Pavon, "Intelligent data analysis for the verification of multi-agent systems interactions", *Intelligent Data Engineering and Automated Learning*, LNCS, vol. 4224, Heidelberg: Springer, 2006, pp. 1207-1214.

[11] H.V.D. Parunak, "Visualizing agent conversations: using enhanced dooley graphs for agent design and analysis", in *Proc. of 2nd Intl. Conf. on Multi-Agent Systems*, 1996, pp. 275-282.

[12] P. Ruiz, J. Botia, and A. Gomez-Skarmeta, "Providing QoS through machine-learning-driven adaptive multimedia applications", *IEEE Trans. Systems, Man, and Cybernetics-Part B* , vol. 34, no. 3, pp. 1398-1411, Jun. 2004.

[13] J.-L. Chen, "Resource allocation for cellular data service using multiagent schemes", *IEEE Trans. Systems, Man, and Cybernetics-Part B* , vol. 31, no. 6, pp. 864-869, Dec. 2001

Multi-Agent System Theory for Resource Management in Home Automation Systems

G. Conte, G. Morganti, A. M. Perdon and D. Scaradozzi

Abstract—**A paradigm for modelling and analysing Home Automation Systems is introduced, based on Multi-Agent System theory. A rich and versatile environment for Home Automation System simulation is constructed for investigating the performances of the system. The exploitation of limited resources (electricity, gas, hot water) depends on behavioural parameters of the individual appliances. In order to deal with the problem of developing systematic design and validation procedures for control strategies, global performance indices for the system are introduced. Different strategies for allocating resources and establishing priorities in their use can therefore be tested and compared.**

Index Terms—**Home Automation Systems, Domotics, Multi-Agent Systems, Resource Management, Energy Management, Simulation Optimisation.**

I. Introduction

THE aim of this paper is to present an approach to the study of Home Automation systems, based on the Multi-Agent System (MAS) theory, which allows to introduce a rigorous formalisation and to define practical tools for the analysis of performances.

Home Automation Systems (HAS) consist of a number of appliances (e.g. washing machine, dishwasher, oven, refrigerator, boiler) and other devices or subsystems (e.g. for heating, air conditioning, entertainment), which are connected by a communication network of some kind. The various elements of the system share common resources (mainly electricity, but also water) to accomplish their specific tasks, according to user requirements.

Since resources are limited (due to costs, contract rules or provider directives), competition between the elements of the system generates internal conflicts. Distributed or centralised control strategies have to be implemented within the system, so to allocate limited resources while maximising performances, using the information dispatched over the communication network and the computational resources the devices, or a dedicated unit of them, may have [1].

From a control theory point of view, Home Automation Systems are very complex objects, characterised by the presence of components and constraints of different kinds, a distributed structure and a hybrid event/time-driven behaviour. Although the market offers devices and solutions that allow to assemble more or less sophisticated Home Automation Systems, efficient design techniques are far from being developed.

G. Conte, G. Morganti, A. M. Perdon and D. Scaradozzi are with Dipartimento di Ingegneria Informatica, Gestionale e dell'Automazione - Università Politecnica delle Marche - Via Brecce Bianche, 12 - 60131 Ancona, Italy.
E-mail: gconte@univpm.it, morganti@diiga.univpm.it, perdon@univpm.it, d.scaradozzi@univpm.it

In particular, general methods for evaluating the performances and for tuning the control strategies are missing. On the other hand, the demand for methods and tools that facilitate the systematic design of efficient Home Automation Systems is made more and more urgent by the increase of the quantity of limited resources that are consumed for domestic use and by the growing competition in the market of appliances and home devices [2].

Here, we present and discuss a specific contribution to the construction of tools for the analysis and design of Home Automation Systems. We start by recalling the notion of Home Automation System based on the Multi-Agent System theory, that was introduced and discussed in [3–6].

On the formal basis that notion provides, we construct an environment for the simulation and emulation of Home Automation Systems. At the same time, we define indices of performances for paradigmatic control strategies, characterised by suitable parameters, that allocate limited resources, by solving conflicts between individual components of the system.

The outcome of this work is a rich and versatile framework in which one can analyse the effects caused on the performance indices by variations in the parameter that characterise the control strategies. In other term, control strategies for Home Automation Systems can be validated and tuned by exploiting the results of simulation, so to increase user satisfaction and system efficiency.

II. Multi-Agent System and Home Automation System definition

Multi-Agent System theory has been employed to model and study several situations, characterised by the presence of autonomous agents which interact and possibly compete in various way, in many areas of computer science, automation and robotics (see e.g. [3, 4, 7] and the references therein).

Another application is the IHome Environment [8], an example of intelligent home that uses the Multi-Agent System technology as a way to control the behaviour of house appliances from the resource consumption and coordination viewpoint. The simulated IHome environment is controlled by intelligent agents that are associated with particular appliances.

In the MavHome project [9] the smart home is seen as an intelligent agent that perceives its environment through the use of sensors, and can act upon the environment through the use of actuators. The home has certain overall goals, such as minimising the cost of maintaining the home and maximising the comfort of its inhabitants.

C@sa [10] is a Multi-Agent system aiming at modelling, controlling and simulating house behaviour according to user and context features, using several “Operator Agent” and a “Supervisor Agent”.

Another notable environment is Adaptive House [11], that focuses on the development of a home that programs itself by observing the lifestyle and desires of the inhabitants, and learns to anticipate and accommodate their needs. Although Multi-Agent System theory is not used in this work, this system is interesting since the control is handled by neural network reinforcement learning and prediction techniques.

Anyway, in all these works, the resource management strategies can neither be fine-tuned nor evaluated by means of well-defined performance indices. Moreover, the constraints on resource consumption are quite generic and not as “hard” as in the present work.

Our approach is based on the use of very special agents, that are not to be seen as computer science agents; we provide an appropriate definition of “Domotic Agent”, that stems from the more general definition of agent as an entity who is capable of action.

A. Home Automation Systems as Multi-Agent Systems

As we have already mentioned, Home Automation Systems consist of appliances and other devices or subsystems, connected by a communication network. Communication may happen through the power line, by means of suitable devices called nodes.

From the point of view we have adopted, we can consider the human user himself as an agent of the Home Automation System. Competing for the resources, his needs will conflict with those of other agents in the same structured environment and the solution of conflicts will follow the same general rules. Privileges and priorities granted to the human user do not represent external disturbances which may interfere with system behaviour and degrade its performances, but are integrated in the laws that govern its operations.

In a real Home Automation System, the agents must have some individual features that facilitate their integration into a larger system, without reducing their ability to work as stand-alone devices and to satisfy the user. Summarising, the features can be informally described as follows:

- *Autonomy*: the capability to perform their own tasks, when required, without support from the Home Automation System or, alternatively, the capability to negotiate resources in interacting with other agents
- *Collaborative behaviour*: the capability to cooperate with other agents in order to share resources and to reach common goals
- *Adaptability*: the capability to modify their behaviour according to the outcome of interaction processes
- *Reactivity*: the capability to react, to some degree, to system actions.

The above features assure the possibility to work sharing common resources, and suitable strategies for optimising individual performances are applied [6].

Multi-Agent approach is very suited to deal with such a situation and allows to model the structure of a general Home Automation System at least at two levels: a global level concerns the overall behaviour and performances of the system, and a local level that concerns the way in which single components and devices are integrated into the system and work.

In other words, in a Home Automation System we have a collection of components, which can be generically called agents, and an overall architecture, which defines the environment in which the agents interact and the modalities of interaction. From this point of view, a Home Automation System qualifies as a Multi-Agent System in the sense of Sycara [3], in which, in particular,

- there is no centralised, global controller or supervisor;
- data and knowledge are decentralised;
- computing power is distributed and asynchronous;
- each agent has incomplete information or capabilities and, as a consequence, limited knowledge and awareness of the overall situation.

In order to allow the system to operate, the architecture of the system and the way in which information flows in it must guarantee that

- each agent may declare itself;
- each agent may detect and possibly recognise other agents;
- each agent may interact with other agents according to given rules (system rules).

B. Domotic Agent

Following the Multi-Agent System theory point of view, an agent, namely the basic component of a larger system, can be defined by characterising its capabilities. Keeping in mind the application we are interested in, a suitable way to characterise an agent is provided by the following definitions.

Definition 1: An *agent* is a virtual or physical entity that:

1) is able to perform specific actions in a given environment;
2) is able to perceive elements of the environment;
3) is able to construct (partial) models of the environment
4) is able to use personal resources and environmental resources;
5) is able to direct its actions toward specific goals;
6) is able to communicate directly with other agents in the environment;
7) is able to offer services to other agents;
8) is able to govern its action according to its possibilities and limitations, to its goals, to its knowledge of the environment, to the resources available.

Remark 1: This definition is formally analogous to the definition of agent in Ferber [7]. It should be noted, as stated in Russell and Norvig [12], that in general the definition of agent “is meant to be a tool for analyzing systems, not an absolute characterization”. For a more extensive discussion of the issues involved in the general definition of agent, the reader is referred to [13].

Now, we can give the notion of *domotic object* and of *domotic agent*, the elementary component of a Home Automation System, by specialising Def. 1.

Definition 2: A *domotic object* is a generic agent in the sense of Definition 1 that has at least the general capacities 1, 4, 5 and 8 and, concerning capacity 6, it is able to communicate to other agents in the environments at least its requirements about environmental resources.

Definition 3: A *domotic agent* is a domotic object that, in addition, has at least the general capacity 2. A *cognitive domotic agent* has also capacity 3.

Remark 2: Note that in this framework "cognitive" is not meant to define faculty of cognition in general, but with respect to the object of our studies.

The above definitions are quite abstract, but, as described in [5], they fit well with the appliances and other devices found in a domestic environment.

C. Home Automation System

After having defined the main elements, namely the domotic object and agents, that will form the overall system, we can give the following Definition.

Definition 4: A *Home Automation System (HAS)* consists of:

1) a set GR of Global Resources;
2) a set DO of Domotic Objects;
3) a set DA of Domotic Agents, subset of DO;
4) one Information Network IN, that connects domotic objects and agents;
5) a set R of Rules that govern the individual behaviour and the concurrent operation of domotic objects and concern
    a) use and transformation of external resources,
    b) communication,
    c) perception and understanding;
6) a set L of operators, called Laws, that describe the time evolution of the global system according to the individual behaviour of objects and of agents.

The notions described must be viewed as conceptual instruments that facilitate modelling and analysis of concrete examples, help in understanding their features and in identifying their critical parameters.

Remark that, in facts, the time evolution of a Home Automation System formally described on the basis of Def. 4 is completely determined by L and it depends, in particular, on the Rules which form R. Then, developing simulation tools and procedures as described in the next Section, it is possible to investigate the effects of different choices of the Rules on the global evolution and behaviour of the system and to evaluate the resulting performances in terms of indices that represent user satisfaction.

III. Simulation Environment

In the virtual environment we construct, agents are characterised in a realistic way, by exploiting the experience in studying and modelling white goods derived from the collaboration with Italian manufacturers. Besides making simulation more effective, this choice is crucial to allow the simulation engine to interface with real appliances.

Anyway, since the duration of an appliance cycle may be quite long, one of the important feature of the simulator is that

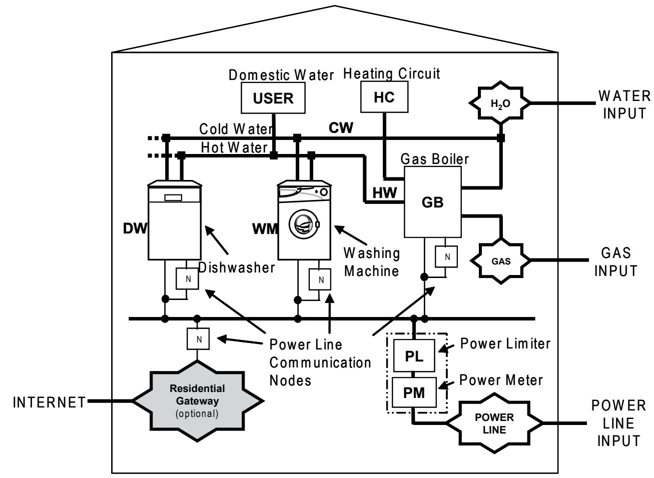


Fig. 1. Home Automation System.

it can work only with virtual agents, whose execution speed can be increased setting a specific parameter, so to shorten the duration of simulation.

The behaviour of each agent in the simulator is modelled as a sequence of transitions from state to state. The sequence of state transitions during normal operation of each agent is time-driven, except in case a required resource is or becomes unavailable. In such case, this event may cause a transition.

Basically, the simulator consists of a software environment, developed using LabVIEW, where programs (virtual instruments) that represent the single agents are executed simultaneously; each program implements the sequence of state transitions that characterises an agent. Agents exchange information by sharing global variables.

To run a simulation, one has to define the virtual environment by indicating the agents and the related global variables and by specifying the values of a number of global and individual (i.e. concerning a single agent) parameters. During the simulation, agents are put into action automatically, according to a chosen schedule, so to simulate normal operations in a domestic environment.

The set of agents in the Home Automation System is composed of the following: washing machine, dishwasher, boiler, human user and metering devices. Other agents can easily be added, if necessary, such as oven, vacuum cleaner, etc. A graphic representation of the considered Home Automation System is given in Fig. 1.

A. Control strategies

Allocation of limited resources is the key problem that the Home Automation System we consider has to deal with, by solving conflict between competing agents. Here, we assume that limitation of resources means that the actual consumption must not exceed a fixed threshold for a limited time T_{lim}.

The need of a control strategy for electricity consumption is particularly perceived in typical Italian home installations, where the contract with the provider determines expressly a 3 kW threshold and T_{lim} may vary around a few minutes;

consumption over the threshold for a time greater than T_{lim} causes a blackout. In this case the user has to switch off some appliance to go back under the threshold and put the master switch on again; it is undoubtedly an unwanted event, not least because sometimes the master switch is placed outside home.

Other possible goals could be minimise the cost, if for instance different rates for resources apply depending on peak or off-peak hours.

Denoting by *overload* the condition in which the threshold is surpassed and analysing available resources and their quantitative limits, we have found convenient to classify the resources in two groups:

1) resources for which overload can persist, without causing degradation in quality, for a (short) time $T_{lim} > 0$, after which, if overload persists, its use becomes unacceptably expensive, or, alternatively, the resource is cut off, making necessary to intervene for restarting the system;
2) resources for which overload could persist indefinitely, without causing cut off and the necessity of restarting the system, but whose persistence in time degrades relevantly the quality, so that it is appropriate to chose $T_{lim} = 0$.

Two paradigmatic examples are that of electricity, for which T_{lim} is greater than 0 and that of hot water, for which T_{lim} is ideally 0, since overloads would cause an abrupt decrease of temperature and/or flow.

Resources belonging to different groups have to be managed with different strategies.

Overall control strategies for managing first group resources can be designed following the lines of the Power Levelling strategy introduced and discussed in [1, 5]. This means that the action of each home automation agent is determined by two parameters, respectively the *Overload Time* t_o and the *Suspension Time* t_s. The Overload Time t_o represents the time the agent waits before stopping its action in case an overload occurs. The Suspension Time t_s represents the time the agent waits before restarting its action after having stopped it because of an overload.

Overall control strategies for managing second group resources can be designed following the lines of the Water Levelling strategy. In case overload occurs when more than one agent at a time tries to use the resource, planning and evaluation turn out to be particularly simple. To do so, in fact, an agent has only to be able to check, before using the resource, if others are already doing so and, in case, to renounce. This kind of strategy assures the highest priority to the agent which first gets the resource and forces the others to wait.

B. Performance indices

In order to compare the results of different simulation scenarios, determining thereby which strategies and policies are the best, we need one or possibly more indices of performance.

We have defined two indices, that point at important aspects: relative delay in completion of tasks and number of overloads. Other choices are possible if different problems have to be investigated.

1) Relative delay: Let the duration of task i, when all the required resources are available, be indicated by TN_i and call it nominal duration. In case of conflict and competition, during a simulation, the time needed to complete the task may take a higher value $TA_i \geq TN_i$. Then, we consider the relative delay

$$\Delta_i = \frac{TA_i - TN_i}{TN_i} \tag{1}$$

and, letting N be the number of tasks in the considered scenario, we define the global percent relative delay $\Delta\%$ as

$$\Delta\% = 100 \cdot \frac{\sum_{i=1}^{N} p_i \Delta_i}{N} \tag{2}$$

where the p_i are weights that can be chosen according to the characteristics of the task. Clearly, user satisfaction increases when this index approaches 0. Nominal durations have been determined using real appliances.

2) Number of overloads: This index simply counts the occurrences of overloads during the simulation and it is denoted by OL. Overloads stress the system and put it at risk or cause a great increase of costs, so the Home Automation System is performing better if OL is kept small.

C. Parameter optimisation

To optimise the choice of parameters in order to achieve better performances is the object of a relatively new research area, the simulation optimisation.

The capabilities of the simulator have been tested on several experiments on a simple Home Automation System, using the Power Levelling strategy with different parameters, and different parameter optimisation procedures have been implemented: a Tabu Search and a Genetic Algorithm [14, 15].

As an example, we can mention the simulation results obtained using the Tabu Search approach, without entering into the optimisation method details, for which the reader is referred to [14].

The parameters that we want to optimise are the Overload Time and the Suspension Time for both the washing machine (WM) and the dishwasher (DW). As first starting solution, we decided to choose a situation of equilibrium with the same parameter values for both the appliances, i.e. $t_o = 50$ s and $t_s = 1000$ s.

Running the simulation with these parameters resulted in 13 overloads and 287.1% as relative delay; clearly it was a very bad solution: the great number of overloads stresses the whole system and the accomplishment of the tasks requires a duration that is almost four times as the expected one.

Applying the optimisation method, the best solution (WM: $t_o = 30$ s, $t_s = 1100$ s; DW: $t_o = 90$ s, $t_s = 700$ s) was found after 14 iterations and it resulted in 2 overloads and 22.6% of relative delay.

The performance of this solution is significantly better than the starting solution, so we can say that our framework has been very effective in finding a good set of parameters starting from a set chosen with no knowledge or prediction about the relationship between parameters and performance index, which was a very poor solution.

IV. Conclusions and Future Directions

Starting from the formal definitions of basic concepts in home automation we have shown how a Multi-Agent approach has been used to develop a coherent, general framework for modelling, simulating and analysing Home Automation Systems. From a conceptual point of view, a key feature of this approach is the possibility of defining indices of performance, that turn out to be functional in designing and optimising global control strategies and behavioural policies.

The simulator has proven to be a useful tool that allows to adopt the most effective strategy for each scenario; when appliances are added or discarded from the house or the restrictions on the resources change, a new simulation optimisation can be easily performed.

Actually we are investigating metaheuristic methods, such as genetic algorithms, for parameter optimisation as well as other simulation optimisation methods in order to design control strategies for energy saving and optimal resource exploitation. Another promising direction could be the use of Artificial Neural Networks to determine a set of resource management parameters that optimise the performance indices.

References

[1] M. Cipriani, "Bringing new services and benefits in a changing home environment," in *LonWorld*, 2001.

[2] V. Vishwanathan, J. McCalley, and V. Honavar, "A multi agent system infrastructure and negotiation framework for electric power systems," in *Power Tech Proceedings*, 2001.

[3] K. Sycara, "Multiagent systems," *AI Magazine*, vol. 10, no. 2, pp. 79–93, 1998.

[4] R. A. Flores-Mendez, "Towards a standardization of multi-agent system framework," *Crossroads*, vol. 5, no. 4, pp. 18–24, 1999.

[5] G. Conte and D. Scaradozzi, "Viewing home automation systems as multiple agents systems," in *Multi-agent system for industrial and service robotics applications, RoboCUP2003*, 2003.

[6] D. Scaradozzi, "Methodologies and techniques for analysis and design of home automation systems," Ph.D. dissertation, Università Politecnica delle Marche, 2005.

[7] J. Ferber, *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1999.

[8] V. Lesser, M. Atighetchi, B. Benyo, B. Horling, A. Raja, R. Vincent, T. Wagner, X. Ping, and S. X. Zhang, "The intelligent home testbed," in *Proceedings of the Autonomy Control Software Workshop (Autonomous Agent Workshop)*, Seattle, January 1999. [Online]. Available: http://mas.cs.umass.edu/paper/134

[9] S. P. Rao and D. J. Cook, "Predicting inhabitant actions using action and task models with application to smart homes," *International Journal of Artificial Intelligence Tools*, vol. 13, pp. 81–100, 2004.

[10] B. De Carolis, G. Cozzolongo, S. Pizzutilo, and V. L. Plantamura, "Agent-based home simulation and control," in *ISMIS*, 2005, pp. 404–412.

[11] M. C. Mozer, "Lessons from an adaptive house," in *Smart environments: Technologies, protocols, and applications*, D. Cook and S. Das, Eds. J. Wiley & Sons, 2004, ch. 12.

[12] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Prentice Hall, 1995.

[13] S. Franklin and A. Graesser, "Is it an agent, or just a program?: A taxonomy for autonomous agents," in *Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages*. Springer-Verlag, 1996. [Online]. Available: http://www.msci.memphis.edu/˜franklin/AgentProg.html

[14] G. Conte, D. Scaradozzi, A. Perdon, and G. Morganti, "Parameter tuning in distributed home automation systems: towards a tabu search approach," in *MED 2008: 16th IEEE Mediterranean Conference on Control and Automation, Ajaccio, France*, 2008.

[15] G. Morganti, A. Perdon, G. Conte, D. Scaradozzi, and A. Brintrup, "Optimising home automation systems: a comparative study on tabu search and an evolutionary multi-objective algorithm," in *MED 2009: 17th IEEE Mediterranean Conference on Control and Automation, Thessaloniki, Greece*, 2009.

A THOMAS Based Multi-Agent System for Recommendations and Guidance in Malls

S.Rodríguez, A.Fernández,V.Julián, J.M.Corchado, S.Ossowski and V.Botti

***Abstract*— This article presents a case study in which the THOMAS architecture is applied in order to obtain a multi-agent system (MAS) that can provide recommendations and guidance in a shopping mall. THOMAS is made up of a group of related modules that are well-suited for developing systems in other highly volatile environments similar to a shopping mall. Because the development of this type of system is complex, it is essential to thoroughly analyze the intrinsic characteristics of typical environment applications, and to design all of the system components at a very high level of abstraction.**

***Index Terms*— Multi-Agent Systems, Virtual Organization, Open Multi-Agent Systems, Dynamic Planning.**

I. Introduction

This article presents a dependable solution for using a novel architecture in designing and building a system for guiding and advising users in a shopping mall. A shopping mall can be considered a dynamic and volatile environment in which shops change, promotions appear and disappear, and the products that are offered are continually changing. As such, a high level design with an abstract architecture is essential.

The architecture we used is THOMAS (*MeTHods, techniques and tools for Open Multi-Agent Systems*) [6][7]. THOMAS is a new architecture for open MAS and is made up of a group of related modules that are well-suited for developing systems in other highly volatile environments similar to a shopping mall. This design will use a high level of abstraction to determine which components are necessary for addressing all of the needs and characteristics of a shopping mall guidance system.

Artificial intelligence techniques have given way to new studies that allow, among other things, modeling the problem of a shopping mall in terms of agents and MAS. The shopping mall is turned into an intelligent environment where users are surrounded by these techniques, but do not need to adapt to them. One of the objectives of MAS is to build systems capable of autonomous and flexible decision-making, and that will cooperate with other systems within a "society" [5]. This "society" must consider characteristics such as distribution, continual evolution and flexibility, all of which allow the members (agents) of the society to enter and exit, to maintain a proper structural organization, and to be executed on different types of devices. All of these characteristics can be incorporated via the open MAS and virtual organization paradigm, which was conceived as a solution for the management, coordination and control of agent performance [8]. The organizations not only find the structural composition of agents (i.e., functions, relationships between roles) and their functional behavior (i.e., agent tasks, plans or services), but they also describe the performance rules for the agents, the dynamic entrance and exit of components, and the dynamic formation of groups of agents[3].

The goal of this study is to present a case study in which the THOMAS architecture is used to build an open MAS for guiding users through a shopping mall. We will propose an application for this architecture and will evaluate its appropriateness for developing an open MAS in a real environment. The first step of this research involves designing the components needed for addressing all the needs and characteristics of a shopping mall system. The design is based on the GORMAS (Guidelines for Organization-based Multi-Agent Systems) [1] methodology, which is specifically geared towards organizations.

This article is organized as follows: section 2 presents the principle characteristics of the architecture and methodologies used; section 3 indicates the MAS that was developed for the actual case study (the shopping mall), and highlights the characteristics provided by the type of architecture used for its development; and the final section presents some of the conclusions obtained by this research.

II. THOMAS Outline

THOMAS [6][7] is the name given to an abstract architecture for large scale, open multi-agent systems. It is based on a services oriented approach and primarily focuses on the design of virtual organizations.

The architecture is basically formed by a set of services that are modularly structured. THOMAS uses the FIPA architecture, expanding its capabilities with respect to the design of the organization, while also expanding the services capacity. THOMAS has a module with the sole objective of managing organizations that have been introduced into the architecture, and incorporates a new definition of the FIPA Directory Facilitator that is capable of handling services in a much more elaborate way, following the service-oriented

S. Rodríguez, J. M. Corchado are with University of Salamanca.
E-mails: srg@usal.es , corchado@usal.es
A.Fernández and S.Ossowski are with University Rey Juan Carlos of Madrid.
V.Julián and V.Botti are with Polytechnic University of Valencia.

architecture directives.

THOMAS consists of three principle components: *Service Facilitator (SF), Organization Manager Service (OMS) and Platform Kernel (PK)*.

The SF primarily provides a place where autonomous entities can register service descriptions as directory entries. The OMS component is primarily responsible for specifying and administrating its structural components (role, units and norms) and its execution components (participating agents and the roles they play, units that are active at each moment).

In order to manage these components, OMS handles the following lists:

- *UnitList*: maintains the relationship between existing units and the immediately superior units (SuperUnit), objectives and types.
- *RoleList*: maintains the relationships between existing roles in each unit, which roles the unit inherits and what their attributes are (accessibility, position).
- *NormList*: maintains the relationship between the system rules.
- *EntityPlayList*: maintains the relationship between the units that register each agent as a member, as well as the role that they play in the unit.

Each virtual unit in THOMAS is defined to represent the "world" for the system in which the agents participate by default. Additionally, the roles are defined in each unit. The roles represent the functionality that is necessary for obtaining the objective of each unit. The PK component directs the basic services on a multi-agent platform and incorporates mechanisms for transporting messages that facilitate the interaction among the various entities.

From a global perspective, the THOMAS architecture offers a total integration enabling agents to transparently offer and request services from other agents or entities, at the same time allowing external entities to interact with agents in the architecture by using the services provided.

The development of MAS is typically based on a design that focuses on each agent independently, and is geared towards each agent's structure and performance. This research presents a new focus in which the design is directed at the organizational aspects of the agents, establishing two descriptive levels: the organization and the agent [4]. The system we developed used the GORMAS [1] organizational methodology.

III. CASE OF STUDY: TORMES SHOPPING MALL

The case study application facilitates the interaction between the users (clients in the shopping mall), the store or sales information, and recreational activities (entertainment, events, attractions, etc.), and defines the services that can be requested or offered. We developed a wireless system capable of incorporating agents that provide orientation and recommendation functionalities to the user, and that can be applied not only in a shopping mall, but also in other similar environments such as a supermarket, an educational facility, medical or health care center, etc [2].

The clients use the agents via their mobile devices and RFID (Radio Frequency Identification) [9] technology in order to consult the store directory, receive special offers and personalized promotions, and ask for recommendations to navigate through the mall or locate other clients. Clients can also use the mechanisms available to them to plan a particular route that allows them to better spend their time in the mall and receive personalized notices.

There are different types of agents that come into play:

- *User agent*, which is in charge of managing client communication, finding and identifying other user devices, and maintaining the user's profile.
- *Shop agent*, which is in charge of maintaining the warehouse (i.e., product database) and the promotions that can be offered to the clients.
- *Guiding agent*, which is charge of managing user profiles, controlling communications, analyzing the promotions, managing all incidents, and most importantly, planning the best route for each user according to the available resources and the user profile.

The Guiding agent can be considered the heart of the system, as it receives the most current information from each of the mall's stores, and interacts directly with the clients to offer personalized services.

The first step in analyzing and designing the problem is to define the following roles that will exist within the architecture:

- *Communicator*: in charge of managing the connections that each user makes.
- *Finder*: in charge of finding users with similar tastes.
- *Profile Manager*: in charge of creating and defining the client profile.
- *Promotions Manager*: in charge of suggesting promotions and offers.
- *Warehouse Operator*: in charge of managing all inquiries made on the warehouse, managing updates and monitoring product shortages.
- *Analyst*: in charge of auditing sales information and the degree of client satisfaction.
- *Planner*: offers recommendation and guidance services to the shopping mall clients. This role is able to dynamically plan and replan in execution time. It suggests routes that clients might want to take through the mall, according to their individual preferences.
- *Client Manager*: in charge of managing the connections between the mall clients, managing the profiles for clients that are visiting the mall, monitoring the state of the clients, and managing the notification service for the mall.
- *Incident Manager*: manages and resolves incidents,

offers a client location service, and manages an alarm system.

- *Directory Manager*: responsible for managing the mall's store directory, including businesses, products, promotions and offers.
- *Device Manager*: makes it possible for the interactive elements within the environment to interact. It deals with devices that use technologies such as RFID, etc.

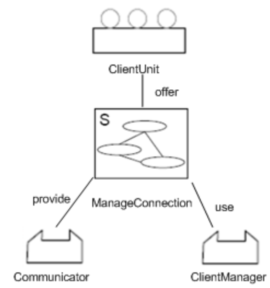


Fig. 1. Diagram of organization model: functional view of *ClientUnit*

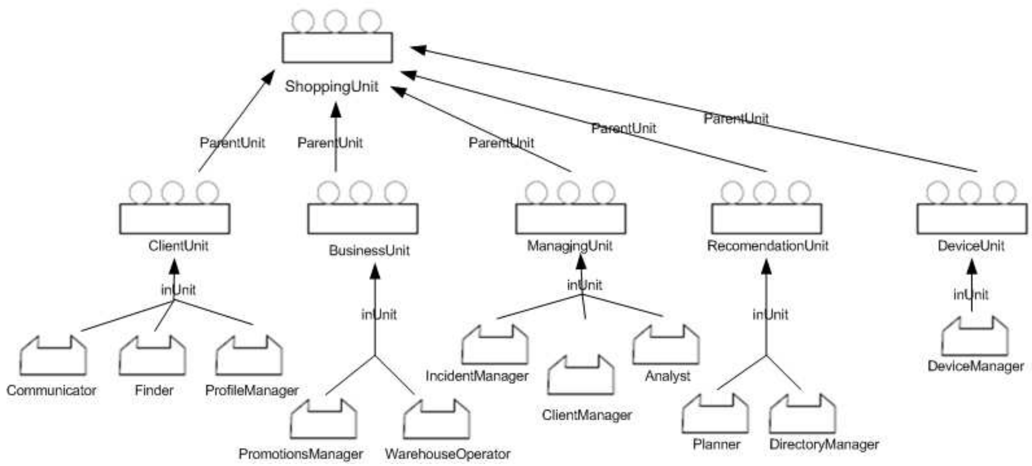


Fig. 2. Diagram of organization model: structural view

TABLE I
MANAGECONNECTION SERVICE IN CLIENTUNIT

Service Specification					
Name: manageConnection **Description:** Manage client connection **Supplied by:** SF **Required by:** **ClientRole:** ClientManager **ProviderRole:**Comunnicator					
Input Parameters					
Name	**Description**	**Mand.**	**Type**	**Value Range**	**Default**
requestTime connectionData operation	Connection time Connection Data Kind Conecction	Yes Yes Yes	date string string		
Output Parameters					
Name	**Description**	**Mand.**	**Type**	**Value Range**	**Default**
connection	Connection established	Yes	connection		
Precondition					
- -					
Postcondition					
- -					

We have also designed an organizational structure. We will first analyze its dimensions, and then proceed to identify the structure that is best suited to apply to the system [1]. Our case study is modeled as a conglomerate (*ShoppingUnit*) made up of five units, each one dedicated to one type of functionality within the setting. The five units are:

- *ClientUnit*, contains the roles associated with the client: Communicator, Finder, and Profile Manager.
- *BusinessUnit*, contains the roles associated with a business: Promotions Manager, Warehouse Operator.

- *ManagingUnit*, contains the roles assigned with global management tasks for the mall: Incident Manager, Client Manager, and Analyst.
- *RecommendationUnit*, contains the roles dealing with recommendations or suggestions made to the client: Planner and Directory Manager.
- *DeviceUnit*, which contains the roles associated with the management of devices: Device Manager.

The diagram in Figure 2 provides a structural view of the organizational model, which is adapted according to a conglomerate pattern. Different services are provided within each unit of the organization. The following section defines the services offered by the units, and uses an example to detail each one and how it has been modeled and described in the architecture. The type of role, the inputs and outputs, and a summary of the functionality for each unit are all explained. Figure 1 shows the internal model of the *ClientUnit* and Table I shows an example of service specification (*ManageConnection)*. The internal structure for *ShoppingUnit* and the remaining units was modeled in the same way.

One side of the diagram models the functional views of the units, which allows us to identify the services specific to each domain, while the other side precisely details the behavior of the organization services, how they interact with the environment, which interactions are established between the system entities, and how they approach the aspects of an open system. The next step is to define the rules in order to establish the control and management of the services. For example, the basic service provided by *ClientUnit* will be *ManageConnection*, which is provided by the agents that take on the role of Communicator. The functionalities offered by this service will allow the clients to control their connection to the system.

Similarly, within the *BusinessUnit* there are roles associated with a particular business and as a result, the services offered will be related to the corresponding promotions, products and sales (e.g., *SendPromotion* or *RetrievePromotion*). The services related to *ManagingUnit* involve the overall management tasks within a shopping mall (e.g., system incidents, data analysis, surveys, client management, notices, etc.). *RecommendationUnit* is comprised of services that request recommendations or suggestions based on user preferences and certain restrictions (time, money, etc.). It also includes planning and replanning the route that the user will follow based on the suggested recommendations, and determines the validity and value of the proposed routes. The *DeviceUnit* services deal with the sensors embedded in the physical system (RFID).

The type of services offered is controlled by the system according to the established norms [7]. The internal functionality of the services is responsible for the agents that are offered, but the system is what specifies the agent profiles, as well as the rules to follow for ordering requests or offering results. In this way, when faced with illicit or improper client performance, the system can act to impose sanctions. The OMS will internally save the list of norms that define the role involved, the content of the norms, and the roles in charge of ensuring that the norm is met. We have defined a set of norms in our system for controlling the performance within each unit. This way, for example, an agent within *ClientUnit* that acts like Communicator is required to register a service as *manageConnection*. If it does not abide by these norms, it will be punished and expelled from the unit. The punishment is logical given that if the agent does not establish a connection within the allocated time, it cannot perform any of the other system tasks.

```
OBLIGED Communicator REGISTER
manageConnection(?requestTime, ?connectionData,
?operation) BEFORE deadline SANCTION (OBLIGED OMS
SERVE Expulse (?agentID Comunicator ClienteUnit))
```

Similarly, we have defined a complete set of norms that will control all of the system performances.

A. Example of service planning with THOMAS

The system considers the client objectives, the available time, and financial limitations, and proposes the optimal route according to the client's profile. The planning model we propose was integrated within a previously developed MAS [2]. We will see the series of steps that are taken within the system when a planning route is requested, and how THOMAS generates the system configuration that will give way to the plan. The first thing is to define the structural components of the organization, that is, the units that will be involved (which are initially empty), the system roles and norms. The indicated service requirements will be registered in the SF. To do so, either the basic OMS services for registering structural components will be used, or the API will directly execute the same functionality. This way, a community type *ShoppingUnit* will be created, representing the organization, whose purpose is to control the shopping mall. It has five internal unit planes: *ClientUnit*, *BusinessUnit*, *ManagingUnit*, *RecommentationUnit*, and *DeviceUnit*, each of which is dedicated to the functionalities we have previously seen. Each unit defines the existing roles, indicating their attributes (visibility, position, etc) and who they inherit them from.

The SF will announce basic services that are required for the overall system functionality. The basic services indicate which services are required (according to the defined norms) when creating the units. Some of these basic services are shown in Table II.

TABLE II
SF: BASIC SERVICES

Service Facilitator					
Entity	**Action**	**Service**	**ClientRole**	**ProvRole**	**Profile**
ClientUnit	Requires	manageConnection	ClientManager	Communicator	ClienteSP
DeviceUnit	Requires	locate	Communicator/IncidentManager	DeviceManager	DispositivoSP
…	…	…	…	…	…

The Service Facilitator will list the services that are needed for the functionality of the system. The basic services are those which are essential (as defined by the norms) when the units are being created. The SF will keep a registry of the services offered by each entity, the action taken by the service, the type of role that can request (client role) and offer (client provider) the service, and its profile.

From this moment on, the external agents can request the list of existing services and decide whether or not to enter and form part of the organization and with which roles. In our case we have clients that use their mobile device to send a request to the system so that it can inform them on the optimal route to take within the shopping mall. In order to carry out this function, we have, for example Co1, Pe1 and Pl1 acting as agents that will carry out the roles of *Communicator*, *ProfileManager* and *Planner* respectively. Agents C1 and C2 represent the clients that would like to receive a planning route.

Initially, all the agents head towards the THOMAS platform and are associated with the virtual "world" organization. As such, the OMS will play the member role in the "world" organization. When SF is asked about existing services in the system, the following response is obtained:

```
ClientUnit Requires manageConnection
ClientRole=ClientManager;ProvRole=Comunicator;
```

Because the service doesn't have an assigned grounding, it cannot be requested. But a functionality can be added, thus obtaining the *Communicator* role.

The Co1 agent wants to offer that functionality, for which it requests receiving the Communicator role for the *ClientUnit*:

```
AcquireRole(ClientUnit, Communicator)
```

If all goes well, the OMS will register Co1 in the role of *Communicator* in *ClientUnit* within the *Entity Play List*. This list shows the roles that the different agents assume within THOMAS.

The Co1 agent has carried out all of the regular steps for acquiring a role within THOMAS. This process is illustrated in Figure 3 where once Co1 has been registered as a member of the THOMAS platform, it asks SF which defined services have a profile similar to its own "communicator information service". This request is carried out using the SF SearchService (message 1), in which CommunicatorInformationServiceProfile corresponds to the profile of the *manageConnection* service implemented by Co1. The SF returns service identifiers that satisfy these search requirements together with a ranking value for each service (message 2). Ranking value indicates the degree of suitability between a service and a specified service purpose. Then Co1 executes GetProfile (message 3) in order to obtain detailed information about the *manageConnection* service. Service outputs are "service goal" and "profile" (message 4). The *manageConnection* profile specifies that service providers have to play a Communicator role within *ClientUnit*. Thus, Co1 requests the AcquireRole service from the OMS in order to acquire this provider role (message 5). AcquireRole service is carried out successfully (message 6), because *ClientUnit* is accessible from the virtual organization, thus Co1 is registered as a *Communicator*.

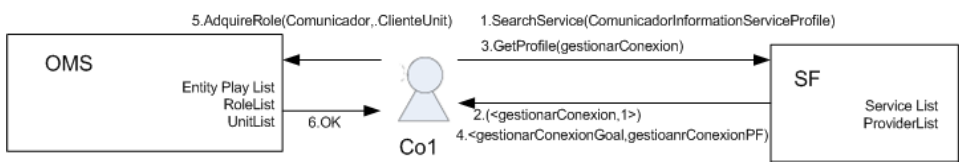


Fig. 3. Agent Co1 registering,

TABLE III
ENTITY PLAY LIST

Entity Play List		
Entity	**Unit**	**Rol**
Co1	ClientUnit	Comunicator
Co1	DeviceUnit	DeviceManager
Pl1	RecommendationUnit	Planner
C1	ManagingUnit	ClientManager
C2	ManagingUnit	ClientManager
C1	BussinesUnit	ProfileManager

Play List shows the roles adopted by agents within THOMAS

There will be another inquiry regarding which services exist within the units. *DirectionUnit*, *RecommendationUnit* and *DeviceUnit* will return the services that are necessary for planning. The SF will again return a list (similar to Table II).

Based on the profiles, we will determine that Co1 is interested in acquiring the role of *DeviceManager* since in this case it wants to interact with the elements within the environment. Co1 will use this role to act as intermediary to process the signals that come from the client devices and make them comprehensible to the system. It will allow the order requested by the client from a mobile device to be understood and executed by the specific device that is the object of the request.

```
AcquireRole(DeviceUnit, DeviceManager).
```

The agent will now be registered as a member of *DeviceUnit* with the role of *DeviceManager*. This role will require the agent to register the Locate service, associating it with the process and grounding that it considers to most useful. If this is not done within the allocated time, the agent will be expelled. The actual norm is as follows:

```
OBLIGED DeviceManager REGISTER Locate(?route)
BEFORE deadline SANCTION (OBLIGED OMS SERVE Expulse
(?agentID DeviceManager DeviceUnit))
```

The agent will be informed of the norm upon carrying out the “AcquireRole”, so that it can take it into consideration if it is a normative agent (otherwise ignore it). To avoid external agents assuming the role of *DeviceManager*, the agent registers a new incompatibility norm in the system. This norm will make it impossible for other agents to take on the same role:

```
RegisterNorm (”norm1”, ”FORBIDDEN Member REQUEST
AcquireRole Message(CONTENT(role‘DeviceManager))”)
```

The Pe1 and Pl1 agents will act in a similar fashion, registering at the end for the corresponding units *ProfileManager* and *Planner*. They too will be required to register the services as indicated by the defined norms. (*GenerateProfile*, *ConsultProfile*, *UpdateProfile*, *MSState*, *UpdateMSGState*, *Replan*, *ValidateRoute*, *ValueRoute*, *ShopListRecovery*) Each one is required for generating the optimal route for the user to follow. The C1 and C2 agents will request acquiring the *ClientManager* role in order to access the basic services: *FindClient*, *GenerateProfile*, *ConsultProfile*, *UpdateProfile*, *MSGState*, and *UpdateMSGState*.

The agents will also consider whether to acquire other system roles that might be necessary for the required functionality. C1 can request existing services from the SF, and will receive a list with all the agents that offer their services. In this case, for example, C1 could be interested in offering the *SendPromotion* service as a suggesting sent to the user. These services are offered from the *BusinessUnit*, for which it is necessary to acquire the role of ProfileManager

```
AcquireRole, (BusinessUnit, ProfileManager).
```

The Entity Play List would end up as shown in Table III.

IV. Conclusions

An important issue in the development of real open multi-agent systems is to provide developers with methods, tools and appropriate architectures which support all of the requirements of these kinds of systems. Traditional MAS development methodologies are not suitable for developing open MAS because they assume a fixed number of agents that are specified during the system analysis phase. It then becomes necessary to have an infrastructure that can use the concept of agent technology in the development process, and apply decomposition, abstraction and organization methods. We propose a methodology that incorporates decomposition and abstraction via the THOMAS architecture for a dynamic MAS environment. This architecture has allowed us to directly model the organization of a shopping center according to a previous basic analysis, to dynamically and openly define the agent roles, functionalities and restrictions, and to obtain beforehand the service management capabilities (discovery, directory, etc.) within the platform. THOMAS provides us with the level of abstraction necessary for the development of our system, and the set of tools that facilitate its development. In THOMAS architecture, agents can transparently offer and invoke services from other agents, virtual organizations or entities. Additionally, external entities can interact with agents through the use of the services offered. A case-study example was employed as an illustration of not only the usage of THOMAS components and services, but also of the dynamics of the applications to be developed with this architecture. In this way, examples of THOMAS service calls have been shown through several scenarios, along with the evolution of different dynamic virtual organizations.

References

[1] Argente, E., Julian,V., Botti,V.: “MAS Modelling based on Organizations“. *9th Int. Workshop on Agent Oriented Software Engineering*. pp.1-12.Citeseer:0.09 (2008)

[2] Bajo, J., Corchado, J.M., De Paz, Y., De Paz J.F., Rodríguez S., Martín Q. and Abraham, A.: “SHOMAS: Intelligent Guidance and Suggestions in Shopping Centres“. *Applied Soft Computing*. ISSN 1568-4946 (2008)

[3] Bernon, C., Cossentino, M., Pavón, J.: “Agent Oriented Software Engineering“ *The Knowledge Engineering Review*, vol. 20, nro. 2 (2005), 99-116

[4] Boissier, O., J.F.H., J. S.: “Organization Oriented Programming, from closed to open organization“s. *Engineering Societies in the Agents World VI*, Ireland: Springer.(2007)

[5] Carbó, J., Molina, J. M., Dávila, J.: “Fuzzy Referral based Cooperation in Social Networks of Agents“, *AI Communications*. vol 18, nº 1 pp 1-13, 2005.

[6] Carrascosa, C., Giret, A., Julian, V., Rebollo, M., Argente, E., Botti, V.: “Service Oriented MAS: An open architecture“. *Proceedings AAMAS 09*. In press (2009)

[7] GTI-IA. “An Abstract Architecture for Virtual Organizations“: The THOMAS project. http://www.fipa.org/docs/THOMASarchitecture.pdf

[8] Dignum, V., Dignum, F.: “A landscape of agent systems for the real world“. Technical report 44-cs-2006- 061, Institute of Information and Computing Sciences, Utrecht. (2006)

[9] Shekar, S., Nair, P., Helal, A.: “iGrocer- A Ubiquitous and Pervasive Smart Grocery Shopping System“, in: Proc. *ACM SAC*. 645-652. (2003)

Mobile Agents and Mobile Devices: Friendship or Difficult Relationship?

Oscar Urra, Sergio Ilarri, Raquel Trillo and Eduardo Mena

***Abstract*—Mobile agent technology has traditionally been recognized as a very useful approach to build applications for mobile computing and wireless environments. However, only a few studies report practical experiences with mobile agents in a mobile medium. This leads us to the following question: can current mobile agent platforms be used effectively in environments with mobile devices?**

In this paper, we study existing mobile agent platforms by analyzing if they are suitable or not in a mobile environment. We identify some key missing features in the platforms and highlight the requirements and challenges that lie ahead. With this work, we expose existing problems and hope to motivate further research in the area.

***Index Terms*—Mobile agents, mobile devices, mobile agent platforms, wireless and mobile environments.**

I. Introduction

Mobile agents [9] are programs that execute in contexts called *places*, hosted on computers, and can autonomously travel from *place* to *place* resuming their execution there. Thus, they are not bound to the computer where they were created and they can move freely among computers.

Mobile agents provide interesting features, thanks to their autonomy, adaptability, and capability to move to remote computers. Thus they can carry the computation wherever it is necessary, without the need of installing specialized servers there (only a generic *mobile agent platform* [18] is needed). In particular, the interest of mobile agent technology for wireless environments has been emphasized in the literature (e.g., see [16]). Thus, for example, instead of communicating a large amount of data from a computer to a mobile device, a mobile agent can move to the computer where the data are stored to process the data locally, filtering the non-relevant data that should not be communicated through the network. As another example, a mobile device could use a mobile agent to perform a processing-intensive task on a fixed computer with the required resources, relieving the overload of the mobile device. This will increase, in turn, its battery life, an important limitation on these devices.

However, and despite there are many research works that emphasize the advantages of mobile agent technology in the context of distributed systems, there are only a few reported experiences on the use of mobile agents in real wireless environments with mobile devices. In most cases, a simple and static wireless environment is considered (e.g., we performed an experimental evaluation in this context in [20]). Moreover, we also believe that it is important to study the challenges that mobile environments imply for a mobile agent platform. For example, in ad hoc networks multi-hop protocols may be required for an agent to travel to a certain node (as each node can only communicate with other nodes within its communication range). As another example, it becomes apparent that mobile agent platforms should provide services for agents to discover other nodes.

This work studies in depth the requirements and current limitations of mobile agent platforms to be usable in a mobile environment, extending the study presented in [21]. We analyze the challenges that need to be solved and highlight the advantages and disadvantages of existing platforms in a mobile environment. The structure of the rest of this paper is as follows. First, in Section II we overview the technological context of this work. In Section III, we introduce the concept of mobile agent platform and motivate the development of this work. In Section IV, we study security issues. In Section V, we consider the challenges of using wireless communications and mobile ad hoc networks. In Section VI, we focus on architectural elements that must be considered. In Section VII, we describe some agent-based system's development issues. In Section VIII, we present some sample application scenarios. Finally, in Section IX we summarize some conclusions.

II. Technological Context

Having mobile agents working on mobile devices involves a set of technological components that should provide the expected functionality when all of them work together. This section describes the most important aspects of wireless communications, mobile devices and operating systems for mobile devices, and mobile agent platforms.

A. Wireless Communications

Mobile devices communicate by using radio signals. There are many standards and protocols used for this purpose. We will focus on the two most commonly used today for local wireless technologies:

- *Bluetooth* [11] was designed for small devices, such as cell phones and PDAs, and it is normally used for the transmission of small amounts of data or to connect to nearby compatible peripherals (e.g., printers, keyboards, or hands-free headsets). It has a limited bandwidth (maximum 3 Mbps), a range of up to 100 meters, and low power consumption.

Oscar Urra, Sergio Ilarri, Raquel Trillo and Eduardo Mena are with the Department of Computer Science and Systems Engineering, University of Zaragoza
E-mail: ourra@ita.es, silarri@unizar.es, raqueltl@unizar.es, emena@unizar.es

- *Wi-Fi* [13] networks, based on the IEEE 802.11 standards, allow to expand traditional Ethernet local area networks to places where either cabling is not an option or mobility is desired or needed. Its popularity is growing very fast and almost every laptop computer made in the last years has a Wi-Fi interface which allows it to connect to an also increasing number of public access networks in places such as hotels, airports and restaurants. Compared to bluetooth, it has a higher bandwidth (54 Mbps), a similar range, and a higher power consumption and cost.

Both bluetooth and Wi-Fi are freely available and they are very popular. The main drawback of these short-range technologies when compared to traditional fixed Ethernet networks is that the latency and bandwidth are not only worse, but also variable depending on the distance and obstacles existing between the sender and the receiver. Moreover, even an established communication could suddenly end if one of the devices moves out of its communication range or enters a shadow area such as a tunnel or an underground room (disconnections are frequent).

B. Mobile Devices

Mobile devices could be classified in three basic types:

- *Cell phones*. They are small, light, cheap and with little computation capabilities. Data communications can be carried out through mobile phone networks or via bluetooth. Mobile phone networks have a variable bandwidth depending on the transmission technology (GSM, GPRS, UMTS, etc.) and are available almost everywhere, but using these networks has an economic cost.
- *PDAs or pocket computers*. They are bigger and more expensive than cell phones, but they also have better processing capabilities. There are many architectures (ARM, MIPS, Xscale, etc.) and several operating systems (see Section II-C) that allow the execution of end user applications similar to those available in desktop computers. Communications can be established through bluetooth, and more recently also via Wi-Fi.
- *Laptop computers*. They have capabilities comparable to those of desktop computers. They usually have both Ethernet and Wi-Fi interfaces, but it is also possible to use bluetooth for data interchange with small devices.

Currently, the three types of mobile devices mentioned above are starting to mix. Thus, cell phones and PDAs are converging into a single device, called *Smartphone* (PDA with a SIM card), being the Apple iPhone one of the most popular Smartphones nowadays. Similarly, laptops and PDAs are mixing into the so-called *Netbooks*, which have less computing capabilities than a conventional PC but also have a lower cost. Another advantage of Netbooks is that most of them (like the Asus Eee PC, or the Acer Aspire One) have the same x86 architecture than PCs and can use the same operating systems and applications.

C. Operating Systems on Mobile Devices

The operating system is the software that act as an abstraction layer between the hardware and the user applications, allowing the programmer to access the different components of the device in a uniform way. In the case of mobile devices, which have a number of special features, it is even more important that the operating system be able to manage the resources in an efficient and flexible way. For example, the duration of the battery (which is a critical factor), the user interface (the keyboard or touch screen), the size and occupation of the internal memory, etc., are examples of resources that need to be considered.

There exist many operating systems that can run on mobile devices, but the most extended ones are the following:

- *Symbian*. It was created for its use on mobile phones by a consortium of manufacturers such as Nokia, Motorola, Samsung or Sony-Ericsson. It is available only for the ARM architecture and is bundled with high-end phones produced by these manufacturers. It is proprietary but expected to become open source during 2009.
- *Windows Mobile*. It was developed by Microsoft and is used by many manufacturers (some of them using other systems too) such as Hewlett-Packard, Samsung, Motorola or Qtek. It is available for the ARM, MIPS and x86 architectures, and there exist two variants: *Windows Mobile PocketPC* for PDAs and *Windows Mobile Smartphone* for high-end mobile phones.
- *Android*. It is the most recent operating system that has appeared in the mobile market, developed by Google. It is derived from Linux and it is licensed also as open source. It is available for the ARM, MIPS and x86 architectures and can run on mobile phones, PDAs, and PC laptops. Due to its novelty, only a few devices use it nowadays, produced by manufacturers such as HTC or Samsung.

The operating system is a key element that should allow an efficient exploitation of the device's resources. Different features of an operating system, such as its scheduling policy, may have an important impact on the performance of a mobile agent platform running on the device.

D. Mobile Agent Platforms

There are many mobile agent platforms available, that differ in several aspects (such as their general architecture, communication style, etc.) and compare differently in terms of performance, reliability or scalability [18]. However, all of them offer similar services to their agents: execution, communication, mobility, tracking, directory, persistence, security, etc. We will briefly describe in this section the main services.

All platforms provide an *execution environment*, which is the most basic service and allows agents to run their code and access other services offered by the platform. The use of bytecode-based languages like Java eases the implementation of this service and its portability between different hardware architectures, and therefore most platforms are implemented in Java. Every device can execute one or more instances of these environments (known as *places* or *containers*), where several agents can be running simultaneously.

One of the strongest points of agents is their communication capability, so the existence of a *communication service* is also very important. When an agent starts a communication, it needs to determine the message to be transmitted and the destination where it must be sent (i.e., the target agent/s). The message must contain the information to transmit and must be intelligible for both the sender and the receiver. To achieve such mutual comprehension, there exist different *agent communication languages* (ACLs) that make the communication among different agents possible [4], [5]. Regarding the destination of the message, it can be a single agent or many of them, which can be located in the same execution environment as the sender or in a remote location. The communication service should provide agents with a common mechanism to build and deliver their messages to their destination, independently of where they are located (location transparency).

The *mobility service* allows agents to move to other execution environments. The process involves three steps. First, the agent determines the destination place and invokes the mobility service to be transferred. Then, its code and data are transferred across a network connection to the destination, where another running platform receives them. Once the transmission has finished without errors, the copy of the agent in the origin is destroyed and a new one is created in the destination from the code and data that compose the agent. This process is fail-proof: If there is any problem with the trip of the agent, the agent that attempted to travel will get the control back and decide what to do next.

The *object tracking service* keeps a record of the location of all the objects present in a multi-agent system [15], such as the agents themselves or the places/containers available in the distributed system. Whenever a new object is created, destroyed, or moved in the system, the service must be aware of such an action and update its location. As mentioned before, it is very important for the agent programmer that this service provide a truly *location transparency* so that, once an object's reference is obtained, it will be kept up-to-date by the system as long as necessary (the programmer will not need to refresh/update this reference).

Besides the services offered by the platform, agents can be programmed to offer different services to other agents or software components. Similarly, agents may also need to use services offered by other agents, to achieve their goals. The *directory service* allows the registration by agents of a description of the services they provide, and a common way to query, locate, and access the services included in the registry.

Thanks to all these services, mobile agents can *live* in a generic distributed environment and perform their tasks effectively. In the next section, we will examine some special difficulties introduced by mobile environments. A mobile agent platform should take this difficulties into account in order to be usable in such an environment.

III. Mobile Agent Platforms in Wireless Environments

Mobile agent technology has been proposed as a key element in the development of many applications, for a variety of reasons (e.g., their capability to exploit the locality of data by moving to the data source instead of interacting remotely using a network). With the increasing popularity of mobile devices, it was a natural step forward to try to apply mobile agents to the mobile environment. Given its portable nature, mobile devices use wireless communications, creating a scenario completely different from a traditional distributed environment with fixed networks. Such an environment has a number of advantages (e.g., the processing is not tied to a fixed location) but also some drawbacks, such as the limited computational power of mobile devices and a short communication range based on wireless technologies –that usually offer a low bandwidth, a high latency, and intermittent/unreliable connectivity–. Thanks to their features, mobile agents can be very useful in wireless applications (e.g., see [16]), as they could help to reduce the negative effects of such limitations. For example, as mentioned before, mobile agents can move to the place where the information is stored and process the data locally, discarding the data that is not relevant and therefore needs not be communicated through the wireless network. As another example, they can transport themselves and their data through networks following complicated and changing paths, by evaluating alternatives continuously based on information about their environment.

However, despite the increasing popularity of wireless services and the advantages of mobile agents in these environments, there are not many practical applications of this technology, except for some proofs of concept. A possible explanation could be that it is very difficult to develop and maintain such mobile agent based applications because existing platforms lack a number of features that should be present for their use in a wireless and mobile environment. Among them, we could emphasize:

- Features related to security, since the use of wireless communications broadcast data that could be intercepted or altered without having any notice.
- Features related to special network topologies, since there may be multiple mobile nodes with short range and unstable communications, which can make the process of transferring data between two nodes challenging.
- Features related to the way the platform itself works, since the mobile agents need different services, such as transportation or communication services.
- Features that should help the developer of agent-based systems, such as monitoring or debugging tools.

Providing the features listed above is important to develop applications based on mobile agents for mobile environments. Thus, for example, an agent should be able to detect the availability of new nearby devices (e.g., to travel to them) and to communicate with other agents easily and efficiently. Mobile agent platforms have usually been developed with a static context in mind and now they must be adapted to a more open and dynamic environment.

Besides, scalability and reliability are two key features which are important even when the mobile agents rely on fixed networks, and therefore even more critical in a challenging environment with wireless networks. Some tests indicate that the scalability/reliability of some platforms should be improved [3]. However, as this need is not tied to the wireless case –although heightened by it– we will focus on the other features; a summary of the analysis performed in this paper is presented in Table I. In the rest of the paper we will consider the following platforms: JADE, Voyager, and SPRINGS.

*J*ADE (http://jade.tilab.com), developed by Telecom Italia Lab since July 1998, was released as open source in February 2000 (last version: JADE 3.8.1, November 2008). It is a popular FIPA-compliant agent platform. An agent is composed of different concurrent (and non-preemptive) behaviors, which can be added dynamically. Among the benefits, we could indicate that there is a wide variety of tools provided (e.g., for remote management and monitoring of agents, and to track interchanged messages) and it can be integrated with different software such as Jess (a rule engine which allows agents to reason using knowledge provided in the form of declarative rules). Besides, it supports the development of ontologies to represent the knowledge of agents.

*V*oyager (http://www.recursionsw.com), developed initially by ObjectSpace in 1997 and currently by Recursion Software (last version: Voyager Edge 7.2.0, April 2009). It is a distributed computing middleware focused on simplifying the management of remote communications of traditional CORBA and RMI protocols. It is a commercial product, but there exists a *Community Edition* freely available valid during one year.

*S*PRINGS (http://sid.cps.unizar.es/SPRINGS/) [3], developed by the Distributed Information Systems Group at the University of Zaragoza in Spain, focuses on scalability and reliability in scenarios with a moderate and high number of mobile agents. Different features of other popular platforms, such as Voyager, have inspired its development.

IV. Security Issues

Security is always a major concern. Besides security problems with mobile agents in fixed networks [1], [22], other problems particular to wireless environments arise. Thus, it is necessary to ensure the privacy of the data and its integrity, and consider authentication and trust issues.

A. Communication Encryption

Wireless communications usually broadcast the transmitted data in an omnidirectional way. The disadvantage is that these data will be received not only by the intended destination but also by anyone within the range of the originating communication device. Although almost every wireless communication protocol (such as Wi-Fi or Bluetooth) can encrypt the data, it is not mandatory and in some circumstances an unencrypted communication can be the only form available (e.g., with public access points or *hotspots*). If a mobile agent is transferred using an insecure channel, its code and data will be exposed to any nearby device. To avoid this problem, the mobile agent platform should be able to encrypt all its communications when connecting to others.

Regarding the considered existing mobile agent platforms, both Voyager and JADE-LEAP can natively use SSL connections –which assure data flow encryption–, whereas SPRINGS currently lacks this feature.

B. Code Integrity

The code of a mobile agent can be altered, either intentionally or accidentally, in many ways. For example, in the first case, a malicious user could modify it while it is running on its device. In the second case, a failure such as a memory loss or a problem with the wireless transmission of the agent could lead to a corruption of its code. The execution of an agent whose code has been altered can lead to unpredictable consequences and should be avoided. Thus, the mobile agent platform should verify the agent's code integrity before starting its execution, for example signing the code with a X.509 digital certificate.

Thanks to the use of SSL by Voyager and JADE-LEAP, any attempt to tamper with an agent while it is being transmitted will be detected. Moreover, Voyager provides a signing mechanism to assure that the agent is not modified while it is stored in the device's memory. The current version of SPRINGS does not provide these features.

C. Authentication and Trust

A mobile device has a number of resources (such as the CPU, the memory, the file system, the user interface, etc.) susceptible to be used by an agent to accomplish its tasks. To avoid abuses on the use of these resources, the platform should state the extent to which it trusts an incoming agent –this is critical in an open environment–. Depending on it, the platform should allow, limit or even deny the mobile agent the access to the resources. There already exist mechanisms of authentication for distributed environments (e.g., Kerberos) that could be used to determine the owner of an agent. Once it is authenticated, different access control policies can determine the resources that the agent can use.

Voyager and JADE-LEAP have mechanisms to verify the identity of agents and other platform components, and grant different privilege levels. SPRINGS currently offers no authentication mechanisms but a basic access control.

V. Network Issues

Existing mobile agent platforms assume the existence of a TCP/IP network where every node can potentially connect to any other. This approach has the advantage of its simplicity, but it prevents considering explicitly other forms of communication that could be beneficial in a context with wireless mobile agents.

One limitation of mobile devices is that their communication interfaces (usually Wi-Fi or Bluetooth) have a relatively short range (a few hundred meters). Therefore, in a mobile ad hoc network a multi-hop routing protocol may be needed for two nodes to communicate. For a mobile agent platform, it would be interesting to access information about: whether the connection is fixed or wireless, the bandwidth available, if an established link (e.g., a Wi-Fi connection) is encrypted or not, the coverage level or the strength of the received signal in the case of a wireless communication, the identifiers of nearby wireless networks or nodes that could be contacted, etc.

With all this information, the platform would be able to take different useful decisions. For example, it could select the most appropriate communication link if there are several

TABLE I
COMPARISON OF MOBILE AGENT PLATFORMS: FEATURES FOR MOBILE ENVIRONMENTS

Feature		Voyager	JADE/LEAP	SPRINGS
Security Issues	Encryption	Yes (SSL)	Yes (SSL)	No
	Code integrity for transmissions	Yes (SSL)	Yes (SSL)	No
	Code integrity for execution	Some (signing)	No	No
	Authentication and trust	Yes (very rich)	Yes (rich)	Very basic
Network Issues	Access to link layer	No	No	No
	Adaptability	Latency measure	Conn. status	Conn. retrying
Architectural Issues	Discovery of nodes	No	No	No
	Transparent service discovery	No (YP)	No (DF)	No
	Adapted tracking approach	No (AgentSpaces)	No (AMS)	No (RNS)
Programming Issues	JVM supported	IBM J9	IBM J9	IBM J9
	Network communications	Sockets, HTTP	Sockets, HTTP	RMI
	Graphical interface	No	Yes	No
Strong points		Messaging, devices	Messaging, ontologies	Scalable, reliable

options available, in the case of a secure channel it would avoid encrypting the communicated data –reducing the CPU utilization and therefore saving battery power in the mobile device–, it could decide not to retry a failed communication if the coverage is poor, etc. Making this information accessible to the agents would also be interesting. For example, an agent could decide to jump to another device/node if its current device is getting out of coverage.

As far as we know, current mobile agent platforms do not have the ability to obtain this kind of information, which would be important in order to make the platform truly adaptive to different network environments. The closest related features that the considered platforms have are the following: Voyager can measure the network latency and detect that a communication is broken when an abnormally high value is obtained; JADE-LEAP can get basic information about the link status (connected or disconnected); finally, SPRINGS retries failed communications automatically according to some defined policy.

VI. ARCHITECTURAL ISSUES

In this section, we provide an overview of some architectural issues that should be considered in a mobile agent platform suitable for a mobile environment: discovery services and tracking and name services.

A. Automatic Discovery Service

As opposed to a fixed distributed infrastructure for mobile agents, a mobile context usually presents an open environment where many different computers/devices may appear and disappear at any time. Therefore, it is of paramount importance to provide agents with appropriate mechanisms to locate nodes to where they can travel. Moreover, the capabilities/services offered by these nodes should be advertised to allow the agents to decide a convenient target node.

1) Discovery of Services: There exist several service discovery protocols, such as the *Service Location Protocol* (*SLP*), the *Universal Plug and Play* (*UPnP*), or the use of the *Jini* technology. The suitability of these protocols for ambient intelligence is analyzed in [14].

Mobile agents should be provided an automatic discovery mechanism to allow them the detection of services of interest[1]. However, as far as we know, these protocols have not been integrated in any existing mobile agent platform. Moreover, it is not clear if they are the best choice in a mobile environment. For example, JADE-LEAP has a federated Yellow Pages (YP) service, but allocating it to a node is not transparent to the programmer and movements of the mobile device may invalidate the convenience of using that allocated YP service.

Finally, we would like to highlight the importance of providing a *semantic matching* of services [17], not only *syntactic matching*, if we want to enable dynamic and flexible interactions among the mobile agents. Thus, *Agent Communication Languages* (*ACLs*) [4] could play an important role. The language proposed by the *Foundation for Intelligent Physical Agents* (the *FIPA* ACL –see http://www.fipa.org/–) is the most popular proposal and it is supported by several mobile agent platforms (e.g., JADE). If services for mobile agents are implemented as web services, as suggested in [23], then existing techniques for web services (UDDI, WSDL, SOAP, OWL-S, etc.) could also be adopted.

2) Discovery of Nodes: It is also necessary for an agent to be able to detect the nodes that are reachable and which services/features provide those nodes. Otherwise, it will be very difficult for the agent to assess the convenience of traveling to another node. Thus, nodes must be auto-descriptive. This is particularly important in a heterogeneous environment where there will be computers/devices with different processing and communication capabilities.

However, no platform provides mechanisms to allow an agent to detect other potential target computers/devices. Although some platforms provide name services to query the computers that can host an agent (e.g., the Region Name Server in SPRINGS [3]), they do not consider that some

[1]Some works propose using mobile agents to implement service discovery [8], [12].

computers/devices may not be accessible from a given location (e.g., if the mobile agent is executing on a device that can only communicate with other devices within its communication range); indeed, the name service itself may be unreachable.

B. Tracking of Mobile Agents and Name Services

As mobile agents move from one computer/device to another, tracking their locations efficiently (which is required if we want to be able to allow communications with those agents from any computer/device) is challenging. To solve this problem, different approaches have been considered. Some mobile agent platforms provide a naming service that can be used to locate an agent and then send a message to its address. An alternative (and also complementary) approach supports communications by using *proxies* (similar to the *stubs* in *RMI*) as a convenient abstraction to refer to remote agents (e.g., to send them messages or call methods remotely). Several platforms, such as SPRINGS and Voyager, support proxies. Related to the concept of proxies, *dynamic proxies* remain valid independently of the agents' migrations (i.e., the reference is updated automatically) [3].

However, existing strategies have been developed with a fixed network in mind and are not appropriate in a mobile environment. For example, SPRINGS assumes the existence of stable *Region Name Servers* (*RNSs*) and *location servers* with tracking responsibilities. This could be unsuitable in a dynamic context because nodes can leave/enter the network at any time and some nodes could be temporarily unreachable. Similarly, according to [10], Voyager uses forwarding chains of proxies, which may be inconvenient because any link (pointer) in the chain could disappear at any time. Thus, we believe that new tracking techniques are needed in this context, avoiding mechanisms that rely on the availability of certain nodes or centralized approaches, in favor of adaptive tracking approaches.

Finally, we should mention that ensuring the uniqueness of agent names and at the same time providing user-friendly mechanisms to address the agents is a challenge in an open and dynamic environment. A potential preliminary solution would imply a shift in the way agents communicate. At present, it is usually assumed that agents identify their partners by name. An alternative would be to identify partners by service, which would make user-friendly agent names unnecessary.

VII. Issues Related to the Development of Systems Based on Mobile Agents

The implementation of mobile agent applications for their execution in mobile devices can be problematic due to a number of issues, such as the availability of Java interpreters, the efficiency of different types of networking protocols for communication, or the existence of tools for the programmer.

A. Java Virtual Machines for Mobile Devices

To guarantee portability across heterogeneous devices and computers, Sun Microsystems created different distributions of the Java platform. From the point of view of this paper, it is interesting to distinguish the following possibilities:

- *Java SE* (*Java Standard Edition*). It is the most widespread edition, used by personal computers and servers running general purpose applications. There are many implementations and *Java Virtual Machines* (*JVMs*) available, usually for free, like the official implementation by Sun Microsystems for PC and SPARC architectures, as well as those offered by other companies such as IBM or HP.
- *Java ME* (*Java Micro Edition*). It is designed for small devices and there are two specifications, one intended for more limited equipment such as mobile phones (*Connected Limited Device Configuration -CLDC-*) and the other, more complete, for PDAs (*Connected Device Configuration -CDC-*).

Sun Microsystems does not develop JVMs for mobile devices, but they are made by third parties. In mobile phones, the manufacturers (e.g., Nokia, Motorola, Sony-Ericsson) include the runtime as an indivisible part of its device. This also happened with PDAs in some cases (e.g., with PDAs by Compaq), but now almost none brings any JVM installed, and so it must be acquired separately. JVMs for mobile devices are neither easy to find nor free, in contrast to the standard Sun's JVM. Among the virtual machines for PDA-like mobile devices, we can highlight the following:

- *Jeode*. Some years ago, it was distributed with PDAs by Compaq. Then, there was the fusion with HP, and the new models did not bring it. The company which originally made it was acquired by another one which no longer continued its development.
- *Cr-EME*[2]. It is a high quality J2ME-compliant commercial machine oriented to the market of embedded devices, and has some optional components such as RMI (see Section VII-B) or AWT, which are sold separately. Its biggest problem is its distribution, as it is only sold in lots of 40 units at 1000 dollars (as of July 2009).
- *IBM Websphere J9*[3]. Another commercial JVM. It is part of the Websphere development suite. It is also a very good option and also has additional components such as RMI and AWT. The price of the runtime (without the development environment) is 25 dollars (as of July 2009), which makes it quite affordable.
- *MySaifu*[4]. It is an open source implementation under development. It is capable of using some features of AWT components and user interfaces. However, it is still at a very early stage regarding networking aspects (e.g., it does not offer support for RMI).

Regarding the considered mobile agent platforms, all of them (SPRINGS, Voyager and LEAP) require the IBM Websphere J9 Java machine when they are executed on mobile devices like PDAs. Voyager, additionally, has a version using the *.NET Compact Framework*, a light version of the .NET by Microsoft.

[2] http://www.nsicom.com

[3] http://www-306.ibm.com/software/wireless/weme/features.html?S_CMP=wspace

[4] http://www2s.biglobe.ne.jp/~dat/java/project/jvm/index_en.html

B. Networking communication protocols

There exist different communication protocols that can be used for agent communication. The choice of one or another can be a difficult issue.

On the one hand, using low-level communication protocols gives a higher performance, but it is more difficult for the programmer. On the other hand, using a high-level protocol is easier, but less efficient and it could be unavailable on a mobile device with constrained capabilities.

Some popular network communication protocols are:

- *Plain sockets*. This is the most low-level communication form that can be used. It is standard, efficient, and it is possible to explicitly choose if they must be connection-oriented (TCP) or not connection-oriented (UDP). They can be used in almost any programming language but the drawback is that any higher-level operation must be programmed using many low-level functions, which is a very tedious task for the programmer and prone to fail.
- *RMI*. The *Remote Method Invocation* is the mechanism provided by Java to invoke functions on remote (or local) objects. The invocation can carry information through the use of input parameters, that can be as simple as integers or characters or as complex as whole objects. In some JVMs for mobile devices, it is an optional component and in others it is not available at all. Although RMI is a Java-only communication form, it fits well with the paradigm of mobile agent because most mobile agent platforms are implemented in Java.
- *HTTP*. The *Hyper-Text Transfer Protocol* used by the Internet browsers have many advantages: It is a standard and ubiquitous. It is also bundled with many programming languages and libraries, and can be easily programmed and debugged (e.g., using a browser). Additionally it can travel through proxies, giving the possibility of traversing firewalls, which are usually open for this protocol but closed for any other. A drawback of HTTP is that it can be inefficient to transfer binary data because in such case the data must be encoded (usually with MIME). Even though there are some proposals to implement mobile agent platforms using the HTTP protocol (e.g., see [7]), currently no popular mobile agent platform is implemented on top of it.

Regarding the considered mobile agent platforms, SPRINGS uses RMI, and both JADE/LEAP and Voyager use plain sockets for intra-platform communication and HTTP for extra-platform communication.

C. Programming and Debugging Tools

To ease the development of distributed applications using agent technology, a number of tools and utilities can be used by the programmer. These tools are not very different from those used for the development of any other software, but not all of them are so available or popular among existing mobile agent platforms. Some examples of interesting features for the developer follow:

- *Programming API and documentation*. In order to develop applications using mobile agents the programmer must be able to build the structure of agents, but he/she must also know the actions they can perform (i.e., move to another execution place, send a message to another agent, query a published service, etc.) and how to program the agent to perform these operations. The mobile agent platform's API allows the programmer to build agents and use from the most basic to the more complex features of the platform. This API should be complete enough to allow the programmer to use every platform capability, and have extensive documentation explaining its use, to ease the development and maintenance of such agent-based applications.
  Regarding the considered mobile agent platforms, all of them have a well-documented API, as well as additional documents that cover other interesting platform issues.
- *Deployment and management graphical tools*. A mobile agent platform can be a very complex software, with many components and configuration parameters. The process of setting up a scenario for the execution of a complex distributed application can be a tedious task, especially if the application development is in progress. The use of a graphical user interface can make more pleasant the task of setting up and deploying initially all the components of a multiagent system. Once it is running and all their components executing as expected, such a tool could also be useful for managing the whole environment, allowing to add or remove components at any time in an easy way.
  JADE/LEAP is the only platform that provides a graphical tool that allows to perform basic operations like creating places and agents, move agents from one place to another, and many others.
- *Logging capability*. During the development of any application, errors and bugs may appear in the software as a natural part of the process. Additionally, in a distributed scenario many components can be physically out of the scope of the programmers (i.e., in remote locations) so it can be difficult to know what is going on at every moment in such components. In a mobile agent platform these components can be agents, service directories, places, etc. It is very important that the different platform components have the ability to log all the relevant information (such as their status, executed actions, etc.) and store such data in a way that can be retrieved later by the programmer for their analysis.
  All the considered platforms give extensive information when some components (such as places or registries) are launched, that can be stored in plain text files. In general, the information provided consists of: the name and version of the platform, the name of the object, their correct or incorrect initialization, and important events and errors (service registration, the creation of an agent, etc.)
- *Debugging and monitoring capability*. Similarly to the logging capability, debugging/monitoring is also a desirable feature for the programmers. The difference with logging is that debugging/monitoring is performed *online*, so that the programmer can follow the status (data, ac-

tions, communications) of an object (agent, place, service, etc.) while it is being executed. Moreover, it is also very useful if the execution can be paused and analyzed, or even if it is possible to modify the contents of program variables or communications during the execution.

Neither Voyager nor SPRINGS provide natively this kind of tools, whereas JADE/LEAP has some debugging and monitoring functions in its graphical interface. Thanks to it, it is possible to follow agents as they move among places, stop their executions, send them customized messages, and even *sniff* and alter the messages the agents exchange. It is also worth mentioning the existence of independent monitoring tools for agent-based systems, such as the 3D monitoring tool presented in [2].

As a summary, in general the considered platforms provide enough tools to the programmer for developing agent-based applications properly although some advanced and handy tools are missing. One notable exception is the graphical interface of JADE, that allows the control and debugging of this platform in a very intuitive way.

The programming tools that we have described in this section are important in any scenario where a system based on mobile agents must be deployed. However, it is obviously in distributed scenarios, and particularly in mobile computing scenarios, where the programmer needs more support from the platform. Particularly, debugging and monitoring capabilities are very important in these contexts, where the number of potential sources of failure increases

VIII. Sample Application Scenarios

In this section, we will show two sample application scenarios that can benefit from mobile agents: The first is a distributed monitoring system using moving cars in a wide geographic area. The second is a simple system for searching documents on a set of personal PDAs.

A. Mobile Agent Technology on Vehicular Networks

A vehicular network (or VANET) is a mobile network whose nodes are cars moving across the roads and highways of a given geographic area. We think that mobile agents could provide interesting benefits in this scenario. As an example, we will summarize in the following our proposal to use mobile agents in cars for monitoring different environmental parameters without needing a fixed and costly infrastructure [19].

In this sample scenario, the cars are equipped with different types of sensors that measure the desired parameter as the cars move inside a designated area of interest or *Monitored Area*. The data collected by these sensors are carried by mobile agents hoping from car to car until they reach a *Monitoring Center* for later processing. In this way, many monitoring tasks can be performed simultaneously, covering a large area and without needing a fixed infrastructure (except for the Monitoring Center), although the mobile agents could also take advantage of already-existing fixed elements like roadside antennas or information panels. To perform the monitoring, all the cars participating in the monitoring VANET must have installed some technological elements such as: one or more sensors for reading the environment data, a wireless communication device (e.g., Wi-Fi or Ultrawide Band), a GPS receiver for knowing the position with respect to the monitored area, and a PDA or laptop capable of executing a mobile agent platform and accessing the rest of the elements. The whole monitoring process consists of five steps (see Figure 1):

1) The monitoring scope is defined. This implies setting the environmental parameters to measure, the location of the monitored area, and the amount of time during which the monitoring must be performed.
2) In the Monitoring Center a number of monitoring agents are created and they are transmitted (they *jump*) to moving cars (it is supposed that this Center has an antenna or is near the road where the monitoring cars are constantly moving).
3) The monitoring agents travel on the car towards the monitored area. At any moment, they can jump to any nearby car if it follows a more promising path towards the target area. To know this, the agents can read the GPS information and also ask other nearby agents about their position and heading.
4) Agents in the monitored area read data from their cars' sensors. They can also jump to other cars and clone themselves to stay within the area and increase the number of samples taken, thus improving the monitoring.
5) When the planned monitoring time expires, the agents must return the measured data to the Monitoring Center for later processing. To achieve this, they follow the same process that they followed to reach the monitored area, jumping from one car to another based on their position and other factors.

This application scenario shows that mobile agents in VANETs can be very useful for monitoring purposes. Moreover, we can think of other contexts where VANETs and mobile agents could fit together, for example to exchange information of the road status some kilometers ahead, to disseminate information in an area about traffic jams, or even to answer queries launched dynamically by drivers (e.g., how many parking sites are in a certain street).

However, using mobile agents in this highly dynamic mobile P2P environment also poses a number of interesting challenges due to the high mobility of the network nodes. Some of these challenges are closely related to those exposed in the previous sections: discovering services in a decentralized way, travelling from one place to another using a multi-hop protocol when nodes are constantly moving, or assuring that data is not altered during transmissions to avoid the dissemination of false information. We plan to extend our work on this area in order to study in detail the benefits and difficulties implied by the use of mobile agents in vehicular networks.

B. Searching Relevant Documents in a Set of PDAs

We may envision a situation where members of a research team in a university store different types of research documents (papers, technical reports, personal notes, etc.) on a wide range of mobile devices to be able to access these documents while they are moving. Besides the importance to keep synchronized

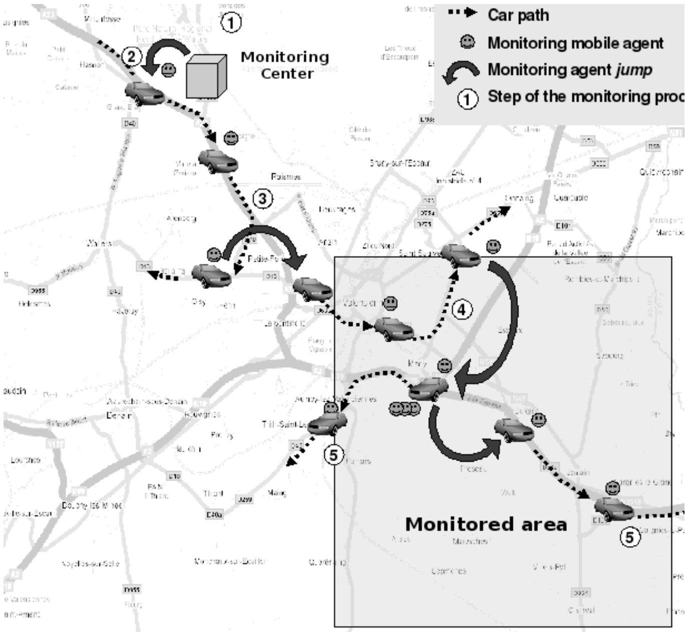


Fig. 1. Using monitoring agents for distributed measurement

the different repositories managed by a single person, it is also interesting to allow him/her to search for relevant documents that may be stored by other member of his/her team. We advocate the use of mobile agents for efficient searching and filtering of relevant documents. In this section, we will show an approach based on mobile agents and compare it experimentally with a traditional client/server approach. From any of these devices a search can be launched, in order to retrieve the most relevant files and transferring them wirelessly to a local directory. The purpose of this experiment is to verify that a distributed search can be more efficient by using mobile agents than using a traditional client-server method.

To start the searching process, the user launches an application in his/her own PDA and enters some keywords (a *search string*) to be searched in the documents. A simple algorithm is used to calculate the relevance of a file for a given search string. We will evaluate two alternative approaches to find relevant documents using that algorithm:

- *Using a client-server approach*. A file server runs on every device containing the files. The search is started from a client that connects to every other device, downloads its files and finally performs the search locally by analyzing all the documents downloaded from all the devices.
- *Using mobile agents*. Every device runs the SPRINGS mobile agents platform and one of them (the client) starts the search. A mobile agent is created and it moves to the other devices, one after another. In each of them, it makes a local search and the relevant files are sent to the origin by another mobile agent created then for that purpose.

The devices used for this test are one laptop and two PDAs. Bluetooth is used for wireless communication, acting the laptop as an intermediate access point for the PDAs. The configuration of the elements is as follows:

- The laptop computer is an Intel Centrino with two 1.66 GHz cores and 2 GB of RAM, running Linux with kernel 2.6.24. It acts as a bluetooth access point for wireless communications. The Sun JVM 1.4 is used, since in this way the same compiled code can run seamlessly in both the PC and the PDAs Java interpreters.
- A PDA Fujitsu-Siemens Loox 720, with a X-Scale processor at 520 MHz, and 128 MB of RAM.
- A PDA HP iPAQ 1940, with a Samsung S3C processor at 266 MHz, and 64 MB of RAM.
- Both PDAs run Windows Mobile 2003 and include an integrated bluetooth adapter (class 2, 10-meter range). The JVM used on the PDAs, in both cases, is the IBM Websphere J9 version 11 for Windows Mobile.

In the experiment, we measure the time that a search takes to complete, depending on the relevance of the search string in the available documents (e.g., a 20% relevance means that 20 out of 100 documents are relevant for the given search string). The parameters used for evaluation are as follows:

- The search is launched from the Fujitsu PDA.
- In the test using mobile agents, the SPRINGS' RNS component runs on the PC, as this is the most static of the three devices involved.
- In every device there are 10 files of 50 KB, adding up to a total of 500 KB.
- We vary the relevance between 0% and 100% in 20% steps, and for every value we repeat each test five times. Average values are reported in the experimental results.
- In every test, we measure the amount of time needed since the beginning of the search process until all the results are retrieved.

The results can be seen in Figure 2. The time needed to complete a search with the traditional client-server method is constant, regardless of the relevance of the search string. This is because, with the client/server approach, all the documents are communicated to the searching device independently of whether they are relevant or not for the search string considered. However, the amount of time needed with the approach based on mobile agents depends of the relevance (for the search string considered) of the different documents stored on the devices. As the figure shows, the use of mobile agents minimizes the amount of data transferred and leads to smaller search delays. Only if most of the documents are relevant, which implies that the local filtering performed by the agents is very limited, the client/server approach performs similarly.

There are two extreme points in the graph, for relevance values of 0% and 100%. In the first case the searching process takes ten seconds before the result (none) is obtained. This is due to the time it takes the agent to be dynamically created and because once it starts moving it makes all its journey through the PDAs sequentially, one site after another. In the 100%

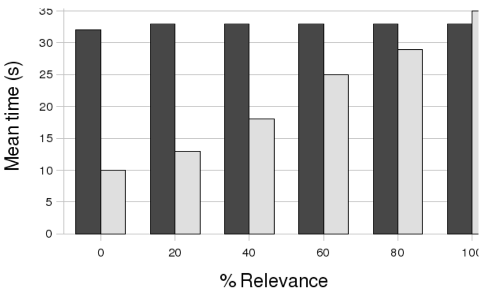


Fig. 2. Times searching with and without agents

case, the method using mobile agents is slightly worse than the traditional client-server approach, for the same reasons (extra overhead of mobile agents with no filtering benefit).

From this experiment we conclude that mobile agents can be used for efficient searching, as they process data locally on every data container, transferring only the relevant data. In this way, the workload is shared among all the devices and the bandwidth usage is minimized, making a better use of both resources, which are so limited on mobile devices.

An enhancement would be to use a more decentralized scenario, since in this case the PC is a central node containing the RNS and acting as a bluetooth access point. In a truly decentralized scenario the PDAs would communicate among them directly (without needing the PC) in a P2P way. Unfortunately, SPRINGS is not yet ready to use such decentralized solutions, and neither Voyager or JADE/LEAP. Another enhancement would be to use a higher number of PDAs and create many copies of the searching agent, that would execute their tasks in parallel. With these enhancements, the searching and filtering efficiency would increase.

In general, mobile agents can be useful in scenarios where we need to collect/disseminate some information from/to a set of mobile devices. For example, a similar scenario to the one presented here could be considered for schedule planning [6].

IX. Conclusions

Mobile agent technology has been highlighted as a very interesting approach to build applications for mobile environments. However, it is hard to find practical applications with real prototypes and using the available mobile agent platforms. One reason is probably that such platforms have been developed with a fixed distributed environment in mind, and not considering the features that may be of special interest in a mobile environment (e.g., reliance against security threats, adaptation to the network technology, and service/node discovery). Considering these features would make the adoption of the technology much easier.

In this paper, we have identified the requirements and desired features of mobile agent platforms to be used in scenarios with mobile devices. With these requirements in mind, we have analyzed the missing features in some popular mobile agent platforms. As future work, we plan to study these issues in detail and propose solutions for SPRINGS [3]. We hope that future research and development efforts will eventually lead to consolidate a good relationship between mobile agents and mobile devices.

Acknowledgment

This work was supported by the CICYT project TIN2007-68091-C02-02

References

[1] M. Greenberg, J. Byington, and D. Harper. Mobile agents and security. *IEEE Communications Magazine*, 36(7):76–85, 1998.

[2] S. Ilarri, J. Serrano, E. Mena, and R. Trillo. 3d monitoring of distributed multiagent systems. In *Third International Conference on Web Information Systems and Technologies (WEBIST'07)*, pages 439–442. INSTICC Press, 2007.

[3] S. Ilarri, R. Trillo, and E. Mena. SPRINGS: A scalable platform for highly mobile agents in distributed computing environments. In *4th Intl. WoWMoM 2006 Workshop on Mobile Distributed Computing (MDC'06)*, pages 633–637. IEEE, 2006.

[4] M. T. Kone, A. Shimazu, and T. Nakajima. The state of the art in agent communication languages. *Knowledge and Information Systems*, 2(3):259–284, 2000.

[5] Y. Labrou. Standardizing agent communication. *Lecture Notes in Computer Science*, pages 74–97, 2001.

[6] S. Macho, M. Torrens, and B. Faltings. A multi-agent recommender system for planning meetings. In *Fourth International Conference on Autonomous Agents, Workshop on Agent-based Recommender Systems (WARS2000)*, 2000.

[7] P. Marques, R. Fonseca, P. Simões, L. Silva, and J. Silva. A Component-Based Approach for Integrating Mobile Agents Into the Existing Web Infrastructure. In *Proceedings of the 2002 Symposium on Applications and the Internet (SAINT'02)*, pages 100–108. IEEE, 2002.

[8] R. T. Meier, J. Dunkel, Y. Kakuda, and T. Ohta. Mobile agents for service discovery in ad hoc networks. In *22nd Intl. Conf. on Advanced Information Networking and Applications (AINA'08)*, pages 114–121. IEEE, 2008.

[9] D. Milojicic, F. Douglis, and R. Wheeler. *Mobility: processes, computers, and agents*. ACM, 1999.

[10] L. Moreau. A fault-tolerant directory service for mobile agents based on forwarding pointers. In *Symp. on Applied Computing (SAC'02)*, pages 93–100. ACM, 2002.

[11] R. Morrow. *Bluetooth: Operation and Use*. McGraw-Hill Professional, 2002.

[12] N. Nehra, R. B. Patel, and V. K. Bhat. MASD: Mobile agent based service discovery in ad hoc networks. In *14th Intl. Conf. on High Performance Computing (HiPC'07)*, volume 4873 of *Lecture Notes in Computer Science*, pages 612–624. Springer, 2007.

[13] F. Ohrtman and K. Roeder. *Wi-Fi Handbook : Building 802.11b Wireless Networks*. McGraw-Hill Professional, 2003.

[14] D. Preuveneers and Y. Berbers. Suitability of existing service discovery protocols for mobile users in an ambient intelligence environent. In *Intl. Conf. on Pervasive Computing and Communications*, pages 760–764. CSREA Press, 2004.

[15] V. Roth and J. Peters. A scalable and secure global tracking service for mobile agents. *Lecture Notes in Computer Science*, pages 169–181, 2001.

[16] C. Spyrou, G. Samaras, E. Pitoura, and P. Evripidou. Mobile agents for wireless computing: the convergence of wireless computational models with mobile-agent technologies. *Mobile Networks and Applications*, 9(5):517–528, 2004.

[17] K. P. Sycara, S. Widoff, M. Klusch, and J. Lu. Larks: Dynamic matchmaking among heterogeneous software agents in cyberspace. *Autonomous Agents and Multi-Agent Systems*, 5(2):173–203, 2002.

[18] R. Trillo, S. Ilarri, and E. Mena. Comparison and performance evaluation of mobile agent platforms. In *3rd Intl. Conf. on Autonomic and Autonomous Systems (ICAS'07)*, page 41. IEEE, 2007.

[19] O. Urra, S. Ilarri, T. Delot, and E. Mena. Using hitchhiker mobile agents for environment monitoring. In *7th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS'09)*, pages 557–566. Springer, 2009.

[20] O. Urra, S. Ilarri, and E. Mena. Testing mobile agent platforms over the air. In *1st ICDE Workshop on Data and Services Management in Mobile Environments (DS2ME'08)*, pages 152–159. IEEE, 2008.

[21] O. Urra, S. Ilarri, and E. Mena. Agents Jumping in the Air: Dream or Reality? In *10th International Work-Conference on Artificial Neural Networks (IWANN'09), Special Session on Practical Applications of Agents and Multi-Agent Systems*, pages 627–634. Springer, 2009.

[22] G. Vigna, editor. *Mobile Agents and Security*, volume 1419 of *Lecture Notes in Computer Science*. Springer, 1999.

[23] J. Zhang, Y. Wang, and V. Varadharajan. A new security scheme for integration of mobile agents and web services. In *2nd Intl. Conf. on Internet and Web Applications and Services (ICIW'07)*, page 43. IEEE, 2007.

Ubik: a multi-agent based simulator for ubiquitous computing applications

Emilio Serrano, Juan A. Botia and Jose M. Cadenas

Abstract—**This paper introduces the development of an infrastructure to study highly complex systems of Ambient Intelligence (AmI) which involve a large number of users. The key ideas about the development of a multi-agent based simulation (MABS) for such purposes, Ubik, are given. The paper also extrapolates effective technologies in the development of multi-agent systems (MAS) to the field of MABS. In particular, the basis for the use of forensic analysis as a method to assist the analysis, understanding and debugging of Ubik in particular and the general MABS are set up.**

Index Terms—**Ambient Intelligence, Multi-agent based simulation, Multi-agent systems, forensic analysis**

I. INTRODUCTION

Ambient Intelligence (AmI) stems from the convergence of ubiquitous computing, ubiquitous communication and intelligent user-friendly interfaces and creates environments that are characterized by their ubiquity, transparency and intelligence[6]. AmI is of great interest to society because in a near future it can help us to do our work, improve our health, do house chores, etc. The research on AmI presents new scientific challenges. Traditional usability engineering methods and tools fail in the development of AmI applications[3]. Established organizations such as the Usability Professional's Organization have recognized the need for new approaches to usability testing[3]. When AmI is applied to a large number of users, there is a point where the real tests are not feasible. This paper tries to provide solutions to these situations.

Multi-agent based simulation, MABS, allow modelers to handle different levels of representation (e.g., "individuals" and "groups", for instance) within an unified conceptual framework. Such versatility makes MABS one of the most favorite and interesting support for the simulation of complex systems [4]. MABS is used in more and more scientific domains [4]: sociology, biology, physics, chemistry, ecology, economy, etc. This paper proposes the use of MABS in AmI. Specifically, complex AmI applications with a large numbers of users are treated. Testing the social behavior of user groups is interesting in these applications. That is, we are interested in the macro-social perspective. Therefore, the overall objective of our research is to increase the usability of this type of complex AmI applications.

About the field of MABS, generally, researchers are more interested in model design than in model execution and the execution analysis. Specifically, the analysis is viewed as less methodical and hand crafted[4]. The field of Multi Agent Systems (MAS), a well-established branch of AI, is complementary in several aspects to MABS[2]. MAS theory has scientific approaches to analyze, understand and debug the social level of agents. This paper aims to argue that technologies used in the analysis of MAS can be used for MABS. In particular, *forensic analysis* is proposed. This technology has already been used successfully to debug MAS in a social level[13]. Forensic analysis is the process of understanding, re-creating, and analyzing arbitrary events that have occurred previously[11].

To achieve the overall objective, increase of the usability of complex AmI applications, four objectives are marked: (1) Developing of an infrastructure to study complex AmI applications. The multi-agent based simulation Ubik achieves this goal. (2) Testing the effectiveness of AmI applications with the infrastructure resulting from the accomplishment of the first goal. This is achieved by extending the Ubik simulation. (3) Development of an infrastructure to assist the analysis, understanding and debugging of group behaviors in general MABS through forensic analysis. (4) Use of the results of the third goal for the simulations resulting in the first two objectives. What we aim is to validate the simulated models, understand them, explain them and, ultimately, achieve the overall objective.

The next section deals with the related work. Then, section III studies in detail the infrastructure for complex AmI applications and the sort of AmI applications which can be studied. Section IV treats the infrastructure for the debugging of MABS. Finally conclusions and future work are given.

II. RELATED WORKS

A *Living Lab* is a laboratory which consists of real life environments where real users together with researchers look together for new services in AmI. In 2000 started first serious plans to build a laboratory that could be used to conduct feasibility and usability studies in AmI[6]. Finally, the HomeLab of Philips was opened on April 24, 2002. Other well-known living labs are AwareHome Project [7] or Adaptative House[10]. This approach of direct testing in laboratories has proliferated. For example, the *European Network of Living Labs* is a grown up initiative coming from the own European Living Lab and sponsored by the European Community. On the open living labs web[1] can be found contact information for living labs in 29 countries, only in Spain 19 living labs are joined with this project. However, these approaches are not feasible in applications for thousands of users or more. Research into ubiquitous computing has also used simulations

Emilio Serrano, Juan A. Botia and Jose M. Cadenas are with Department of Information and Communications Engineering at the University of Murcia, Spain.
E-mails: {emilioserra, juanbot, jcadenas}@um.es

[1]Open living labs website: http://www.openlivinglabs.eu/

such as TATUS [5] and Ubiwise [1] which extend first person shooter games (FPS). However, these simulations, which model the physical environment in a 3D view, do not model the user which is a real player of the game. Besides, they are valid for only a few users and not for thousands of them. Hence, this paper proposes the use of MABS.

Luke et al. [9] introduce the basis of the platform we use to program the simulation Ubik, MASON[2], as well as several simulations using it. For example, the simulation *Anthrax Propagation in the Human Body* has in common with Ubik that treats a field with which you can not experiment with in a laboratory. This simulation, like ours, is useful to help researchers in the field to explore what-if scenarios and to create strategies to attend humans effectively. The MABS have also been used to check the effectiveness of networking applications as we intend. In this way, *Network Intrusion* is an agent-based model designed to study computer network security[9]. The parameters of the model allow researchers to understand the effects of changes in security policies, just as Ubik does with the AmI applications policies. In general, any work on MABS offers interesting points of our research; the innovation of this paper is the use for AmI applications.

Simulation typically generates huge amounts of data. Analysis of the simulations have not received the attention it deserves. Within MAS theory, we can find literature that deals with the analysis of agents' behavior. In this way, Serrano et al. [13] detail the creation of an infrastructure for forensic analysis for MAS. Forensic analysis is often the basis of most proposals about debugging in MAS and it is often accompanied by comprehensible data representations. There are works that analyze the behavior of the agents groups with *petri nets*, *AUML diagrams*, extensions of the *propositional dynamic logic*, *statecharts*, *dooley graphs*, and so on. There are also works that combine forensic analysis and data mining to achieve simple representations of the agent society[14]. Each of these methods to analyze and debug the MAS are directly usable in this field of MABS.

III. UBIK, A MULTI-AGENT BASED SIMULATION FOR THE AMI DOMAIN

Any AmI facility involves a financial outlay, much larger when AmI is applied in large domains as a skyscraper or even a whole city. For example, we do not know the cost of the HomeLab of Philips [3], a laboratory to study AmI applications, but we are quite sure that is much greater than that needed for the development of any MABS. Simulation analysis helps to verify the effectiveness of AmI applications before installing them. It also allows to check the results of the AmI in diverse and configurable societies, which make these applications more robust. Anyway, the most important application of the presented proposal is using it in AmI applications with so many users that real tests are totally unworkable. For example, HomeLab consist of a house with a living, a kitchen, two bedrooms and a study where these applications cannot be performed.

Ubik[3] is an infrastructure to study complex AmI applications which involve a large number of users. It is a MABS programmed in MASON and aims to be as descriptive as possible to be useful for AmI. A longer description implies a more complex process of understanding things of interests for us to explore new technologies to assist the analysis of simulations (section IV). Figure 1 shows Ubik between MASON and AmI applications.

As it will be explained below, figure 1 also shows this proposal is useful using other MABS a part of Ubik. Different MABS frameworks can be used a part of MASON too. And obviously, the assistance in emergencies is not the only interesting AmI application to develop with the presented proposal. Finally, figure 1 shows an Analyzer layer which provides useful resources for the developer in charge of studying the simulations results.

The modeled space in Ubik is treated as an environment compounded by buildings devoted to offices where the main activities are related to working tasks. Regarding the modeled agents, the simulation does not model humans with a cognitive point of view, but reactive. We are interested in human behavior as a member of a group of other humans which, predictably, will have the same behaviors to events of interest. The simulated space and agents must be considered to select valid AmI applications to be studied in Ubik. For example, a scenario in which the human has a reactive behavior is in an emergency. Figure 1 shows this concrete AmI application over Ubik.

The simulated building has a number of floors, stairs, elevators, hallways, rooms (bathrooms, kitchens, resources...), etc. A key idea is that the whole building is configurable, so the user can simulate virtually any scenarios. It is considered that any area without any specific purpose, such as a hallway or room, is a valid work space where an agent can place a table and do its work. More complex structures such as offices or cubicles would require more complex and laborious specifications to set the building, for example, expressed with ontologies. Figure 2 shows the floor of a building simulated with Ubik.

In Ubik, each worker is an agent. The workers have behaviors: staying on the work place, visiting another worker, downing to the street, walking around the office, convening meetings, attending meetings, attending emergency, going to certain rooms and so on. On the other hand, there are different types of workers: managers, subordinates, secretaries, service personnel, and so on. There is a fitness, linked to every worker, which determines the movement velocity. This velocity is lower if the agent is using the stairs. All these factors are configurable to simulate a multitude of societies in Ubik. Figure 2 represents the workers as an oval, more clearly whether they are on the work place, and a line indicates where they are watching.

To study AmI applications in Ubik, first the simulation is extended with the concept of as physical device (i.e. displays, computers, domotic devices, etc.). Then, the business logic is modeled as an AmI application (see figure 1). The effect is that the applications and devices have influence on the simulated space and, indirectly, in the agents. For example, an application of AmI in which the space and the agents are compatible with

[2]MASON website: http://cs.gmu.edu/∼eclab/projects/mason/

[3]Code, photos and videos of Ubik on: http://ubiksim.sourceforge.net

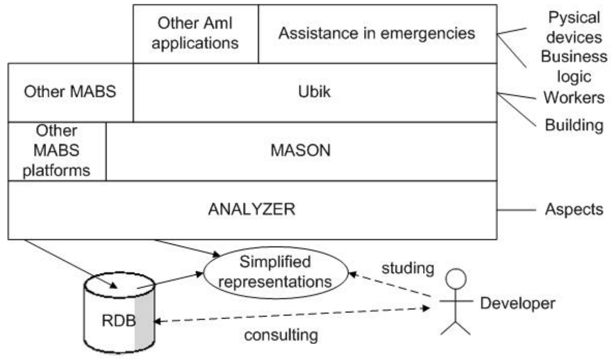


Fig. 1. Proposal of the paper

the Ubik simulation is the assistance in emergencies. With regard to the building, it is required to add sensors that detect emergencies, for example fire. Lighting text based panels are also required to show the emergency situation and to guide workers toward the nearest stairs or elevator. With regard to the agents, new features which affect to the application must be considered. For example, people using wheelchairs move with a normal speed through the office but are unable to use the stairs. The simulation must also consider the concept of configurable emergencies. This is useful to simulate lots of scenarios in order to make the application more robust.

An important factor in Ubik, for its purpose, is the ability to operate with a large number of agents as well as a large space for these agents. For example, the largest already built skyscraper, Taipei 101, has 101 floors and a total construction area of 420,000 square meters. One of the main goals of the simulation platform MASON is supporting a large number of agents and being efficient at the same time[12]. Ubik, developed in MASON, has supported studies with simulations of more than 200,000 agents and 200 floors efficiently. How could be made a real test of an AmI application for 200,000 people?. What is the best way to evacuate 200,000 people from a building? We hope that MABS in general and Ubik in particular get closer to the answer. The following section is a proposal for analysis, understanding and debugging of Ubik, or any other simulation, through forensic analysis.

IV. Debugging social simulations from a forensic perspective

As noted above, the analysis phase of MABS has been traditionally performed in an exploratory and intuitive manner[4]. One of the weaknesses of MASON, and MABS platforms in general, is that they offer few facilities to monitor and debug the simulated models[12]. MASON offers the possibility of fixing inspectors in individual properties (of the model or of the agents) to be monitored / recorded / modified from the simulation. However, we miss many options as views and records of the artificial society as a whole.

Forensic analysis is the process of understanding, re-creating, and analyzing arbitrary events that have previously occurred[11]. Serrano et al. [13] detail the creation of an infrastructure for forensic analysis in MAS. That technology can be brought to MABS. The key idea is to capture interesting elements of a simulation and to store them in a relational database (RDB). Once the RDB has captured the essence of a simulation, it is simple to make simplified representations of the stored data. These representations assist the analysis process. The field of MAS has already work with representations for these porpoises. For example, showing summaries of cooperation cores or similarities in the societies of agents[14]. Figure 1 shows the analyzer that records data in a RDB. It also shows a developer consulting that RDB and studying certain representations to analyze MABS.

Regarding how to capture elements of the simulation, it could be thought that it is something trivial. Calls to RDB can be simply programmed into the model or platform. This presents a major problem, this code would serve only for the specific model or the specific version of the platform. In addition, the logging code would be dispersed throughout the model or platform code. The solution in this proposal is the use of *Aspect oriented programming (AOP)*[8]. AOP can isolate the aspect of registering a MABS execution in order to treat it separately and in a modular manner. For now, our work is focused on the MASON platform. However, all the programmed aspects have equivalence in other platforms. Hence, replicating the results for other platforms is easy if they are distributed with the source code necessary to rebuild the platform as MASON does. Figure 1 shows how the analyzer can be used on different platforms. In this way we hope that our work is useful for the entire MABS community.

Regarding what elements are interesting to capture in a simulation, in a first approximation, it can be said that all those elements which were chosen to be monitored in the simulation are of interest. Here, the clarity of the platform code is a major factor. MASON code is extremely clear. Any method that starts with "get" in the code of an agent (implements the *Steppable* interface) or a model (implements *SimState*) returns a property that is displayed when this agent or model respectively is being inspected. Programming aspects to intercept all the calls to these methods is a trivial task. The body of the aspect store these properties in a RDB. Note that a complicated or confusing core of the platform would complicate extremely this stage. Once this basic approach has been covered, interesting elements present in all the simulations can be added to be registered. For example, the moment of creation of agents, their destruction or the interaction between them. The data

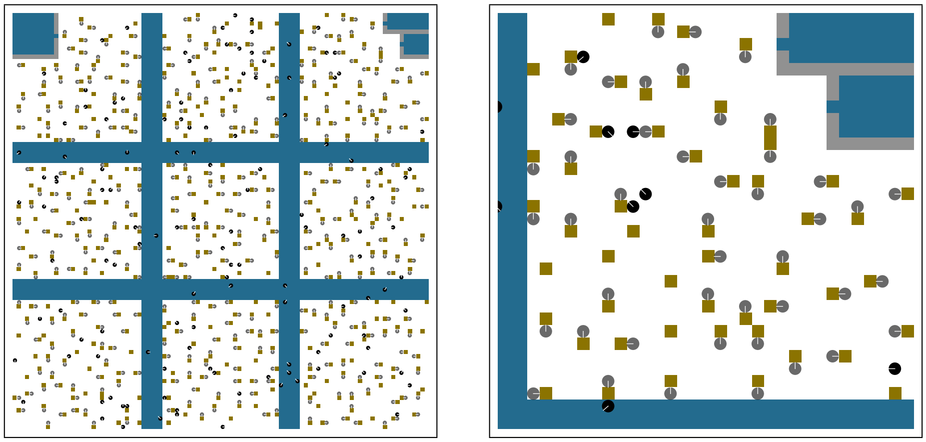


Fig. 2. On the left, a vectorial image of floor in Ubik with 500 agents. On the right, the zoom in the upper right corner of the floor

in RDB support simplified representations to be even more useful. In any case, the mere fact of having a RDB, which is a powerful tool for Querying, is a breakthrough that allows to do exploratory data mining, for example.

V. Conclusions and Future Work

This paper set the basis of an infrastructure to study applications of ambient intelligence (AmI) which involve a large number of users. The infrastructure is a multi-agent based simulation (MABS) called Ubik. The paper also introduces an infrastructure to assist the analysis, understanding and debugging of a MABS extrapolated of multi-agent systems (MAS). The infrastructure consists of forensic analysis by aspect oriented programming (AOP).

The introduction to the paper has introduced how new techniques are needed to test AmI applications and how the MABS, a versatile technology which is at its very peak, can be used for that purpose. It has also pointed that MABS have opted for an analysis in an intuitive manner while the field of MAS has several proposals for such purposes. Related work has explained the classic approaches to test AmI applications, how MABS also serve for these purposes and the most interesting theories to analyze MAS. Section III has presented Ubik as a MABS to test AmI applications and the details and restrictions on those AmI applications. It also has given the example of a possible application of high social interest and which can be tested using Ubik, emergency assistance in an office building. Section IV has presented a proposal for analysis, understanding and debugging MABS. This proposal is based on forensic analysis and AOP. It has also shown that this infrastructure provides the MASON platform with a powerful analytical capacity.

Regarding future works, the first one is to extend Ubik to make it more descriptive. For example, with detailed configurations of space and agents expressed with ontologies. We also intend to include several AmI applications of social interest as emergency assistance has been included. The infrastructure to assist the analysis of MASON simulations also generates promising future works. The first one is a survey about representations to simplify the data of the forensic analysis. Specifically, those representations that were successful in the development of MAS and may be exploitable in MABS. In general, we are interested in integrating our work with any technology that increases the usability of AmI applications or assists the analysis of MABS.

Acknowledgements

This research work is supported by the Spanish Ministry of Education and Science in the scope of the Research Project "Análisis, Estudio y Desarrollo de Sistemas Inteligentes y Servicios Telemáticos" through the Fundación Séneca within the Program "Generación del Conocimiento Científico de Excelencia".

References

[1] John J. Barton and Vikram Vijayaraghavan. Ubiwise: A ubiquitous wireless infrastructure simulation environment, tech. report hpl2002-303. *HP Labs*, 2002, 2002.

[2] Rosaria Conte, Nigel Gilbert, and ao Sichman Jaime Sim˙ Mas and social simulation: A suitable sommitment. In *Proceedings of the First International Workshop on Multi-Agent Systems and Agent-Based Simulation*, pages 1–9, London, UK, 1998. Springer-Verlag.

[3] Boris de Ruyter and Emile Aarts. Ambient intelligence: visualizing the future. In *AVI '04: Proceedings of the working conference on Advanced visual interfaces*, pages 203–208, New York, NY, USA, 2004. ACM.

[4] Alexis Drogoul, Diane Vanbergue, and Thomas Meurisse. Multi-agent based simulation: Where are the agents? In Jaime S. Sichman, Francois Bousquet, and Paul Davidsson, editors, *Proceedings of the Third International Workshop on Multi-Agent-Based Simulation MABS 2002, Bologna, Italy*, LNAI 2581, pages 1–15. Springer Verlag, Berlin Heidelberg, July 2002.

[5] D. Lewis T. O'Donnell D. O'Sullivan E. O'Neill, M. Klepal and D. Pesch. A testbed for evaluating human interaction with ubiquitous computing environments. In *TRIDENTCOM '05: Proceedings of the First International Conference on Testbeds and Research Infrastructures for the DEvelopment of NeTworks and COMmunities*, pages 60–69. IEEE Computer Society, 2005.

[6] Michael Friedewald, Olivier Da Costa, Yves Punie, Petteri Alahuhta, and Sirkka Heinonen. Perspectives of ambient intelligence in the home environment. *Telemat. Inf.*, 22(3):221–238, 2005.

[7] C.D. Kidd, R. Orr, G.D. Abowd, C.G. Atkeson, I.A. Essa, B. MacIntyre, E. Mynatt, T.E. Starner, W. Newstetter, et al. The Aware Home: A Living Laboratory for Ubiquitous Computing Research. *LECTURE NOTES IN COMPUTER SCIENCE*, pages 191–198, 1999.

[8] Ramnivas Laddad. *AspectJ in Action: Practical Aspect-Oriented Programming*. Manning Publications Co., Greenwich, CT, USA, 2003.

[9] Sean Luke, Claudio Cioffi-Revilla, Liviu Panait, and Keith Sullivan. Mason: A new multi-agent simulation toolkit. In *Proceedings of the 2004 Swarmfest Workshop*, 2004.

[10] M.C. Mozer. The neural network house: An environment that adapts to its inhabitants. In *Proceedings of the American Association for Artificial Intelligence Spring Symposium on Intelligent Environments*, pages 110–114, 1998.

[11] Sean Philip Peisert. *A model of forensic analysis using goal-oriented logging*. PhD thesis, La Jolla, CA, USA, 2007. Adviser-Sidney Karin.

[12] Steven F. Railsback, Steven L. Lytinen, and Stephen K. Jackson. Agent-based simulation platforms: Review and development recommendations. *SIMULATION*, 82(9):609–623, September 2006.

[13] Emilio Serrano and Juan A. Botia. Infrastructure for forensic analysis of multi-agent systems. volume Proceedings of the Programming Multi-Agent Systems Workshop at AAMAS'08, 2008. Estoril, Portugal.

[14] Emilio Serrano, Jorge J. Gómez-Sanz, Juan A. Botía, and Juan Pavón. Intelligent data analysis applied to debug complex software systems. *Neurocomputing*, 72(13-15):2785 – 2795, 2009.

Interpretation of User's Feedback in Human-Robot Interaction

B. De Carolis and G. Cozzolongo

***Abstract*— In this paper we will propose the use of social robots as interface between users and services in a Smart Environment. We will focus on the need for a robot to recognize the user's feedback, in order to respond and revise its behaviour according to user's needs. As we believe speech is a natural and immediate input channel in human-robot interaction, we will discuss the importance of recognising, besides the linguistic content of the spoken sentence, the attitude of the user towards the robot and the environment. In this way, the meaning of the user dialog will be made clear when hardly recognisable by the analysis of the utterance structure. Then, we will present the results of the application of a potential approach used for integrating the linguistic analysis with the recognition of the valence and arousal of the user's utterance. In order to achieve this goal, we collected and analysed a corpus of data to build an interpretation model based on a Bayesian network. Then we tested the accuracy of the model using a test dataset. Results will show that the integration of the linguistic content with the recognition of some acoustic features of spoken sentences perform better in recognising the key aspects of user's feedback.**

***Index Terms*— Intelligent robots applications, Man-Robot interaction.**

I. INTRODUCTION

FOLLOWING Ambient Intelligence (AmI) vision [1], a Smart Environment (SE) has the main aim of facilitating users in interacting with its services by making their fruition easy, natural and suitable to their needs. However, most of the times, user interfaces for handling functions and services of SE require navigation through menu options just to switch off the lights [2] or a complex setting procedure in order to change the behaviour of the environment in typical scenarios.

Smart Environments should assist their users in a proactive and responsive way, trying to recognise user's behaviours and needs so to respond as it is expected from them. Moreover, they should learn from user's feedback and progressively revise their interaction and behavioural rules.

To this aim in a previous project we developed an agent-based system for controlling a smart home environment [3]. In this system, the proactive response of the environment was mainly triggered by sensor data obtained from the user and other aspects in the surrounding environment (e.g. location, noise, temperature, light conditions) reasonably combined with explicit user actions in certain contexts. An intelligent agent acting "behind the scene", was able to infer user needs from his/her actions and, with reference to the current context, to answer the user in an appropriate way by changing the state of the environment [4].

After the first evaluation phase of the project, 80% of subjects declared to feel uncomfortable in interacting with an invisible presence and without explicit control over the home services. They declared to prefer to have an explicit physical interface as a counterpart to the environment, in order to request services, clarify some potential misunderstanding about task execution, express their approval and disapproval and change the behaviour of the environment according to their needs, as for the system illustrated in [5].

Afterwards, according to several research studies about the topic [6] [7], we decided to introduce the figure of an intermediary between the user and the environment; in particular we decided to employ a social robot as interface between the two participants.

Social robots can be thought, on one hand, as a mobile and intelligent interface to the environment system. On the other hand, they embody the role of friendly companions [8][9] improving the level of robots acceptance by humans. Sony AIBO and iCat, PaPeRo for instance, have been created with this purpose [10]-[12].

These kinds of robots are able to communicate and interact with users following a social behaviour. To this aim the robot could express personality and emotions but, most of all, it has to understand the social cues in interacting with the users [13] [14]. However, if we do not want users to consider robots as useless toys besides interacting, entertaining and engaging users, then they should support people in many aspects of daily life.

For instance, it could provide useful information for decision making, remind tasks and scheduled activities and warn users on possible dangers. In this case, it is necessary to consider some requirements such as the awareness about the user and environment situation, the recognition of his/her intentions, the generation of strategies and plans for satisfying the recognized user goals and the monitoring of effects of plans execution, in order to recover from errors.

As a result, the key goal of our research is to develop a SBDI (Social, Belief, Desire and Intention [15]) mind for this type of robot. Therefore, after having designed the basic architecture of a SBDI [16], we started to develop its behavioural models.

Berardina De Carolis is with University of Bari.
E-mail: decarolis@di.uniba.it

Giovanni Cozzolongo is with University of Bari.
E-mail: cozzolongo@di.uniba.it.

Since speech is a natural way for humans to interact with robots [17] [18], we initially faced the problem of interpreting the user's utterances during the interaction dialogue, in order to understand the valence and therefore the effect that the user wants to achieve through the provided feedback as to revise the reasoning behaviour accordingly.

In our system we consider two sources of knowledge coming from the user spoken input: the **linguistic information content** and the **acoustic features** of the utterance. Here we present results of an empirical study aiming at demonstrating the feasibility of this approach for clarifying the user's feedback intent in spoken utterances when interacting with a social robot acting as a mediator in smart environments.

The paper is structured as follows: in Section II the results of the first experiment for evaluating the impacts of social robots in the interaction with an SE are presented. In Section III we explain the motivations for using both linguistic content and acoustic features of the spoken sentence for disambiguating the user's feedback. Then we present how we annotated a corpus of human-robots spoken dialogues, collected during some experiments. Starting from the analysis of this corpus, we built a model aiming to interpreting the feedback intention of the user. This model is described in Section V. Then, we tested the accuracy of the model in properly recognizing the category of user's feedback. Conclusions and future work directions are discussed in the last Section.

II. UNDERSTANDING COMMUNICATIVE INTENT

In order to express their intentions humans use words and transfer emotions and emphasis by modulating their voice tone. In fact speech conveys two main types of information: it carries linguistic information according to the rules of the used language and paralinguistic information that are related to acoustic features such as variations in pitch, intensity and energy [19]-[23].

Usually the first component conveys information about the content of the communication and the second one about the user's attitude or affective state.

According to several studies [18][22] the linguistic analysis is not enough to properly interpret the real user's communicative intent towards the robot and the environment behaviour. For instance, the user can pronounce the same sentence with different emotional attitudes in order to convey different communicative intents. In fact, if the sentence *"where are you going?"* is pronounced with a negative attitude, it should not be interpreted as "the user want to know where the robot is going" but as "the user is probably disapproving my behaviour" and therefore it should be considered as a negative feedback. In this case, the robot should stop and revise its belief set accordingly.

Another example, typical of domestic environments, is the following: "It is hot in here". According to [24][25] this sentence can be interpreted in different ways:

– the user is *informing* the hearer about his/her perception of the room temperature;
– the user is *requesting* indirectly for someone to open the window;
– the user is *complaining* about the temperature (usually expressed *emotionally*);
– the user is *complaining* about the temperature implying that someone should better keep the windows closed (usually expressed using *sarcasm*).

In this case, the effects that the user wants to achieve are completely different: in one interpretation the user wants someone to open the window, in another one he/she wants to achieve the opposite effect.

Then, while words still play an important role in the recognition of communicative intents, taking into account the user attitude while speaking adds another source of knowledge that is important for understanding the interpretation of the utterance.

Integrating more than one modality should, in fact, help to resolve ambiguities and compensate for errors [26].

Starting from the work of Breazeal and Aryananda [27] and taking into account some theories that state the importance of the voice tone in interacting for educative and training purposes with pre-verbal infant [28] and domestic pets [29] we decided to start investigating how paralinguistic features of the spoken input can be used to improve the recognition of the user's communicative intent.

Research in emotional speech has shown that acoustic and prosodic features can be extracted from the speech signal and used to develop models for recognising emotions. Much of this research has used acted corpus of speech as training data and their research did not take into account the semantic content of what being conveyed [21]-[23].

According to Litman [30], in natural interactions users convey emotions by combining several signals. Therefore, acoustic-prosodic features should not be considered alone but combined with other information sources such as the linguistic content of the spoken sentence. Indeed, while acoustic-prosodic features address how something is said, lexical features represent what is said and, together, these features have shown to be useful for recognising intentions in human communication.

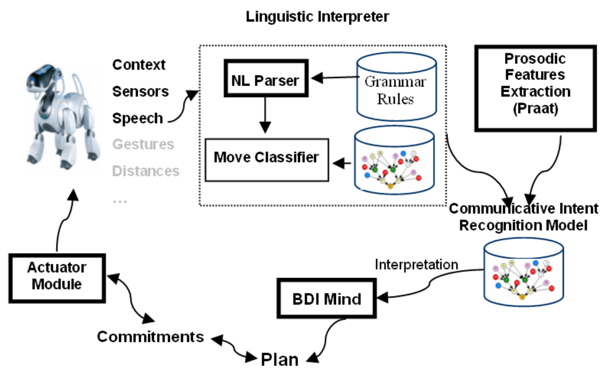


Fig. 1. Reasoning Schema of the Environment Mediator Robot

To this aim we have coupled a linguistic parser that transforms the sentence in a speech act with an acoustic analyzer able to extract the prosodic features of the user spoken input. Then, using a probabilistic model based on a Bayesian Network (BN), our system infers the user's intention by combining these two knowledge sources. BN allows handling uncertainty and incomplete world representation typical of this kind of domain [31]. In fact, interaction with smart environments contains various uncertainties that are mostly caused by the imperfectness and incompleteness of data or by the difficulty of a certain correct interpretation of human behaviour.

Figure 1 illustrates the schema architecture of the speech based intention recognition module integrated in a general BDI reasoning model [32].

The transcript of the spoken input is analyzed by the linguistic parser in order to recognize linguistically the type of user's move, then contextual features and the presence of cue words are used to provide a better classification of the move (i.e. inform, ask, etc.) using a BN model that will be illustrated later in the paper. At the same time the audio file relative to the user's move is analyzed using Praat functions [33].

Once we have extracted the parameters relative to pitch, intensity and energy of the sound, these, together with the interpreted category of move become the evidences or the initial distribution (in case of uncertainty in parsing the sentence) of some of the variables of the BN that allows inferring the valence and intention relative to the user's move.

Since the main aim of this research does not specifically concern the prosodic features analyser but the one of understanding how this integration can improve intention recognition, we will focus the rest of the description on model building and on its validation in the context of the interaction with a smart environment.

III. Corpus Annotation

There is a substantial body of literature on emotion recognition from speech [19]. These works show how certain emotional states are often correlated with particular acoustic features such as pitch (fundamental frequency F0), energy, timing and voice quality [34].

Obviously, the quantity and types of features to consider depend on how many emotions or attitudes are relevant for the purpose of the system.

For instance, in the work of [27] which is also aimed at recognising affective intent in human-robot interaction, it was important to distinguish among four categories of affective intents: approval, prohibition, attention, comfort. These categories were selected since the main aim of the work was to "educate" the robot like parents do with their children [28]. According to these results, pitch mean and energy variation seem to work with a good accuracy in distinguishing between the mentioned categories.

An experiment similar to the one presented in this paper was performed by Batliner et al [35] for recognizing emotions in children speech. Their system based on Neural Networks was able to classify speech with an overall recognition rate of about 70% for all four classes: *neutral* without any labels, *laughter, vocative* and *marked* for any other combination of prosodic peculiarities.

We did not consider children speech in our experiment because the potential users of our domotic application were adults. However, in the optic of extending the use of this application to families, children speech should be considered.

Another important research work aiming at recognizing the "social attitude" of users towards an Embodied Conversational Agent is represented by [36]. The adopted approach is also based on corpus annotation and analysis of linguistic and prosodic features in order to classify the user's attitude towards the agent.

In our research, as a first step for building the recognition model, the following classes of variables were considered, as they were deemed to be very important for achieving our goal:

– the typical *input moves* used to provide feedback during the interaction with the robot in a smart environment;
– the *feedback intention* to be recognized;
– the user *attitudes* that could influence intention interpretation.

A. Data Collection

In the data collection phase, our aim was to identify the basic set of user moves in terms of speech acts, communicative intents, attitudes, and to discover the relations among them.

In collecting the corpus dataset, we could use two approaches based on "acted" or "natural" moves. Obviously natural collected corpus is 'pure' and 'real' in terms of emotional content when compared with elicited or simulated content. However, it is difficult to collect these data in a controlled experiment setting. In fact, as far as concerns naturally-occurring behaviour, simply observing and recording the user's moves "in the wild" would produce a collection of data in which cause-effects and other types of relations are difficult to be generalized in machine understandable patterns [37].

A way to avoid this problem and to keep the naturalness in the user behaviour could consists in eliciting data in particular scenario that allow to control as many as relevant factors as possible [38]. Therefore we decided to collect data using this approach.

For collecting scenario-driven data we performed a set of Wizard of Oz (WOZ) studies [39]-[41].

In this experimental setting, we considered our research laboratory as a smart environment. We thought it was an appropriate place since it was well equipped (computers, microphones, cameras, wireless connection, bluetooth, etc.) for performing these types of experiments.

In this experiment we used two groups of 8 subjects each with an age between 20-28 years old equally distributed in gender and background. For both groups the lab room conditions were the same: the air condition was off, the windows were closed and the room was quite hot, the light was on. The Wizard was performed by the same person in all the experiments.

We assigned to each group the same goal:

finding a paper a professor left in the lab and complete the set of references within 10 minutes using a computer connected to the Internet. When finished, the subject had to send an email to the professor with the paper attached.

However, the behaviour of the robot was different for the two groups.

For the first group of subjects AIBO was behaving in a *cooperative* way: the robot was helping the user in achieving his/her goal, providing explanations. In particular AIBO was greeting the subject, showing him/her where the professor left the paper and which computer could be used to search for the missing references. During this subtask AIBO was near the subject observing what he/she was doing, ready to provide help on request without disturbing the subject during task execution.

For the second group AIBO was behaving *not cooperatively*. In particular, AIBO was not helping the user in achieving his/her goal by ignoring the subject requests for three times before answering. Moreover, when the subject was busy in completing the missing references of the paper AIBO was trying to distract him/her from task execution.

In our opinion, this difference would elicit two sets of users' behaviours that could emphasise interaction situations in which the subject approved or disapproved the robot behaviour.

In all the experiments the robot was controlled through an interface by the same person who guided its behaviour following a working script to evoke the previously mentioned situations. To this aim we have developed a tool that helps not only to drive the robot, but also to record the history of the interaction, included the user voice and the robot moves, beside all the data received from the robot sensors.

Before starting with the experiments we administered to each subject a simple questionnaire aiming at collecting some personal data (age and gender) and at understanding their background (department, year of course, artificial intelligence background).

Following the described approach we collected a corpus of 592 moves (25 moves on average for subjects belonging to the first group, 49 for those belonging to the second one).

Each move was recorded using a wireless microphone whose output was sent to the speech processing system. We assigned to each utterance, a unique identifier, the correspondent transcript and the related 16-bit single channel, 8 kHz signal (in a .wav format).

B. Data Annotation

This corpus was annotated in a previous phase of the project as reported in [42], but with two annotators only for each step. In order to improve the reliability of the annotation process, we decided to use more people for each step. Moreover, we added a new annotation step related to the presence of cue words that are typically used when conveying a particular communicative intent. Then, we re-analyse all the data.

Since we are interested in finding the relations between the type of user move, the use of cue words, the attitude and the feedback intention, we divided the annotation process into 4 steps using different annotators for each step so that the labelling of each component was not influenced by the others. Then we completed the annotation by adding the acoustic features related to each sentence.

Examples of collected user moves are listed in Table I.

TABLE I
SAMPLE OF COLLECTED FEEDBACK MOVES TRANSLATED IN ENGLISH.

User spoken sentences
Hi AIBO.
It's hot in here.
What are you doing?
Which is the computer I'm supposed to use?
Where are you going?
OK.
Yes/No.
Thanks.
You are stupid.
Well done.
Good.
Can you show me where is the computer?
Show me where is the computer.
Speed up.
I told you several time don't bother me.
Please stop bothering me.
AIBO.
…

Step 1.

As far as the type of linguistic content is concerned, our aim was to formalize the spoken sentence as a Speech Act.

This formalization was chosen since its semantics is related to effects that the speech act may achieve in the user mental state and therefore it can be used to let our robot reasoning about the beliefs that induced the user to say something or about the effects that he/she wanted to achieve in the mental state of the robot.

For this annotation step we chose the set of categorical labels listed in Table II. This list is a subset of the Speech Act family.

In this annotation phase we needed people that were familiar with the speech act theory. Therefore, we involved five annotators distributed as follows: a professor teaching computational linguistic, two Ph.D. students researching about these topics and two under-graduated students of the computational linguistic course.

TABLE II.
CATEGORIES OF USER MOVES

Move Type	Function
Greet(U,AIBO)	The user U greets AIBO
Call(U,AIBO)	U calls AIBO
Request(U,AIBO,a)	U asks AIBO to perform an action a
Order(U,AIBO,a)	U orders to AIBO to perform an action a
Inform(U,AIBO,f)	The user informs AIBO about a fact f.
Ask(U,AIBO,f)	U makes a question to AIBO about a fact f.
Thank(U,AIBO)	U thanks AIBO.
Reproach(U,AIBO)	U reproaches AIBO's behaviour
Compliment(U,AIBO)	U makes a compliment to AIBO.
Acknowledge(U,AIBO)	U acknowledges AIBO's behaviour

We provided the annotators with the set of human written transcripts of all subjects moves collected during the WOZ study in both modalities and we asked them to use the labels in Table II to annotate them. They could also introduce new labels if they did not recognize in the move any of the listed speech acts.

In order to test the validity of our findings we used the method found in [43]. Then, to have a measure of the level of agreement between annotators, we calculated the percentage of cases that were labelled in the same way, we computed the percentage of agreement and then we calculated the Kappa statistics**.**

Kappa is an index which compares the agreement against what might be expected by chance. Kappa can be thought of as the chance-corrected proportional agreement and possible values range from +1 (perfect agreement) via 0 (no agreement above that expected by chance) to -1 (complete disagreement). This index is widely accepted in the field of content analysis and allows different results to be compared.

Table III shows the frequency of every move category in the corpus and summarizes inter-annotator agreement. The annotators agreed in recognising most of the moves from their linguistic content. Only the level of agreement about the "inform" and "request" speech acts was lower than we expected.

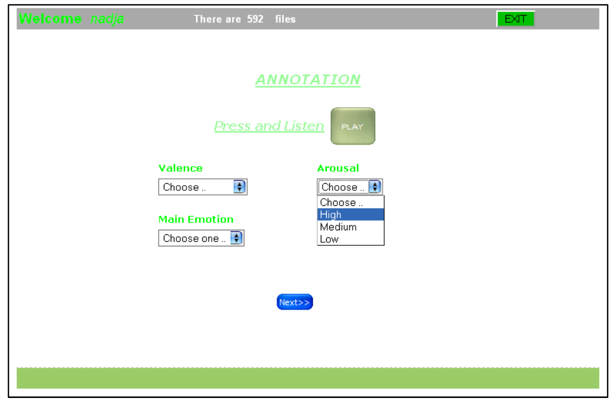


Fig. 2. Annotation web page

Step 2.
In this step our main goal was to identify which was the user's attitude in the feedback sentence that could change the linguistic interpretation of the underlining intention.

In this annotation phase we did not need people with a specific knowledge, as in the *step 1*, therefore we used a website (Figure 2) and we invited 25 people to annotate the corpus.

For our purpose, we did not need a very sophisticated distinction between all the possible user affective states.

In our opinion, in this type of interaction it is important to understand which is the **valence** of the user attitude, since it is related to the achievement of a particular goal and the **arousal**, since it indicates how important was that goal for the user and therefore indicates the urgency to recover that state.

In the case of feedback sentences these two dimensions are more important than recognising the emotion itself.

For instance, a negative valence will indicate a failure in the achievement of the user's goal and if it is correlated with a high arousal will allow to distinguish hanger towards an event (e.g.: a move of the robot), from sadness related to a personal mental state.

TABLE III.
INTER-ANNOTATOR AGREEMENT

Label	% agreement	%frequency	Kappa
Greet	0,9	12%	0.8
Call	0.8	9%	0.6
Request	0.7	15%	0.4
Order	0.8	10%	0.6
Inform	0.6	4%	0.2
Ask	0.8	13%	0.6
Thank	1	9%	1
Reproach	0.8	13%	0.6
Compliment	0.8	7%	0.6
Acknowledge	0.9	8%	0.8

Therefore, for the valence dimension we considered the *positive*, *neutral* and *negative* categories measured along a 5-point scale (from 1- very negative to 5- very positive).

The *positive* valence was important to recognize positive feedback towards the robot and the environment while the *negative* one was important to recognize the disappointment towards the robot behaviour. The *neutral* attitude was considered important for interpreting the user intention only from the linguistic part of the user input.

The arousal was measured in a 3-point scale from high to low. The annotators had to listen to the audio file and set the appropriate value in the drop down menu (Figure 2). While the setting of the arousal and valence was compulsory, annotators could optionally set the main emotion recognized in the sentence. This label, however, is still not considered in building the model. But it could be used in a future phase of the project. The resulting annotation was then stored in a database. Results are shown in Table IV-a and IV-b.

TABLE IV-A.
VALENCE

	% agreement	Kappa
very positive	0.8	0.6
Positive	0.58	0.166
Neutral	0.55	0.117
Negative	0.75	0.5
very negative	0.95	0.9

TABLE IV-B
AROUSAL.

	% agreement	Kappa
High	0.9	0.8
Medium	0.5	0
Low	0.7	0.4

Apparently the annotators clearly agreed in recognizing strong attitudes (very positive and very negative valence and high arousal) while we cannot say that for neutral and positive valence there was a different agreement from the one expected by chance. Moreover, a negative attitude was better recognized than a positive one.

Step 3.

This step was aiming at understanding which categories of feedback were relevant in our domain. Another group of 25 annotators were invited to annotate the intention underlying the feedback sentence by listening to the audio on the web site. The labels used in this case represent the effect that a speech act may achieve (its perlocutionary part)[24]. In this way it was in theory possible to relate the recognised move (linguistic part) and its prosodic characteristic with the move effect.

We identified the ones reported in the first column of Table V together with the percentage of agreement and the results of the Kappa statistics.

TABLE V.
RESULTS OF THE INTENTION ANNOTATION PHASE.

Feedback Type	***Function***	***agreement***	***kappa***
WantToDo(U,AIBO,*a*)[a]	The user U want the robot to do some action	0,75	0,5
WantToKnow(U,AIBO,*f*)	The user U wants to know a fact.	0,8	0,6
KnowAbout(AIBO,*f*)	The user U wants that AIBO knows about a fact.	0,7	0,4
Approve(U,AIBO)	Positive feedback corresponding to the intention to approve;	0,75	0,5
Disapprove(U,AIBO)	Negative feedback corresponding to the intention to disapprove;	0,8	0,6
GetAttention(U,AIBO)	The user U wants to get the attention of the robot;	0,9	0,8
SocialCue(U,AIBO,*c*)	The user U performs a social communicative act such as greeting, soothing, etc..	0,75	0,5

The overall agreement is good, showing that listening to the sentences (linguistic content and acoustic features) allows interpreting its meaning and the user's feedback intention with a good accuracy.

Moreover, in order to understand the correspondence

TABLE VI.
RELATION BETWEEN SPEECH ACTS AND FEEDBACK INTENTIONS.

	Greet (SC)	***Call (GA)***	***Request (WTD)***	***Order (WTD)***	***Inform (KA)***	***Ask (WTK)***	***Thank (SC)***	***Reproach (DA)***	***Compliment (A)***	***Acknowledge (A)***
WantToDo (WTD)	0	0	61	47	0	0	0	0	0	0
WantToKnow (WTK)	0	0	0	0	0	47	0	0	0	0
KnowAbout (KA)	0	0	0	0	7	0	0	0	0	0
Approve (A)	0	0	0	0	10	0	36	0	27	42
Disapprove (DA)	0	13	28	12	11	31	10	59	8	0
GetAttention (GA)	0	32	0	0	2	0	3	18	0	0
SocialCue (SC)	71	8	0	0	1	0	4	0	6	6

between the speech acts annotated on the corpus transcripts and the intentions recognised from the spoken sentences we calculated a matrix showing the relations and the differences between the two annotations.

In this matrix, shown in Table VI, the columns indicates the speech acts and their intended communicative effect as formalized in the Speech Act Theory and the rows the categories of feedback that we are considering in our domain.

The results seem to confirm our hypothesis and therefore the need to integrate this two knowledge sources (the linguistic and the acoustic component of the sentence) for recognising the user intention. In fact, almost all the communicative intentions can be expressed using different speech acts. But, while it is easy to classify a *thank* or a *greet* as the intention to provide a "Social Cue", a "disapprove" can be expressed using almost all the categories of speech acts. As we will see later on, the integration of the linguistic interpretation with the prosodic features allows recognising it with a quite good accuracy.

Step 4.

The goal of this step was to recognize in the transcript of the collected moves the presence of cue words that could help in identifying the nature of the feedback, according to the categories in Table VII. Cue words could belong to an **emotional lexicon** or could be just words with a clear meaning, such as a verb or an object.

We classify as belonging to the **emotional lexicon** all the words, in a language, containing information about an emotional state in their semantic representation. This could be either an emotion or a feeling. Besides exclamations (i.e. "oh, damn!"), cue words could be nouns (fear, thanks, etc.) or verbs (to hate, to love), or adjectives (angry, furious, sad, happy, etc.).

In our domain, cue words are classified as belonging to these categories:

– *social*: i.e. hi, ciao, hello, thanks, AIBO, etc.
– *attention*: i.e. AIBO, hei, you, aho, etc.
– *positive*: i.e. well, good, yes, love
– *formal*: i.e. please, could
– *negative*: no, stop, bad, stupid, hate
– *action verbs*: move, do, go, speed up, help, etc.
– *conversation verbs*: tell, know, etc.

Table VII shows the relation between the category of intention and the category of cue words with the level of agreement.

Step 5.

After the human annotation process, the dataset needs to be completed with the related acoustic features. We used Praat functions in order to perform a macro-prosodic or global analysis and to extract from the audio file of each move features related to:

TABLE VII.
RELATION BETWEEN INTENTIONS AND CUE WORDS.

Feedback Type	cue words categories	agreement	kappa
WantToDo(U,AIBO,a),	formal, action verbs	0.7	0,4
WantToKnow(U,AIBO,f)	formal, conversation verbs	0.75	0,5
Approve(U,AIBO)	positive	0.7	0,4
Disapprove(U,AIBO)	negative	0.75	0,5
GetAttention(U,AIBO)	attention	0.8	0,6
SocialCue(U,AIBO,c)	social	0.9	0,8

- The variation of the fundamental frequency (f0): pitch minimum, mean, maximum and standard deviation;
- The variation of energy (RMS): max and standard deviation.

We did not consider the speech rate, because we are going to interpret very short sentences and move on average in our domain.

As we have already pointed out we consider the *activation* and the *arousal*, which can be used to distinguish high activation emotion from low activation one, by analysing the value of the *f0*; the *average f0* values, its *range,* its *dynamics* and *intensity* are all higher for high activation emotion. In this way we can distinguish *surprise*, *joy*, *anger*, *fear* from *sadness* and *disgust*.

The *valence* dimension allows distinguishing positive from negative emotions. Negative emotions features are characterized by fast f0 lowering, rising in intensity, high prominence of the maximum energy values, longer pauses. On the basis of this features it is possible to distinguish fear and anger from joy.

After adding these features to the dataset, it was necessary to transform the numeric values relatives to the pitch and energy into discrete values, in order to handle these data in our model.

To this aim we used a three-value scale (low, medium, high). In order to assign each numerical value corresponding to the pitch and the energy values to one of these discrete values, we calculated the 33% and 66% percentage. We divided in this way the numeric interval of each of the extracted features into three parts. Then, values falling into the first numeric set were considered as low, those falling in the second one as normal and the rest high.

Step 6.

Then, we completed the annotation of the collected corpus with the following data: the name of the audio file, the gender of the subject that pronounced the sentence, the event that triggered the user sentence, the user and environment situation. These additional data were recorded during the experiment.

The following Table VIII shows an annotated element of the corpus corresponding to the question “where are you going?” expressing the intention to disapprove the robot action.

TABLE VIII.
AN EXAMPLE OF THE ANNOTATION FOR AN ELEMENT

Name	Value
File	Aibo10
Sentence	where are you going?
Gender	f
Move	ask
Env context	c 10
Event	AIBO makes noise
Feedback	disapprove(U,AIBO,move)
Valence	1
Arousal	3
P mean	High
P max	High
P min	High
P var	Normal
E max	High
E var	High

The variables *p_min*, *p_max*, *p_mean*, *p_var* identify the values of the pitch minimum, maximum, medium and its variance, respectively. The values of the maximum energy and its variance are identified with *e_var* and *e_max* respectively.

IV. BUILDING THE MODEL

In order to learn the dependencies among acoustic features, linguistic content of the user move, attitude and intention, we decided to use a probabilistic model, as it seems to be appropriate in the considered context. Indeed, understanding human attitude and intention from speech input involves capturing and making sense of imprecise, incomplete and sometimes conflicting data [31].

As far as concern the building of the model we could use a learning algorithm, able to learn both the structure and the parameters of the network, or we could design the structure according to the theories that guided our experiment steps and then learn the parameters accordingly.

In a first phase of the project we used the NPC learning algorithm of Hugin 6.5 (see www.hugin.com for more details) on the labelled database of cases in the corpus. In this way we could learn both the structure and the parameters of the network.

We randomly extracted 37 cases from the labelled database of selected cases, because 37 was the average of moves for each subject collected during the experiment.

As we will show later on, this subset was used in the testing phase, while the rest of cases (555) were used for learning the network.

The model resulting after some optimization steps is shown in Figure 3.

Variables in this network are mainly related to:

– the recognized **move category**: this information is extracted by the linguistic parser and belongs to one of the categories listed in Table II;
– the presence and category of **cue words** that may change the linguistic interpretation of the move; this variable may then assume values in the set described in Table VII;
– the **environment context** situation that may have triggered the feedback move (i.e. the room temperature is 32° C);
– the **robot action**;
– the **valence** associated to the move that can assume values in the set very positive (5), positive (4), neutral (3), negative (2) and very negative (1);
– the **arousal** associated to the move that can assume values in the set high (3), medium (2) and low (1);
– the extracted **acoustic features**: pitch features and energy variation and maximum were considered relevant to the recognition;
– the **feedback intention** beyond the speech act: this is the variable that we want to monitor using the model and can assume values in the set described in Table VI.

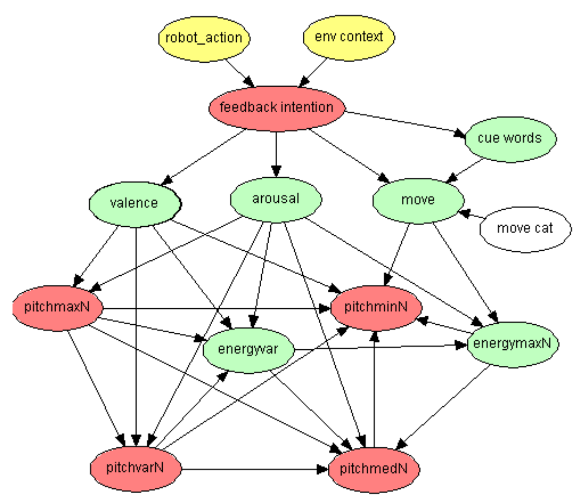


Fig. 3. The structure of the feedback recognition model.

In learning the model, we have set some constraints in the higher level of the structure. In particular we set the dependence of the *feedback intention* node with the four main variables resulting from the annotation process: the valence, the arousal, the move category and the cue word. These three variables, then, are related to the acoustic parameters whose relations are represented in the lower section of the network. This part has been completely learnt both in the structure and the parameters.

Further more, as we wanted to use the model in a more dynamic way, during the interaction, we had to link the *feedback* to the *robot_action* and to the *env_context* variable. This two variables could help in identifying the cause of the user feedback.

For examples, the user could disagree about the temperature situation in the room by saying: "it is hot in here" with a negative attitude. In this case the intention to disapprove does not regard directly the robot behaviour but the comfort situation of the environment. While, in case the user reproaches the robot when it moves or makes some noise, the disapproval regards its action.

V. EVALUATING THE MODEL

The evaluation of the model was performed on a subset of data extracted randomly from the selected corpora. As we said in the previous Section, we selected randomly 37 entries from our annotated corpus before the learning process (37 was the average of moves for each subject).

Then as input evidences of our model we used the following values: first **only** the **speech features**, then only the **linguistic content** of the **move** and then **both** of them. Finally we compared the predicted results with the human annotation in all the three cases.

The global accuracy of the model, expressed as the total of correct predictions for every category on the total of the moves in the dataset, is of about 60% in the case of speech features only or linguistic content only, while it increases to the 75% when considering both features.

Since we use a probabilistic model we considered as correct a prediction of the value of the feedback intention variable the one of its six states with the highest probability.

Table X reports detailed results about the prediction of each feedback category in the three cases.

We used *F1-score* that combines *precision* and *recall* to evaluate the performance of the model. For a feedback intention category fc_i , we define *precision* as the ratio of the number of moves classified correctly as fc_i and the number of moves in the corpus annotated as fc_i . We define *recall* as the ratio of the number of moves classified correctly as fc_i and the total number of moves in the corpus belonging to the class fc_i . The closer the precision, the recall and F1-score are to 1, the more accurate is the prediction.

As a general comment we can say that the model performs a better prediction when using speech + move data vs. the other two conditions, especially in those cases when the linguistic content contrasts the voice tone in identifying the type of feedback intention. This is evident in the case of the "Approve" and "Disapprove" class where the recognition from the linguistic content only performs worse.

For instance, considering the case in which the robot moves towards a place despite the user will, the sentence:

"where are you going?"

is parsed as

"Ask(U,AIBO,where(is_going,AIBO))"

with a cue word belonging to the *action_verb* category.

The model, in this case, classifies the feedback intention as:

"Want_to_Know(U,where(is_going,AIBO))"

taking the state of the variable with the highest probability (47% on 6 possible choices), while the probability of the disapprove intention is around 16%.

However, if we take into account evidences about acoustic features, then the most probable feedback intention becomes the disapproving one (63%) and the most probable valence of the voice tone becomes negative (54%) with an high arousal (82%).

To monitor the behaviour of the robot we have developed a

TABLE IX.
CONFUSION MATRIX FOR INTENTION RECOGNITION - SPEECH ONLY/TEXT ONLY/ SPEECH+TEXT .

		WantToDo	*WantToKnow*	*KnowAbout*	*Approve*	*Disapprove*	*GetAttention*	*SocialCue*	*F1-score*
WantToDo	Speech	55	31	0	0	20	2	0	0.59
	Text	69	37	0	0	0	2	0	0.62
	S+T	74	21	0	5	8	0	0	0.77
WantToKnow	Speech	10	26	0	0	11	0	0	0.41
	Text	17	30	0	0	0	0	0	0.41
	S+T	9	32	0	0	6	0	0	0.59
KnowAbout	Speech	0	0	4	1	2	0	0	0.44
	Text	0	0	5	0	0	2	0	0.5
	S+T	0	0	4	1	1	1	0	0.61
Approve	Speech	0	0	3	71	0	19	18	0.72
	Text	11	12	5	51	0	17	15	0.61
	S+T	0	0	1	85	0	10	15	0.83
Disapprove	Speech	13	17	3	0	119	17	0	0.68
	Text	17	20	3	0	89	21	19	0.66
	S+T	2	9	1	0	136	9	12	0.81
GetAttention	Speech	1	5	1	0	15	29	4	0.43
	Text	0	0	0	0	12	38	5	0.52
	S+T	0	0	0	0	9	41	5	0.66
SocialCue	Speech	0	0	0	13	15	15	52	0.61
	Text	0	0	0	5	0	11	79	0.74
	S+T	0	0	0	3	7	9	76	0.75

tool. Figure 4-A shows how this tool allows monitoring the result of the propagation of the evidences in the model in the described example.

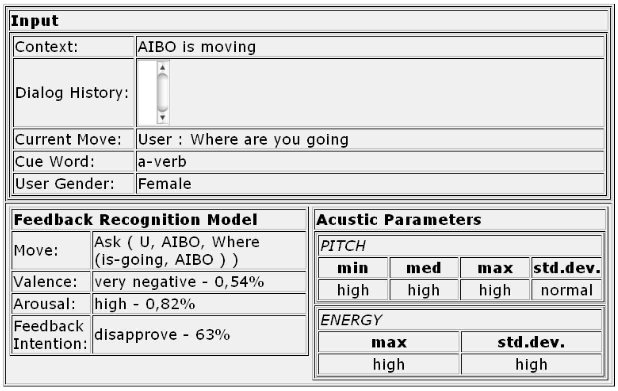


Fig.4a The result of the model prediction after the propagation of evidence values.

Figure 4-B shows another example of prediction. In this case the user enters in the room and says with a negative attitude:

"it is hot in here"

in this case the model recognizes another disapproving feedback about the environment situation even if the move is classified linguistically as an inform speech act.

This is inferred by considering the very negative valence and high arousal deriving from the prosodic features and the information coming from the environment context.

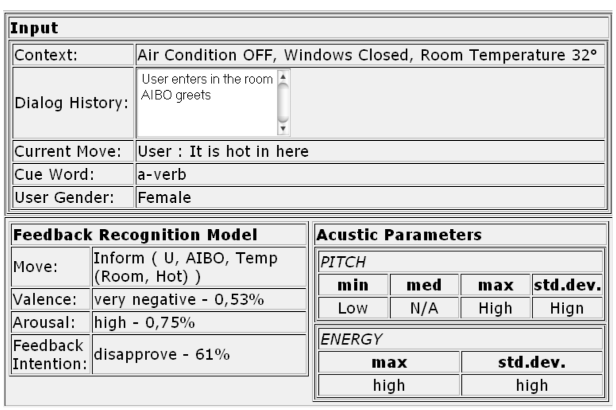


Fig. 4b. Another example of prediction.

VI. Conclusions and Future Work

As part of ambient intelligent research, there is a need to recognize the user's intention during the interaction in order to adapt the environment behaviour accordingly. The interaction may happen in a seamless way, for instance combining sensors data, or through an embodied agent that acts as a "mediator" between the user and the environment.

In this paper we presented our results in recognising user's feedback intention when this mediator is represented by a social robot that is considered responsible for the success of interaction between the user and the environment.

Since spoken input is considered one of the more natural ways to interact with robots, we focused our research on the analysis of the spoken sentence using two information sources: from the linguistic content and intonation of the voice. In particular, we performed one set of experiments based on the Wizard of Oz method whose results were annotated and analyzed in terms of linguistic communication content, presence of cue words, valence and arousal of the voice tone and intention.

This collected corpus was used to learn the structure of the Bayesian network model to be used for recognising the probability for a user to have a particular intention toward the robot and/or the environment.

In order to validate this model we used as testing dataset a subset of moves randomly extracted from the corpus.

The performed experiment proved that using both knowledge sources for recognising the user's intention improves the prediction accuracy of the model.

The same approach could be used in other domains such as e-learning, e-health, etc. where user's emotion and intention recognition plays a key role for adjusting the behaviour of the system. In e-learning, for instance, the student engagement is constantly monitored [30]. Understanding user's feedback is a crucial aspect of effective learning environments, therefore monitoring continuously and unobtrusively student's frustration, boredom, enthusiasm, is important for tuning the presentation of learning material, determining the success of the learning process [44].

In our future work we plan to integrate this model in the architecture of NICA (as the name of the project Natural Interaction with a Caring Agent). NICA is a robot aiming at assisting elderly people in their daily life tasks. NICA has to be able to provide what we collected as two types of results, one showing a service-oriented assistance but also social and emotional feedback. For this reason it is important to extract from the user spoken sentence information about his/her attitude towards the robot and the environment.

In this application domain, we need to investigate on how previous intentions influence the current one so as to express this relation as a function to build a temporal link in a Dynamic Belief Network (DBN, [34]). Moreover, we want to use the recognised feedback intention as a triggering condition for planning the most appropriate robot's behaviour.

Results obtained so far from the overall response of users to the experiments seem to confirm that the idea of using a social robot as a mediator is a good way to overcome barriers that people may find in using smart environments.

Acknowledgments

We wish to thank Vincenzo Silvestri and Irene Mazzotta for their help during the experiments.

REFERENCES

[1] G. Riva, F. Vatalaro, F. Davide, and M. Alcaniz, (Editors). “Ambient Intelligence: the Evolution Of Technology, Communication And Cognition Towards The Future Of Human-Computer Interaction” (Emerging Communication), 2005.

[2] D. Randall, “Living Inside a Smart House: A Case Study,” in Harper, R., (editor), Inside the Smart Home, Springer-Verlag, London. 227-246. 2003.

[3] B. De Carolis, G. Cozzolongo, S. Pizzutilo, V.L. Plantamura, “Agent-Based Home Simulation and Control”. In Proceedings of ISMIS 2005, LNCS. Springer: 404-412. 2005.

[4] B. De Carolis, G. Cozzolongo, S. Pizzutilo. “A Butler Agent for Personalized House Control”. In Proceeding of ISMIS 2006, LNCS. Springer. 2006.

[5] A. Gárate, N. Herrasti, and A. López, “GENIO: an ambient intelligence application in home automation and entertainment environment”. In Proceedings of the 2005 Joint Conference on Smart Objects and Ambient intelligence: innovative Context-Aware Services: Usages and Technologies (Grenoble, France, October 12 - 14, 2005). sOc-EUSAI '05, vol. 121. ACM Press, New York, NY, 241-245. 2005.

[6] G. von Wichert, and G. Lawitzky. “Man-Machine Interaction for Robot Applications in Everyday Environments”. In Proc of the IEEE Int. Workshop on Robot and Human Interaction 2001 (RO-MAN 2001), Paris, Bordeaux, Frankreich, 18.-21. Sept. 2001.

[7] S. Buisine, and J.C. Martin, “Experimental Evaluation of Bi-Directional Multimodal Interaction with Conversational Agents”. Proceedings of Interact’03. 2003.

[8] K. Dautenhahn “Robots as social actors: Aurora and the case of autism”. In Proc. CT99, The Third International Cognitive Technology Conference, August, San Francisco, pages 359-374, 1999.

[9] T. W. Bickmore and R. Picard. “Establishing and maintaining long-term human-computer relationships”. Transactions on Computer-Human Interaction, 2004.

[10] J. Osada, S. Ohnaka, M. Sato: “The scenario and design process of childcare robot, PaPeRo”. Advances in Computer Entertainment Technology 2006.

[11] C. Bartneck, and J. Reichenbach, “Subtle emotional expressions of synthetic characters”. The International Journal of Human-Computer Studies (IJHCS), 62(2), pp. 179-192. 2005.

[12] Sony. (1999). AIBO. from http://www.AIBO.com.

[13] C. Breazeal. “Toward sociable robots”. Robotics and Autonomous Systems, 42:167–175, 2003.

[14] L. Cañamero and A. Blanchard and J. Nadel J., “Attachment Bonds for Human-Like Robots”, International Journal of Humanoïd Robotics. 2006.

[15] H. Jiang, and J.M. Vidal “From Rational to Emotional Agents”. In Proceedings of the AAAI Workshop on Cognitive Modeling and Agent-based Social Simulation, 2006.

[16] B. De Carolis, G. Cozzolongo, “Social Robots for Improving Interaction in Smart Environments”. In Proocedings of the Workshop Emotion in HCI. In conjunction with the 20th British HCI Group Annual Conference AISB07. 2007.

[17] A. Drygajlo, P.J. Prodanov, G. Ramel G., M. Meisser, and R. Siegwart, “On developing a voice-enabled interface for interactive tour-guide robots”. Journal of Advanced Robotics, vol.17, nr. 7,p.p. 599-616, 2003.

[18] S. Nakamura, Toward Heart-to-Heart Speech Interface with Robots. ATR UptoDate. Summer 2003.

[19] R. Banse, and K.R. Scherer, ”Acoustic profiles in vocal emotion expression”. Journal of Personality and Social Psychology. 1996.

[20] C.M. Lee, S.S. Narayanan, R. Pieraccini, “Combining acoustic and language information for emotion recognition”. Proceedings of ICSLP, 2002.

[21] J. Liscombe, J. Venditti, and J.Hirschberg, “Classifying subject ratings of emotional speech using acoustic features,” in Proc. of EuroSpeech, 2003.

[22] S. Mozziconacci and D.J. Hermes, "Role of intonation patterns in conveying emotion in speech", in Proceedings, International Conference of Phonetic Sciences, San Francisco, August 1999.

[23] P.Y. Oudeyer. “The production and recognition of emotions in speech: Features and Algorithms”. International Journal of Human Computer Studies, 59(1-2):157-183. 2002.

[24] J.L. Austin,. How to do things with words. Oxford: Oxford University Press. 1962.

[25] J. Searle, “Speech Acts: An Essay in the Philosophy of Language”, Cambridge, Eng.: Cambridge University Press. 1969.

[26] W.E. Bosma and E. André, “Exploiting Emotions to Disambiguate Dialogue Acts”, in Proc. 2004 Conference on Intelligent User Interfaces, January 13 2004, N.J. Nunes and C. Rich (eds), Funchal, Portugal, pp. 85-92, 2004.

[27] C. Breazeal and L. Aryananda, "Recognizing affective intent in robot directed speech", Autonomous Robots, 12:1, pp. 83-104. 2002.

[28] A. Fernald, “Four-month-old infants prefer to listen to motherese”. Infant Behavior & Development, 8, 181-195. 1985

[29] J. Serpell, “The Domestic Dog: Its evolution, behavior, and interactions with people”. Cambridge University Press, New York, New York. 1995.

[30] D. Litman, K. Forbes, S. Silliman, “Towards emotion prediction in spoken tutoring dialogues”. Proceedings of HLT/NAACL, 2003.

[31] F.V. Jensen. “Bayesian Networks and Decision Graphs. Statistics for engineering and information science”. Springer, New York, Berlin, Heidelberg, 2001.

[32] A. Rao, M. Georgeff: BDI agents: From theory to practice. In Proceeding of the 1st International Conference on Multi-Agent Systems (San Francisco, CA, June 1995), V. Lesser, Ed., AAAI Press, pp. 312–319. 1995

[33] PRAAT: www.fon.hum.uva.nl/praat.

[34] I. Poggi and E. Magno Caldognetto, “Il parlato emotivo. Aspetti cognitivi, linguistici e fonetici”. Atti del Convegno Il parlato italiano. Napoli, 12-15 febbraio 2003. Napoli, D’Auria. CdRom, 2004.

[35] de Rosis, F, Batliner, A, Novielli, N, and Steidl, S (2007). ’You are Sooo Cool, Valentina!’ Recognizing Social Attitude in Speech-Based Dialogues with an ECA. In: Affective Computing and Intelligent Interaction, edited by Ana Paiva, Rui Prada, Rosalind W. Picard. Springer Berlin / Heidelberg, pages 179-190.

[36] Batliner, A., Hacker, C., Steidl, S., N¨oth, E., D’Arcy, S., Russell, M., and Wong, M. (2004). “You stupid tin box” - children interacting with the AIBO robot: A cross-linguistic emotional speech corpus. In Proc. LREC 2004, Lisbon.

[37] D. Kulic and E. A. Croft. “Estimating Intent for Human Robot Interaction”. Proc of the Int. Conf. on Advanced Robotics. Coimbra, Portugal, June 29 - July 3, 2003.

[38] F. Enos and J. Hirschberg, “A framework for eliciting emotional speech: Capitalizing on the actor's process”. LREC 2006 Workshop on Corpora for Research on Emotion and Affect, Genova, 2006.

[39] S. Whittaker, M. Walker, M. and J. Moore, “Fish or Fowl: A Wizard of Oz evaluation of dialogue strategies in the restaurant domain”. Language Resources and Evaluation Conference. 2002.

[40] S. Oviatt, and B. Adams, “Designing and Evaluating Conversational Interfaces With Animated Characters”. In J Cassell, J Sullivan, S Prevost and E Churchill: Embodied Conversational Agents. The MIT Press, 2000.

[41] I. Bretan, A.L. Ereback, C. MacDermid, and A. Waern, “Simulation-Based Dialogue Design for Speech-Controlled Telephone Services”. Proceedings of CHI’95. 1995.

[42] G. Cozzolongo, B. De Carolis, S. Pizzutilo, “Social robots as mediators between users and smart environments”. In Proceedings of the 12th international Conference on intelligent User interfaces (Honolulu, Hawaii, USA, January 28 - 31, 2007). IUI '07. ACM Press, New York, NY, 353-356. 2007

[43] J. C. Carletta, “Assessing agreement on classification tasks: the kappa statistic”. Computational Linguistics, 22(2), 249-254. 1996.

[44] D’Mello S.K., Craig S.D., Gholson B., Franklin S., Picard R.W. & Graesser A.C.: Integrating Affect Sensors in an Intelligent Tutoring System. In Proceedings of the Workshop on Affective Interactions: The Computer in the Affective Loop Workshop, International Conference on Intelligent User Interfaces, 2005, 7-13.

A Flexible Control Architecture for Extended Autonomy of Robotic Assistants

Christopher King, Xavier Palathingal, Monica Nicolescu, Mircea Nicolescu

***Abstract*—A major challenge in deploying robots into the real world is the design of an architectural framework which can provide extended, natural and effective interactions with people. Within this framework, key issues that need to be solved relate to the robots' ability to engage in interactions in a natural way, to deal with multiple users, and to be continually perceptive of their surroundings. In this paper we propose a robot control architecture that addresses these issues. Our architecture has three main key features. First, it enables the representation of complex, sequential and hierarchical robot tasks, typically needed for service applications, in a behavior-based framework. Second, it provides the robot with flexibility in dealing with multiple users, such as to accommodate multiple user requests and task interruptions, over extended periods. Third, through its visual detection mechanism, the architecture allows the robot to identify when people are requesting its interaction. We demonstrate our approach on a Pioneer 3DX mobile robot, performing service tasks in a real-world environment.**

***Index Terms*—Human-robot interaction, behavioral robotics, personal robots.**

I. INTRODUCTION

A MAJOR challenge in designing robots for service or assistive applications is to enable natural and accessible interaction between robots and non-technical users, while ensuring extended, robust performance in complex, uncertain, dynamic human environments [14]. While significant advances have been made in increasing the complexity of tasks that robots can perform, a limitation that prevents robots from operating outside of the lab is that they lack the perceptual abilities to perform in real-world applications. In most cases, robots are programmed to execute a single task, which is switched on/off by a programmer. Robots typically perform these tasks "blindly", being generally unaware of other people and robots in the environment. In addition, such robots are limited to performing the single task that the programmer had requested. We propose a control architecture that introduces a level of flexibility and perceptual ability that allows robots to overcome traditional limitations and operate in more dynamic settings. Our architecture equips robots with the ability to monitor their surroundings and detect when other social agents require their interaction. Our architecture provides the means for long-term autonomy by enabling robots to manage a large repertoire of tasks over extended periods. Finally, our system is designed for realistic assistive applications, where multiple people are simultaneously competing for the robots assistance.

E-mail: cjking|xavier|monica|mircea@cse.unr.edu
Computer Science and Engineering Department
University of Nevada, Reno.

Vision-based perception provides the richest information required for effective human-robot interaction. In particular, the ability to distinguish between different people and to identify basic human postures is essential for the success of the interaction. While significant work has been done in these areas [2], [21], robust recognition routines are frequently too computationally expensive to be run in real time on a robot, especially when a robot must identify multiple object patterns and in a dynamic, real-world environment. In this paper, we describe a real-time identification and posture recognition algorithm and we demonstrate its use in interactive experiments using a mobile robot.

The contribution of this paper is a framework that addresses three key issues for human-robot interaction in the context of service applications: 1) complexity and robustness of task representations, 2) long term interactions with the environment and other agents, and 3) detection and recognition of multiple users.

The remaining of the paper is structured as follows: Section II gives a discussion of related work. Section III presents our interactive framework, Section IV describes our control architecture, Section V discusses our vision-based perceptual approach and Section VI describes the experimental setup and results. Finally, Section VII contains our conclusion.

II. RELATED WORK

The importance of developing robots that can provide useful services has been widely recognized and shown by the numerous application domains for which robots have been designed: agriculture and forestry, mining and construction [5], exploration and inspection [30], undersea applications [35], cleaning [11], education [26], search and rescue [22], space exploration [3], medicine and health [18]. In these approaches, the interaction between humans and robots is mostly a means for performing a job, in which the robots are regarded as tools that can be instrumented (i.e., tele-operated) toward achieving some desired goals.

The nature of applications for service and assistive robotics requires a different level of autonomy and interaction, in which the robot is regarded as a partner [9] to the human user. A significant effort in this area has been shown in designing entertainment and toy robots [32], [20], in which the focus is on designing robots that exhibit social, human-like characteristics: expressing or perceiving emotions, being sociable [8], establishing or maintaining relationships, and making friends [1]. For domains such as service or assistive applications [16], [27], a significant challenge is that robots not only can express but also understand this wide range of social cues. In this context, awareness of the world [10], [33] serves to achieve better interaction between humans and robots. To date, the issue of awareness for human-robot interaction has mostly been addressed as enhancing a human's awareness of the robot's activities [12]. With social interactions becoming more prevalent in the robot domain, it is important that the focus shifts toward increasing the robot's awareness of the world and humans around it. The approach we propose in this paper provides new capabilities that allow robots to perform in highly interactive environments, with numerous users, and multiple requests. We are able to achieve this interaction without resorting to complex hybrid architectures found in previous systems [6], [7]. Our system allows for the representation and execution of hierarchical tasks (typical for hybrid systems) within the framework of behavior-based systems, using a unique representation throughout the entire architecture. With our proposed architecture, a robot can provide service over extended periods (limited mostly by battery power requirements), with no intervention or help from the human designer.

III. Interactive Framework

The problem we address in this work is aimed at increasing a robot's autonomy over extended periods, and providing it with the skills needed in typical service or assistive application domains.

A service or personal robot will most likely have to perform in the presence of multiple users. This imposes constraints on the robot's behavior, as it will have to adjust its execution to accommodate several users who may solicit its attention and/or services during overlapping intervals. A typical example of such a situation is making a request to a robot while it is still working on another, previously assigned task. The robot needs to handle such situations appropriately: First, it should be detect the new call and interrupt its current activity to receive the request, it should then make appropriate action regarding which task to pursue next. The situation may further be complicated by different levels of authority existent between users, or by various priorities of the requested tasks.

Our framework for enabling this functionality consists of two computational modules for *visual detection* and *control*, linked into a unified control architecture. The role of the *visual detection* module is to identify, at any time, if any known users are attempting to interact with the robot. This module relies on postures of people that the robot is trained to recognize (as described in Section V). If any person/posture is detected, this information is transmitted to the *control module*, which takes appropriate decision on what the robot should do next. If the posture is only detected for a brief time, this represents a case in which the person was merely a passer by. However, if the posture persists in the robot's visual field, this is an indication that a person is trying to get the robot's attention. The job of the *control* module is to decide on the appropriate action to take in these circumstances.

As previously mentioned, in our system the robot is trained to recognize models of human users, consisting of different postures from different people. Currently, the robot associates each posture with a different task (robot service) that the users would like to request from the robot. Each task has an associated priority, which is either *low*, *regular* or *high*. When a posture is detected, the robot starts performing the service associated with that posture, unless the robot was already engaged in a task of higher or equal level of priority. In this situation, the robot adds the new request to a queue of tasks, and continues with its previous task execution. However, if a higher-level request is received, the robot switches to the new task, and moves the currently executing task to the queue. The robot processes the tasks from the queue based on their priority and incoming order. Our architecture provides the flexibility of using different priority queue strategies, as will be demonstrated in the experimental results. This prioritized task-switching process is typical for the types of decisions people make in their daily activities, and is also expected to occur in the service robot domain. While people perform this activity switching with great ease, the difficulty for the robots is to keep track of the status of the tasks when they are interrupted, such that the task could be resumed from the same point at a later time. This poses a significant challenge for the control architecture and the corresponding task representations. In this paper, we propose a behavior-based control architecture, which through its representations lends itself naturally to recovering from interrupted tasks, without the need to explicitly store any additional state information. This architecture is described in the next section.

IV. Control Architecture

The architecture we propose in this paper is aimed at providing an appropriate infrastructure for executing complex, sequential and hierarchical tasks, similar to what robots might have to perform in real-world applications. We base our approach on the Behavior-Based Control (BBC) paradigm, one of the most popular approaches to embedded and robotic system control. The contributions of the proposed control architecture are that **1)** It enables the use of both command *arbitration* and *fusion* within a single control representation and that **2)** it allows the encoding and robust execution of sequential and hierarchical tasks. Historically, the two main action selection mechanisms of *arbitration* and *fusion* have been mostly employed separately in robot control [28], thus limiting the range of tasks that robots can execute. By recognizing the ability of *arbitration* to encode temporal sequences and the ability of *fusion* to combine concurrently running behaviors, we merge the strengths and features of both within a unique task representation. For behavior representation we use a schema-based approach, similar to the work in [4]. This choice is essential for the purpose of our work because schemas with BBC provide a continuous encoding of behavioral responses and a uniform output in the form of vectors generated using a potential fields approach.

Our controllers (Fig. 1) are built from two components: *behavior primitives* (BPs) and *fusion primitives* (FP), which through the combination processes described below result in controllers in the form of *behavior networks* [25]. The *behavior primitives* perform a set of actions under given (relevant) environmental conditions. These primitives are meant to express the basic, general capabilities of the robot and need not be oriented to accomplishing a broad range of tasks. A *fusion primitive* encapsulates a set of multiple concurrently running primitive behaviors through linear combination of the motor commands. Each primitive behavior component brings its own contribution to the overall motor command. These contributions are weighted and fused through vector addition. For example, an *obstacle avoidance* behavior could have a higher impact than *reaching a target*, if the obstacles in the field are significantly dangerous to the robot. Alternatively, in a time constrained task, the robot could give a higher contribution to getting to the destination than to obstacles along the way. These weights affect the magnitude of the individual vectors coming from each behavior, thus generating different modalities of execution for the task.

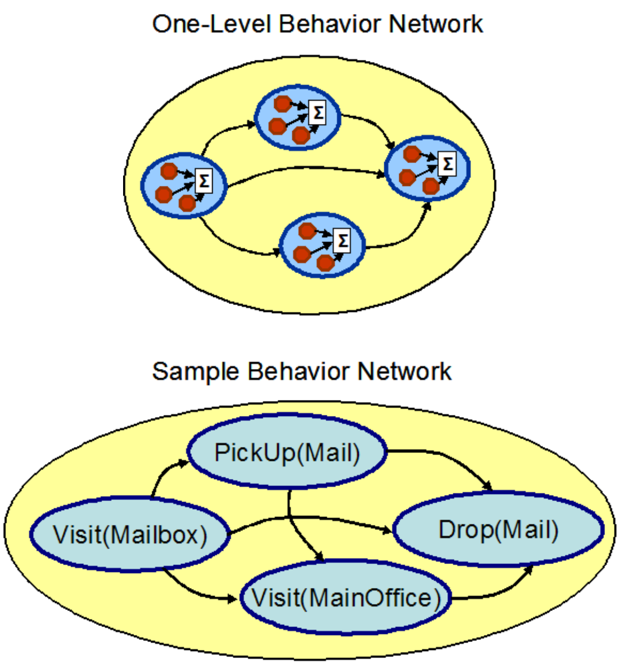


Fig. 1. Top: Representation of a generic behavior network, built of fusion primitives. Bottom: a sample behavior network. The links between fusion primitives represent task-specific precondition-postcondition dependencies.

A. Fusion Primitives

Each *fusion primitive* (Figure 2) has a representation of the goals it achieves, expressed as abstracted environmental states. The state of the goals is continuously monitored and updated from sensory data. The component *behavior primitives* receives information from the sensors, which is first used to detect if the behavior is active or not, given its preconditions. For example, in an *obstacle-avoidance* behavior, the presence or absence of an obstacle is abstracted from the range-finder information. If an obstacle is present, the precondition is met and the behavior is active. Otherwise, the behavior remains inactive. The *active/not active* status of all behavior primitives is encoded in a n-dimensional vector, where n is the number of BPs. This vector, which we call a *behavior applicability condition* (BAC), contains for each behavior a 1 or a 0, depending on whether the behavior is active or not. For a given set of n *primitive behaviors*, theoretically there could be 2^n combinations representing whether the n behaviors are active or not, based on their pre-conditions. Practically, this number is much smaller, due to the fact that some behaviors are triggered by similar environmental conditions (such as the presence of an obstacle, for example), and thus some combinations are impossible to achieve. For each possible BAC, the *fusion primitive* has a different set of fusion weights, which are used for behavior combination. The sets of weights for the multiple possible BACs are stored in a table, as shown in Figure 2. The index of each row in the table is the decimal equivalent of the n-bit BAC value.

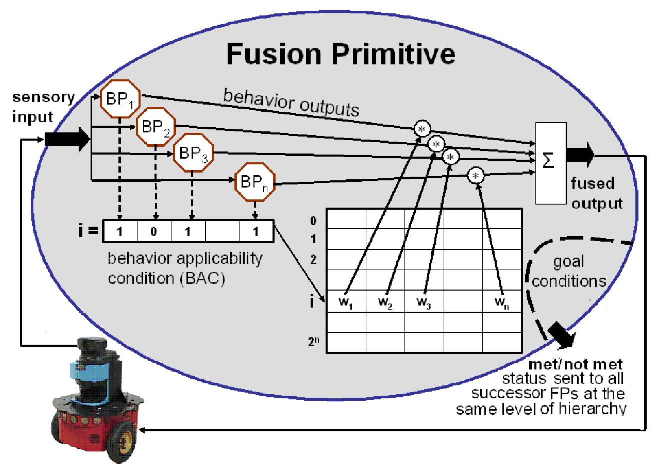


Fig. 2. Representation of a fusion primitive: Sensory input activates a corresponding set of stored weights (BAC) to fuse the underlying behavior primitives (BP).

The weights from the corresponding BAC modulate the magnitude of control vector output by the individual primitives, thus influencing the resulting command from fusion and consequently the way the robot interacts with the world. At each time step t, each *behavior primitive* BP_i provides a response output vector v_i^t, which represents a desired heading for the robot. The *fusion primitive's* output is a linear combination of the vectors $[v_1^t \cdots v_n^t]$, according to the BAC superposition weights $W^t = [w_1^t \cdots w_n^t]$:

$$V_r^t = \sum_{i=1}^{n} w_i^t v_i^t \tag{1}$$

We consider heading to be the most important consideration for behavior fusion in 2D navigation[1]. Consequently, we normalize command vectors to unit length.

The multiple BACs represent different environmental situations, since different behaviors are "applicable" in each case. The weights of behaviors within each BAC encode the mode of performing the current task given the situation and, thus

[1]Speed could easily be incorporated into our formulation. However speed is marginalized over time by the slow drive of our robots.

within each BAC, the weights of the applicable behaviors are constant. For example, for a *target reaching* task, the robot could behave under the influence of *corridor-follow, target-follow* and *avoid-obstacle* behaviors if in the presence of obstacle, but would behave only under the influence of *target-follow* if in an open space.

Inferring the fusion weights is a challenging task that would normally require time-consuming fine-tuning. In a previous work, we developed a method that allows weights to be learned through human-provided demonstration [23]. A controller, using weights evolved in this manner, was shown to be sufficiently robust to handle complex environments. Using this method, we trained a single controller to drive the robot for all our experiments.

B. Hierarchical Task Representations

With fusion primitives alone, a controller can only encode *flat representations* of tasks involving sequencing of fusion primitives. While such an architecture is expressive and flexible, it does not have the modularity needed when new, more complex tasks would have to be created from already existing ones. The best solution would be to specify a new task using abstractions of these existing modules, rather than combining their underlying behaviors into a larger, flat network. We enable this higher-level of representation by grouping fusion primitives into *behavior networks* [19], [25]. Behavior networks can be nested, allowing for the construction of hierarchical representations of robot tasks. In these networks, the links between components represent task-specific precondition-postcondition dependencies. These links provide a simple and natural way of representing complex sequences of activities and also of hierarchically structured tasks (Figure 3).

We use the term *metabehavior* to describe both fusion primitives and nodes of a behavior network in that both have similar functions in the network. Each metabehavior encapsulates information about the behavior's preconditions and its goals (postconditions). These conditions are continuously monitored whenever the behavior is active, in order to ensure the proper execution of the task. The postconditions of a behavior network node will be true when the execution of the subnetwork it represents is finished. The only difference between a behavior network node and a fusion primitive is that it activates underlying metabehaviors, while a fusion primitive activates only its component primitive behaviors. Thus, when a metabehavior is not active, all subordinate metabehaviors are disabled, and therefore can be regarded as not relevant to the task. When a behavior network node becomes active, all its underlying components are enabled, and the subnetwork becomes the current "network" that is being executed. When the execution of the subnetwork finishes, the behavior network node updates its goal status. Since the successor behaviors continuously check this status, they will detect the achievement of that goal and the execution continues with the new network node. To perform tasks encoded with this representation, the robot starts by activating the metabehavior at the topmost level in the task. The execution of the task's steps proceeds as described above.

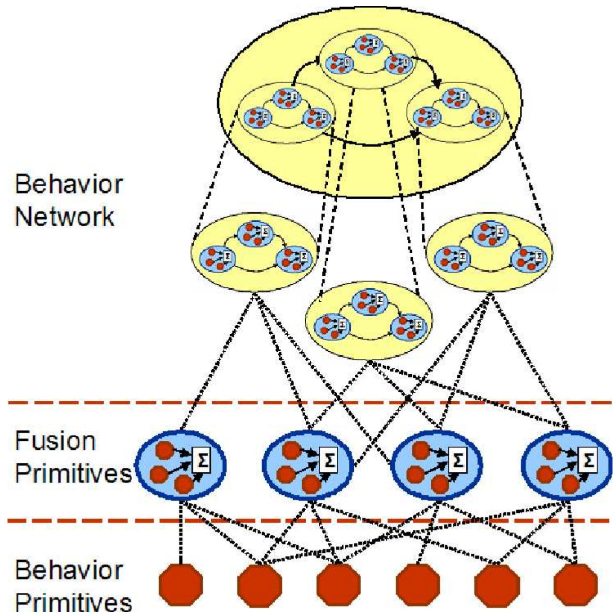


Fig. 3. A generic hierarchical task representation. The links between fusion primitives in the behavior network represent task-specific precondition-postcondition dependencies.

In this architecture, using the links as task-specific activation conditions enables the reusability of behaviors and run-time reconfiguration of robot tasks. The behavior network representation has the advantage of being adaptive to environmental changes, whether they be favorable (achieving the goals of some of the behaviors, without them being actually executed) or unfavorable (undoing some of the already achieved goals). Since the pre and post-conditions of behaviors are continuously monitored, the system executes the behavior that should be active according to the current environmental state, thus providing the robot a sense of "awareness" about its progress in the task. With these capabilities, the architecture enables the robot to keep track of the completed parts of the task, which allows dealing with task interruptions without any additional modifications. When a task is interrupted, its execution is suspended, but the behaviors preserve the current status of execution. When the task is resumed, the information implicitly stored in the behavior network controller enables the robot to continue the task from the point where it was interrupted. The behaviors' continuous grounding in sensory information allows the robot to correctly perform the task, even if the environmental conditions have changed since the task was suspended.

V. Vision-Based Human Posture Recognition

The role of the *visual awareness* module is to provide the robot with the capability of detecting the presence of humans that might be interested in interacting with the robot. Toward this end, we developed visual capabilities that can recognize human postures that are likely to be relevant to the robot-human interaction. The postures are described below and examples are shown in Fig. 4 (first row).

The Standing Posture – The most obvious posture to recognize as it is displayed frequently and is often an indication that a human is on the move or engaging in a task.

Arms-Up Posture – Humans learn at a young age that they can attract another's attention by raising their hand and a robot should respond accordingly.

Kneeling Posture – Since many robots are significantly smaller than humans are, one must crouch or kneel to pass an object to, or take an object from the robot's gripper. A robot should therefore recognize a crouching human and be able to determine if the human is holding an object.

Object Posture – Held-objects were trained independently from the human. This increases model robustness and allows the robot to orient itself toward the object.

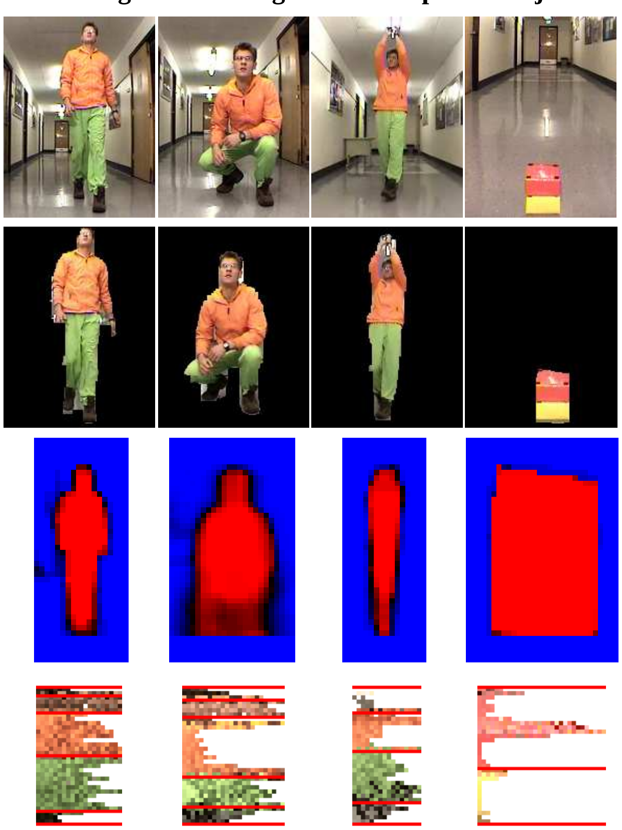


Fig. 4. Set of postures. First row: original images. Second row: detected foreground. Third row: shape model. Fourth row: color model.

A. Related Work in Visual Identification / Tracking

The identification and tracking of objects in a video feed is reasonably easy when a relatively static background can be maintained. In the simplest case, background pixels are modeled using a single video-frame, or they can be represented using Gaussian [31] or non-parametric distributions [13]. Models can be adaptive to slowly changing conditions [13], [24], [34], and even robust to smooth and linear camera movements [17]. In all these cases, pixels in subsequent frames are compared to the background model. If the pixel features (e.g., color, texture, motion) are inconsistent with the model, they are grouped and segmented as foreground.

Despite the success of background modeling techniques, they are unsuitable for use on a mobile robot in an uncontrolled environment. Camera movement is usually too complex to be stabilized by motion-modeling algorithms and the limitless variability of the shape, color, and texture of background features precludes the use of standard feature-based background models. Consequently, robotics applications typically use foreground-modeling techniques, which use object features to identify and track the object. Foreground modeling traditionally requires an offline acquisition period where values in the foreground models are assigned or trained. The robot is then switched to a tracking phase, where it searches each incoming image for a region that is sufficiently similar to the model.

A foreground modeling technique commonly used in robotic applications models an object as a colored blob. During acquisition, users must manually select a region of interest in the image. This process can be repeated on the same object under different lighting conditions and the algorithm will assign a range of values to represent the region. Schlegel *et al.* [29] automates this process by requiring users to stand directly in front of the robot's camera for an 'introduction'. The system then generates a two-dimensional color histogram (in red-green chrominance space) describing a rectangular region on the user's chest. Though this approach offers some convenience, it requires users to stand in a specific location during training and the resulting model can only be used to track colors corresponding to the user's shirt.

B. Overview of Approach

Our proposed method improves the convenience and speed of previous techniques, and is particularly suitable for robotic applications. Unlike [29], the training stage is fully automated, by using a background modeling technique to segment the person from the background. This allows the subject to move freely during the training stage. We also propose improvements to Schlegel's color modeling approach [29]. Instead of modeling a single region of color within the object, we identify and model multiple color-regions. Each region is modeled in three-dimensional RGB color-space, and is represented as a mixture of Gaussians. For increased robustness, our model also incorporates shape information without sacrificing speed.

The following sections describe how models are generated during training, how these models are used to locate and track users and their postures, and presents some implementation optimizations that allow for real time operation, even while searching for multiple models.

C. Training

Each posture from each person is separately trained during an off-line acquisition process from a stationary robot. Currently, training requires users to interact with a GUI. However, it would be relatively straight forward to automate the process in the future. Training requires a minimum of 5 seconds per posture and continues until the model stabilizes.

Before the object can be modeled, it must be segmented from the image. Segmentation is accomplished using an adaptive background modeling technique similar to the one described in [24]. Each pixel of the background is modeled as a Gaussian distribution in the RGB color space, with the mean

(μ_r,μ_g,μ_b) and standard deviation (σ_r,σ_g,σ_b). For each new pixel (x_r,x_g,x_b), the Gaussians are updated using a learning rate α, as follows:

$$\mu_i = \alpha x_i + (1-\alpha)\mu_i \tag{2}$$

$$\sigma_i^2 = max(\sigma_{min}^2, \alpha(x_i - \mu_i)^2 + (1-\alpha)\sigma_i^2) \tag{3}$$

where *i = r, g, b*.

The σ_{min}^2 term is introduced to prevent the variance from decreasing below a minimum value when the background remains constant for a long period.

Object segmentation is accomplished by comparing new pixels to the background model. A pixel is labeled as part of the foreground object if, for any value:

$$(x_i - \mu_i)^2 > (2\sigma_i)^2 \tag{4}$$

where *i = r, g, b*.

Segmented pixels are grouped together as connected-components to form a blob corresponding to the target object. Examples of a segmented image are shown in Fig. 4 (second row). As described in the next sub-sections, the segmented objects from each frame are combined to produce shape and color models.

1) Modeling Shape: Despite the fact that a human silhouette can be highly variable, there is enough regularity to warrant the inclusion of a shape-based model. The segmented foreground region from each frame is first normalized in terms of position and height, then quantized into a 32×w array of square blocks, where w is a function of the object's height to width ratio.

A map is then generated that contains (for each block) the likelihood of that block being a part of the foreground. Given N training frames, the probability at each block i is:

$$p_{shape}(i) = \frac{1}{N}\sum_{k=1}^{N} fg_k(i) \tag{5}$$

where $fg_k(i)$ is 1 if block i belongs to the foreground in frame k, and 0 if it belongs to the background.

High values in this map thus correspond to regions that are likely to be foreground and low values correspond to likely background regions. An illustration is provided in Fig. 4 (third row), where bright red regions correspond to foreground, blue regions correspond to background and black regions do not strongly correspond to either region.

2) Modeling Color: A common characteristic of human figures is that color remains relatively constant in the horizontal direction, while demonstrating more variability vertically. Variability usually occurs at the transitions between the hair and face, face and shirt, shirt and pants, and pants and shoes. It should also be noted that the relative size and location of these regions remain reasonably consistent even as a human moves.

To exploit the natural grouping of colors, our approach divides a target object into a vertical stack of horizontal color bands. We use 32 bands to represent our models. This number was selected because it provides sufficient resolution for representing the object, and because it allows for certain optimizations on a 32-bit system. Each band is modeled separately using a mixture of Gaussians.

During training, the foreground region is normalized in terms of height and quantized into the 32 bands. The pixel-values corresponding to each band are then accumulated into a histogram in RGB color space. At the end of the training period, the histogram is modeled as a mixture of Gaussians.

To produce a more robust representation of the object, adjacent bands are merged if their color distributions are sufficiently similar. Grouping begins with the most similar bands and continues with progressively less similar bands until the model is reduced to between one and six regions. Fig. 4 (forth row) shows the color distribution of each object. Colors are arranged so that the most dominant values are shown on the left and the least dominant on the right. The red lines delineate the regions after merging.

The resulting model contains information about the color composition, vertical location, and size of the prominent regions of color within an object and can be used for the detection and tracking of the object in a video sequence.

D. Detection and Tracking

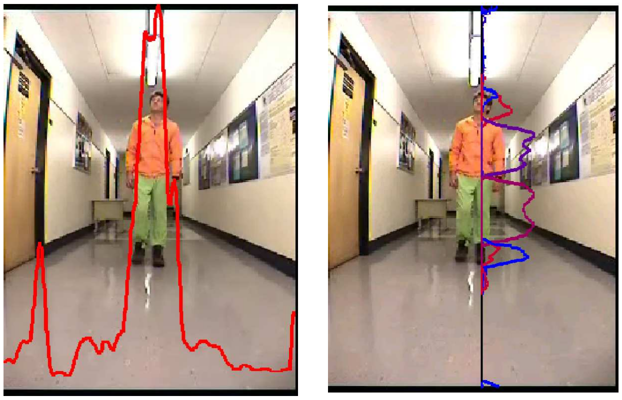
Since humans tend to assume an upright posture, they will usually occupy a larger proportion of an image in the vertical direction than they will in the horizontal direction. This property simplifies an object search because it allows promising x-axis locations to be identified before considering y-axis locations.

Every pixel in the image is assigned a probability, which represents the likelihood that that pixel color is present in the foreground object. Pixels with color values matching the most prominent colors in the target model are assigned high probabilities, while colors not found in the model are assigned a probability of zero. For a given model, the probability that pixel i with color (x_r,x_g,x_b) belongs to the model is determined using all Gaussians in all bands of the color model:

$$p_{color}(i) = \frac{1}{N_{bands}}\sum_{bands} \frac{e^{-\left(\frac{(x_r-\mu_r)^2}{2\sigma_r^2}+\frac{(x_g-\mu_g)^2}{2\sigma_g^2}+\frac{(x_b-\mu_b)^2}{2\sigma_b^2}\right)}}{\left(\sqrt{2\pi}\sigma_r\right)\left(\sqrt{2\pi}\sigma_g\right)\left(\sqrt{2\pi}\sigma_b\right)} \tag{6}$$

The resulting probability values are then summed for every column of pixels to form a probability distribution with respect to the x-axis. The most prominent local maxima in this distribution are identified as promising x-axis locations. An example of such a probability mapping is shown in Fig. 5(a).

For each x-axis candidate, a 16-pixel wide column is defined, centered at the x-location. Pixels in every row of the column are assigned probability values, which represent the likelihood that the pixel's color is present in the corresponding object-color-region. Probabilities are determined as in Eq. 6, but with the sum computed over bands in that region. Probabilities are summed for every row, to find the y-axis distribution.



a) X-axis candidates b) Region probabilities

Fig. 5. Person detection and tracking.

Fig. 5(b) shows a model, which has been divided into four prominent regions. Probability mappings for each region are superimposed on the image.

As seen in Fig. 5(b), the probability maps tend to be high within the areas of their corresponding colors and the intersection between two probability mappings offer a good estimation of the border between adjacent colors. This effectively determines the location and vertical size of each region of color within an object.

In order to determine a final measure of similarity, the object-shape probability map is incorporated. As described previously, certain blocks of the shape-map will have a high probability of falling on the figure while other areas of the map (typically towards margins) will have a low probability. The shape-based probabilities are used to weight the color-based probabilities for each region, in order to produce a final similarity score.

E. Efficiency

It should be noted that although the model describing each object can comprise as many as six different color regions, each containing several Gaussian distributions, our implementation executes a one-time preprocessing step that compiles the Gaussians into a 3D array indexed by (R,G,B). With a single reference to this array, a probability measure can be obtained to determine the likelihood that that pixel is part of an object. If the probability of being present in an object was greater than zero, up to 6 additional arrays can be referenced to determine the probability that the pixel is contained in the object's sub-regions. These optimizations allow the tracking to be performed in real-time (20 frames/sec), even when 15 models are involved, and on a modest 1 GHz computer. Although the probability arrays require more memory than any other data structure, the color-space is sub-sampled to minimize the demands. The current implementation quantizes the color-space into $32\times32\times32$ = 32,768 different colors, and requires a total of 64-bits to store the region and sub-region probability values. This requires about 262 KB of memory per model.

VI. Experimental Setup and Results

We validated our approach with a Pioneer 3DX mobile robot equipped with a SICK LMS-200 laser rangefinder, two rings of sonars, and a pan-tilt-zoom (PTZ) camera. For robot control we use the Player robot device interface [15]. Our validation consists of quantitative and qualitative evaluation for the *visual awareness* and the *robot control* modules.

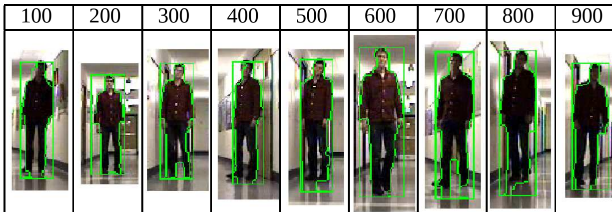
A. Validation of Visual Awareness

Tracking and distance estimation. We tested the tracking component using an experiment where the robot was programmed to pursue a person. Computation of the person's distance from the robot was based on the camera calibration procedure described in [36] and on the assumption that the floor is flat and the person's feet are always on the floor.

For this trial, the robot accurately pursued a human-target, who alternated between forward and backward movement through a 100 meter-long hallway. Frames from the robot's camera are shown in Table I, where the green rectangle and the green outline have been generated by the detection and tracking module. In order to assess the accuracy of distance estimation, the computed positions at each displayed frame are shown together with those obtained from the robot's laser rangefinder (ground truth).

It is worth emphasizing that the experiment illustrates the ability to perform real-time tracking and distance estimation for a moving target while the robot (and its camera) is also moving.

TABLE I
Tracking distance estimation.

100	200	300	400	500	600	700	800	900
								
7.3	8.3	7.3	7.3	6.4	5.8	5.8	5.8	7.0
7.7	8.2	7.3	7.3	6.8	5.6	6.1	6.3	6.6

Top Row: Frame number. *2nd row*: Frame image. *3rd row:* Estimated distances (meters). *4th row*: Ground truth (laser) distances.

Posture recognition – qualitative validation. Fig. 6 shows the recognition of postures in the presence of multiple persons. The system was trained on postures from 3 users (which are correctly recognized), while the 2 unknown users are (correctly) ignored. This experiment shows that the approach can robustly detect and track multiple postures from multiple users, while ignoring irrelevant persons in a possibly crowded environment.

Posture recognition – quantitative validation. To quantitatively estimate the recognition accuracy, we trained the system on 5 users with 3 postures each. After training, subjects were asked to display each posture for about 30 seconds, while they moved through the camera's field of view. Measurement of correct posture frequency commenced 10 frames after each change in posture and was continued for 200 frames. In this

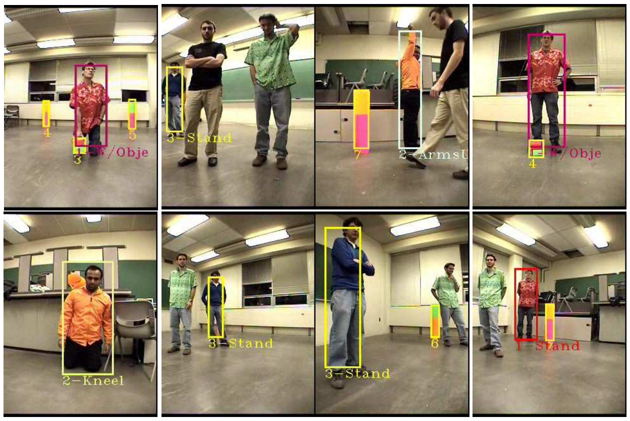


Fig. 6. Posture recognition – qualitative validation. Modeled users: red, orange, blue shirt. Unknown users: black, green shirt.

scenario, both training and testing was conducted from a stationary robot. The robot was not moved between training and testing periods. Table II shows the percentage of frames in which the algorithm correctly recognized the postures.

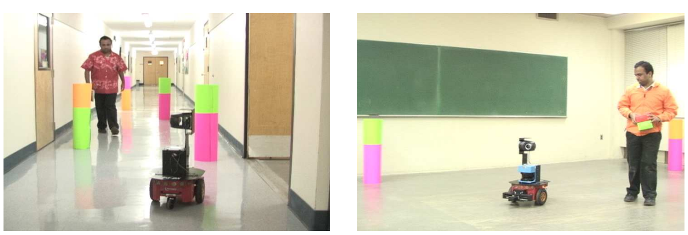
TABLE II
POSTURE RECOGNITION – QUANTITATIVE VALIDATION

	User1	**User2**	**User3**	**User4**	**User5**
Standing	92.5%	91%	99.5%	100%	100%
Kneeling	97%	98%	99%	100%	100%
Arms-up	95.5%	100%	100%	99%	100%

Postures were trained and then tested for 200 frames. Table displays percentage of frames that contained the correctly identified posture.

B. Validation of Robot Control

We performed the robot control experiments in a classroom and an office building setup, as shown in Fig. 7. In these experiments two different users interacted with the robot. For each user, the robot was trained to detect the following postures: *standing*, *arms-up*, *kneeling (with object)*, and *kneeling (without object)*, using the method described in Section V.



Office building Classroom

Fig. 7. Experimental setup for real-world environment

The robot's set of behavior primitives consists of: laser obstacle avoidance, attraction to a goal object, attraction to unoccupied space, attraction to walls, rear sonar obstacle avoidance, tangent wall follow, circular avoid, and pick up and drop objects. The behaviors produce a motor command output in the form of a vector in the robot's coordinate system. With these behaviors we created a set of fusion primitives and task controllers, which constitute our robot's repertoire of services. As mentioned previously, we used the method described in

TABLE III
ASSOCIATION BETWEEN USER POSTURES AND TASKS.

Posture ID	**Kneeling**	**Arms Up**	**Kneeling with object**
User 1 Task Code (Priority) Sequence	T1 (Medium) Drop object → Visit(Target1) → Visit(Target3) → Visit(Target2) → Visit(Target3) →	T2 (High) Visit(Target3) → Visit(Target1) → Visit(Target3) →	T3 (Medium) Take object → Visit(Target3) → Visit(Target2) → Visit(Target3) →
User 2 Task Code (Priority) Sequence	T4 (High) Drop object → Visit(Target1) → Visit(Target3) → Visit(Target2) →	T5 (Medium) Visit(Target2) → Visit(Target3) → Visit(Target1) →	T6 (High) Take object → Visit(Target1) → Visit(Target3) → Visit(Target2) →

The identification of postures triggers the robot to visit a series of targets as outlined in this table. Priority levels dictate which sequences are completed first.

[23] to train the robot's controller. The robot's tasks consist of a series of target reaching and object transport duties. To complete a task, a robot must visually locate and drive to all targets in the correct succession. We selected these duties because they are similar to what a service delivery robot might encounter and because they can be easily achieved by our current platform, though our approach should work well with other duties or platforms. Each of these tasks has a given priority and is associated with one of the users' posture, as shown in Table III. The *visit target* component of each task is a metabehavior, whose goals are achieved when the robot is at a specified distance with respect to the target.

In addition to the above tasks, the robot is equipped with a wandering task (T0), which has a *low* priority and is executed as long as the robot has no requests to service. The standing posture is not associated with any task, but rather serves as a trigger from the *visual detection* module that a user is in vicinity.

In our experiments the two users requested services from the robot over an extended duration, in order to demonstrate the main features of our approach: 1) ability to detect the presence of multiple people, 2) ability to handle multiple requests, 3) ability to handle task interruptions and 4) extended robot autonomy.

The behavior of the robot equipped with these capabilities follows a natural type of social interaction. Once the *detection* and *control* modules are started, our robot is in complete control of its own capabilities and decides for itself what activities it needs to perform. Immediately upon starting, the robot begins wandering, waiting for any requests for services from human users. If the robot detects a user (through his/her standing posture), the robot briefly interrupts its task, and slows down until the user approaches it at a given distance. If within several seconds no new postures are detected (i.e., no requests from the user), the robot resumes its task, ignoring that user for some predefined period, unless the user later displays a non-standing posture. This later step is needed in order to avoid infinite loops of attending to a passer-by user. In the case when the user displays a non-standing posture, the

TABLE IV
SEQUENCE OF TASK REQUESTS FOR THE FIRST SCENARIO.

Posture detected	Request	Robot action	Current task	Task queue
None	None	Robot starts off wandering	T0	NULL
User1 kneeling	T1	Switch to T1	T1	T0
None	None	Robot finishes T1, switches to T0	T0	NULL
User1 w/ object	T3	Switch to T3	T3	T0
User1 arms up	T2	Switch to T2	T2	T3, T0
User2 kneeling	T4	No switch, continues with T2	T2	T4, T3, T0
None	None	Robot finishes T2, switches to T4	T4	T3, T0
None	None	Robot finishes T4, switches to T3	T3	T0
None	None	Robot finishes T3, switches to T0	T0	NULL
User2 arms up	T5	Switch to T5	T5	T0
User2 w/ object	T6	Switch to T6	T6	T5, T0
User1 kneeling	T1	No switch, continues with T6	T6	T1, T5, T0
None	None	Robot finishes T6, switches to T1	T1	T5, T0
None	None	Robot finishes T1, switches to T5	T5	T0
None	None	Robot finishes T5, switches to T0	T0	NULL

robot turns and approaches the person (if it has not done so already). If the posture is detected for more than a few seconds, this is an indication that a new task has been requested. To avoid multiple detections, the robot ignores a person that had just requested a new task for a predefined period, and also ignores requests for tasks that are either in the queue or that are currently being executed. Currently, the robot produces various motor sounds to provide users with feedback that a task request was received. In future implementations we will include an auditory recognition module and speech synthesis capabilities, such that the robot could provide feedback through verbal communication.

We performed experiments for two different task requests scenarios, with each scenario repeated four times. We chose to use the same sequence of requests for each scenario, in order to establish a baseline for evaluation, both from the perspective of task execution (the *control module*) and from the perspective of the posture recognition (the *visual detection* module). For each scenario we used a different task priority scheme, as described below. In tables IV and V we show the sequence of tasks that were requested during the two scenarios, and we also indicate the correct robot action response for further validation with the experimental results.

Results from Scenario 1. Each run took approximately 20 minutes. During the course of the experiment, the robot correctly identified the postures (and thus the requests) in every case but one. It took the correct decisions regarding which tasks to perform (given priorities and incoming order) and it correctly finished executing all tasks. The only error occurred in the fourth run, in which the robot detected a request for task 4 instead of task 1. During this run, the priority method used was to process tasks with highest priority first; for tasks of equal priority, a LIFO (last-in-first-out) method was used. Fig. 8 (a) shows the order in which the robot received and serviced the requests during the four runs for scenario 1. The moments of time when the requests are received are indicated by red squares. Green squares indicate the time of task completion. Task lines that are interrupted without a marker indicate switches from the current task to a task of higher priority. Requests that are not serviced immediately after they are received belong to tasks that are of lower or equal priority with the task currently being executed. The plots show the moments of time when these tasks are serviced from the robot's task queue.

Results from Scenario 2. Each run took approximately 20 minutes. During the course of the experiment, the robot correctly identified the postures (and thus the requests), took the correct decisions regarding which tasks to perform (given priorities and incoming order) and it correctly finished executing the tasks. The only difference in execution occurred in the third run, in which the users requested task 6 before task 4. However, this being a change in scenario, the robot correctly identified and serviced the requests. During this run, the priority method used was to process tasks with highest priority first; for tasks of equal priority, a FIFO (first-in-first-out) method was used. Fig. 8 (b) shows the order in which the robot received and serviced the requests during the four runs for scenario 2. For these experiments we also recorded the progress of the robot in each task, to demonstrate that the robot is able to keep track of what parts of each task have been finalized. As a result, the figure shows that if the robot is interrupted in the middle of a task, upon resuming its execution the robot will continue from the point where the task was interrupted, instead of performing it from the start. This behavior is shown in the execution of task 3: the task is interrupted after performing its first two steps; when the robot resumes the task it only performs the last (third) step to complete it, showing that it remembered what parts of the task have already been done.

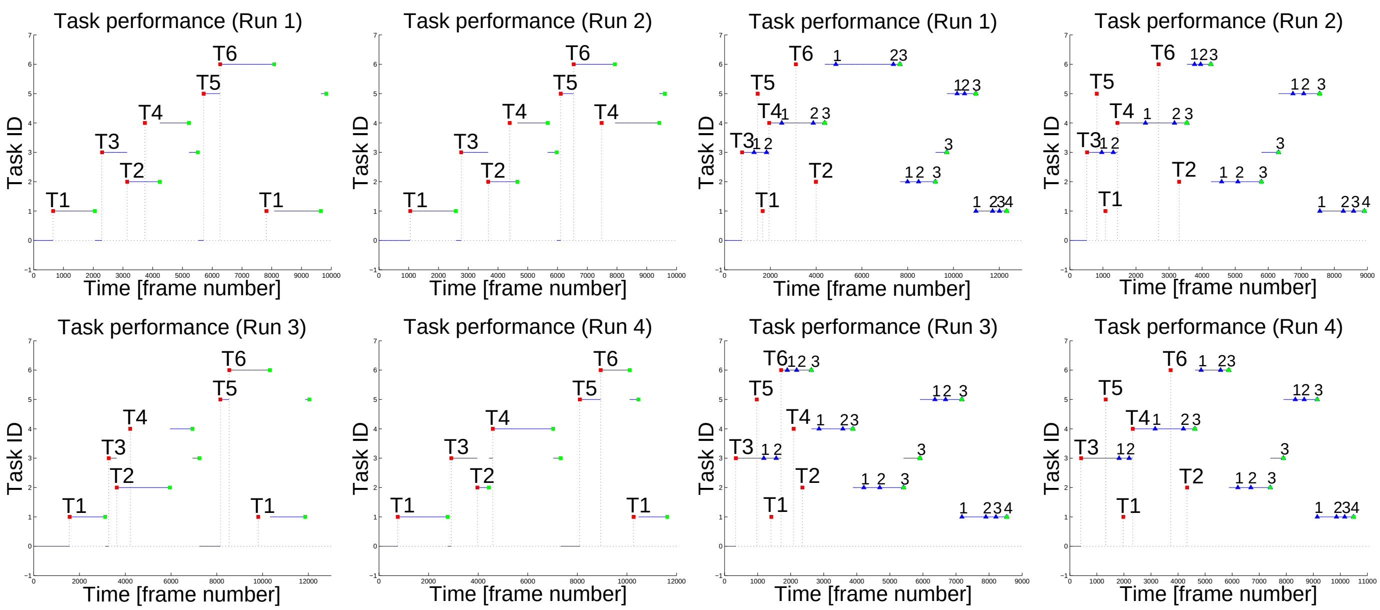
VII. CONCLUSION

In this paper we propose a framework for developing robot assistants that addresses two key issues of human-robot interaction: the ability to detect and recognize multiple users,

TABLE V

SEQUENCE OF TASK REQUESTS FOR THE SECOND SCENARIO.

Posture detected	Request	Robot action	Current task	Task queue
None	None	Robot starts off wandering	T0	NULL
User1 w/ object	T3	Switch to T3	T3	T0
User2 arms up	T5	No switch, continues with T3	T3	T5, T0
User1 kneeling	T1	No switch, continues with T3	T3	T5, T1, T0
User2 kneeling	T4	Switch to T4	T4	T3, T5, T1, T0
User2 w/ object	T6	No switch, continues with T4	T4	T6, T3, T5, T1, T0
User 1 arms up	T2	No switch, continues with T4	T4	T6, T2, T3, T5, T1, T0
None	None	Robot finishes T4, switches to T6	T6	T2, T3, T5, T1, T0
None	None	Robot finishes T6, switches to T2	T2	T3, T5, T1, T0
None	None	Robot finishes T2, switches to T3	T3	T5, T1, T0
None	None	Robot finishes T3, switches to T5	T5	T1, T0
None	None	Robot finishes T5, switches to T1	T1	T0
None	None	Robot finishes T1, switches to T0	T0	NULL



a) Scenario 1. b) Scenario 2.

Fig. 8. Robot task execution and request identification for Scenarios 1 (a) and 2 (b). Red squares indicate new requests. Green squares indicate task completion. Blue triangles indicate subtask completion. Numbers above blue triangles represent subtask ID.

and extended interaction with users and the environment. Our detection mechanism is built on visual capabilities that allow the robot to identify multiple users, with multiple postures, in real-time, in dynamic environments where both the robot and human users are moving. Extended human-robot interaction is supported by a novel control architecture which allows a robot to accommodate multiple user requests and task interruptions and it enables the representation of complex, sequential and hierarchical robot tasks. The architecture provides the robot with flexibility in dealing with multiple users, such as to accommodate multiple user requests and task interruptions, over extended periods. We validated our approach on a Pioneer 3DX mobile robot, performing service tasks in a real-world environment. Our experimental results demonstrate the robot's ability to engage in interactions in a natural way, to deal with multiple users, and to be constantly aware of their surroundings, thus advancing service robotics toward deployment of robots into the real world.

ACKNOWLEDGMENT

This work was supported in part by NSF CAREER Award IIS-0546876 to Monica Nicolescu and by the ONR Award N00014-06-1-0611.

REFERENCES

[1] Omron, is this a real cat? a robot cat you can bond with like a real pet. NeCoRo is Born, News Release, Oct 2001.

[2] A. Agarwal and B. Triggs. Recovering 3d human pose from monocular images. *PAMI*, 28(1):44–58, January 2006.

[3] R. Ambrose, H. Aldridge, R. Burridge, W. Bluethman, M. Diftler, C. Lovchik, D. Magruder, and F. Rehnmark. Robonaut: Nasa's space humanoid. *IEEE Intelligent Systems Journal*, Aug 2000.

[4] R. C. Arkin. Motor schema based navigation for a mobile robot: An approach to programming by behavior. In *IEEE Conference on Robotics and Automation, 1987*, pages 264–271, 1987.

[5] R. Barea, L. Boquete, M. Mazo, E. Lpez, and L. M. Bergasa. Electrooculgraphic guidance of a wheelchair using eye movements codification. In *International Conference on Field and Service Robots*, Helsinki, Finland, June 2001.

[6] R. Bischoff and V. Graefe. Hermesa versatile personal robotic assistant. *Proceedings of the IEEE*, pages 1759–1779, Nov 2004.

[7] M. Boada, R. Barber, and M. Salichs. Visual approach skill for a mobile robot using learning and fusion of simple skills. *Proceedings of the IEEE*, pages 157–170, 2002.

[8] C. Breazeal. Toward sociable robots. *Robotics and Autonomous Systems*, 42(3–4):167–175, 2003.

[9] C. Breazeal, A. Brooks, J. Gray, G. Hoffman, C. Kidd, H. Lee, J. Lieberman, A. Lockerd, , and D. Mulanda. Tutelage and collaboration for humanoid robots. *International Journal of Humanoid Robotics*, 1(2), 2004.

[10] C. Breazeal, A. Edsinger, , P. Fitzpatrick, and B. Scassellati. Active vision for sociable robots. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 31(5):443–453, 2001.

[11] C. Contest. First international cleaning robot contest, lausanne, switzerland, Oct 2002.

[12] J. Drury, J. Scholtz, and H. Yanco. Awareness in human-robot interaction. In *Proc., IEEE Conf. on Systems, Man and Cybernetics*, 2003.

[13] A. Elgammal, R. Duraiswami, D. Harwood, and L. Davis. Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. *Proceedings of the IEEE*, 90:1151–1163, 2002.

[14] T. Fong, I. Nourbakhsh, and K. Dautenhahn. A survey of socially interactive robots. *Robotics and Autonomous Systems*, 42:143–166, 2003.

[15] B. Gerkey, R. T. Vaughan, and A. Howard. The player/stage project: Tools for multi-robot and distributed sensor systems. In *Proc., the 11th International Conference on Advanced Robotics*, pages 317–323, 2003.

[16] H. Httenrauch, A. Green, M. Norman, L. Oestreicher, and K. S. Eklund. Involving users in the design of a mobile office robot. *IEEE Transactions on Systems, Man and Cybernetics, Part C*, 34(2):113–124, 2004.

[17] J. Kang, I. Cohen, and G. Medioni. Continuous tracking within and across camera streams. In *Computer Vision and Pattern Recognition*, pages 267–272, 2003.

[18] H. Krebs, B. Volpe, M. Aisen, and N. Hogan. Increasing productivity and quality of care: robot-aided neurorehabilitation. *Journal of Rehabilitation Research and Development*, 37(6), 2000.

[19] P. Maes. How to do the right thing. *Connection Science Journal, Special Issue on Hybrid Systems*, 1(3):291–323, 1990.

[20] F. Michaud and S. Caron. Roball, the rolling robot. *Autonomous Robots*, 12(2):211–222, 2002.

[21] G. Mori and J. Malik. Estimating human body configurations using shape context matching. In *ECCV02*, page III: 666, 2002.

[22] R. Murphy. Human-robot interaction in rescue robotics. *IEEE Systems, Man and Cybernetics Part C: Applications and Reviews, special issue on Human-Robot Interaction*, 34(2), May 2004.

[23] M. Nicolescu, C. Jenkins, and A. Olenderski. Learning behavior fusion estimation from demonstration. In *IEEE Intl. Symp. on Robot and Human Interactive Communication, (RO-MAN 2006)*, pages 340–345, Hatfield, United Kingdom, Sep 2006.

[24] M. Nicolescu, G. Medioni, and M.-S. Lee. Segmentation, tracking and interpretation using panoramic video. In *IEEE Workshop on Omnidirectional Vision*, pages 169–174, 2000.

[25] M. N. Nicolescu and M. J. Matarić. A hierarchical architecture for behavior-based robots. In *Proc., First Intl. Joint Conf. on Autonomous Agents and Multi-Agent Systems*, pages 227–233, Bologna, Italy, July 2002.

[26] I. Nourbaksh. An affective mobile robot educator with a full-time job. *Artificial Intelligence*, 114(1–2):95–124, 1999.

[27] J. Pineau, M. Montemerlo, M. Pollack, N. Roy, and S. Thrun. Towards robotic assistants in nursing homes: Challenges and results. *Special issue on Socially Interactive Robots, Robotics and Autonomous Systems*, 42(3-4):271–281, 2003.

[28] P. Pirjanian. Behavior coordination mechanisms - state-of-the-art. Tech Report IRIS-99-375, Institute for Robotics and Intelligent Systems, University of Southern California, Los Angeles, California, 1999.

[29] C. Schlegel, J. Illmann, H. Jaberg, M. Schuster, and R. Worz. Vision-based person tracking with a mobile robot. In *British Machine Vision Conference*, pages 418–427, 1998.

[30] K. U. Scholl, V. Kepplin, K. Berns, and R. Dillmann. Autonomous sewer inspection: Sensorbased navigation. In *International Conference on Field and Service Robots*, Helsinki, Finland, June 2001.

[31] C. Stauffer and W. Grimson. Learning patterns of activity using real-time tracking. *IEEE Transactions on PAMI*, pages 747–757, 2000.

[32] T. Tashima. Interactive pet robot with emotion mode. In *Proceedings of the 16th Annual Conference of the Robot Society of Japan*, 1998.

[33] A. Thomaz, G. Hoffman, and C. Breazeal. Experiments in socially guided machine learning: Understading human intent of reward/punishment. In *Human-Robot Interaction Conference*, 2006.

[34] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: Principles and practice of background maintenance. In *Intl. Conf. on Computer Vision*, pages 255–261, 1999.

[35] J. Yuh, T. Ura, and G. Bekey. *Underwater Robots*. Kluwer Press, 1996.

[36] Z. Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *Intl. Conf. on Computer Vision*, pages 666–673, 1999.