RoboSmith: Wireless Networked Architecture for Multiagent Robotic System

doi:10.3991/ijoe.v6i4.1468

D. Floroian, D. Ursutiu, F. Moldoveanu and L. Floroian
Transilvania University of Brasov, Romania

***Abstract*—In this paper it is presented an architecture of a flexible mini robot in a multiagent robotic system wireless commanded. In a multiagent system the value of an individual agent is negligible since the goal of the system is essential. Thus, the agents (robots) need to be small, low cost and cooperative. RoboSmith is designed based on these conditions. The proposed architecture divides a robot into functional modules such as locomotion, control, sensors, communication, and actuation. Any mobile robot can be constructed by combining these functional modules for a specific application. Embedded software with dynamic task uploading and multi-tasking abilities is developed in order to create better interface between both robots and the command center and between the robots. The dynamic task uploading allows the robots change their behaviors in runtime. The flexibility of the robots is given by the fact that the robots can work in multiagent system, as master-slave, or hybrid mode, they can be equipped with different modules and they may be used in other applications such as mobile sensor networks remote sensing and plant monitoring.**

***Index Terms*—wireless network, intelligent agents, mobile robot, intelligent architecture.**

I. INTRODUCTION

The main goal of this work is to develop a flexible mini robot that to be used in implementation of a multiagent robotic system, and to present the software architecture. The nature of the multiagent systems (even if applied in robotics) brings some limitations and conditions to the design of a reliable platform [1, 5, 11, 13, 14]. The size, the cost, and the cooperative abilities via specific tools are some of the limitations and conditions. The size and cost limitations are closely related since smaller size means less material and thus less cost. With the advances in microelectronics fabrication technologies, the size and the cost of the chip used in the systems decreased significantly, which allows the designers to meet the cost limitation. Even though the electronic components have become smaller significantly, the mechanical parts and the battery sizes are still reasonably large. The size limitation on a mini robot is imposed by mostly the mechanical parts and the batteries [3, 6, 9]. The fundamental problem with reducing the size of the mechanical devices is that they become inefficient or not fully functional when their size is shrunken. For example, when the size of a DC motor or of a gear gets smaller, their power and their durability are reduced significantly. These limitations have determined our design.

There have been many definitions of a robot in the literature since the beginning of the robotics field in the 1940s. After they have been introduced in the factories, the robots became mobile and smaller with the progresses made in mechanical and electrical engineering fields. After the mobility of the robots was developed, the artificial intelligence field brings its contribution to robotics by making them autonomous and smarter [2, 4, 8, 12].

The multiagent systems represents a relatively new area in the computer science and a very new area in robotics, which started to be developed in 1980s but that only in the mid-1990s gained widespread interest [5, 7, 14, 16]. The multiagent systems are compositions of computing elements that possess autonomous action, and which are able to interact among themselves, not only for exchanging messages but also for a more elaborated kind of communication that resembles social activity (cooperation, coordination, negotiation, etc.).

The robot autonomy requires good communication and sensing skills. These skills can be achieved using multiagent technology, which uses agentified components. Also this problem can be approached by using conventional elements of the robot not necessarily by agentifying the whole robot [1, 12, 16].

The robots need sensors mainly for many reasons. First, the sensing is the purpose of the mission. Second, the sensing is necessary for survival in the robot's environment: e.g. to determine an obstacle on the planned path to the target. Finally, the sensing is needed to enable the robots to sense their own configurations and their relationships with the environment. Once with sensing, a robot needs to make decisions (to think) to be able to adapt to the environment and to change the environment according to the mission [4, 10, 15].

No machine can survive in an environment without proper feedback from it. The mechanical and sensor errors can easily accumulate and put the robot in a dangerous situation. Thus, cognition is essential for robots' survival. Acting is another essential requirement for the evolution of intelligence. Acting is an ability to act on the environment, to survive and accomplish the mission. Manipulation and mobility are key components of action even though they may not be necessary at the same time for small robots. Most of the time, mobility is enough for small robots to accomplish the mission. This changes the current image of a robot from "one-armed iron-laborer" to "a mobile creature" mostly moved by wheels. In recent years, special attention has been given to robots mostly inspired from the nature (e.g. from human cooperation).

The multiagent society is encouraged by the human society. The cooperation between agents let them accomplish a complex mission with their rather limited skills. This type of behavior brings the communication component to the picture because the robots can cooperate with each other only with effective communication between them. A

considerable number of papers have been devoted to these topics [1, 5, 10, 12, 13, 14, 16].

The organization of the paper is as follows. A flexible design architecture based on the multiagent technology is described in Section II. Then, in Section III, a modular mini robot is described following the concepts presented above. In Section IV is presented the robotic multiagent system, RoboSmith, which links together the mini robots. Finally, Section V gives the conclusions.

II. FLEXIBLE DESIGN ARCHITECTURE

There has been a significant amount of research in the reconfigurable, modular and flexible robotics in the recent years [4, 6, 12]. Most researches have been done on the multiple identical modules that construct a single robot. The proposed architecture has a vertical modularity based on the horizontal layers multiagent architecture in which the layers are not identical one to another. It slices a robot into functional abstract layers such as locomotion, control, sensors, communication and actuation.

These concepts avoid main disadvantages of the horizontal layering (which require control of race conditions over the actuators) because there is a single agent which controls the actuators. Any mobile robot can be constructed by combining the above layers for a specific application. A sub-module is a piece of hardware which accomplishes the functionality of an abstract functionality (and also provides the skilling for respective agent), i.e. a wireless communication sub-module for a communication layer.

In our flexible architecture, the robot can be built by combining a locomotion layer, the sensory layers, an actuator layer, and the purpose specific layers. The software flexibility is given by the multiagent technology. Fig. 1 presents the proposed hardware layers and Fig. 2 presents the software functionality [12].

In this model, each layer can be implemented by the corresponding hardware. For example, a sensor layer may be an ultrasound sensory board or a proximity sensory board, or perhaps both. The sub-modules are designed such that they have a unique signature and a standard pin connection. The sub-modules can be added at any level and the position does not affect its module's operation.

Since the layers can be combined in any order, an application specific robot can be quickly constructed. For example, if a new problem domain requires legs rather than wheels, the wheeled sub-module can be instantly swapped with the legged sub-module. Also, in the software application, the wheels agent is replaced by legs agent. This is essential for the flexibility of the applications since agents might be equipped with complementary skills instead of having the same skills.

The programming of the applications in robotics is far from standardized. The primary reason is that each robot is composed of very special hardware designed for a specific goal. The result is that the software also becomes specific. This is very convenient to the multiagent systems which promote the reengineering instead of reprogramming. Also the layers architecture can be easily implemented in the multiagent systems. The communications between agents are standardized by FIPA (Foundation of Intelligent Physical Agents) regulations. These facts make the multiagent technology to be very useful in this situation.

A layered architecture is simultaneously reactive and deliberative. The agents deliberate and make decisions based on the symbolic representation (model) they have on the external world. These agents make more effort to model the complex entities of the external world. The reactive agents suppose the existence of basic behaviors or sequences of actions that execute concurrently from the lowest level of intelligence. These behaviors are, in turn, used by more complex ones to create more complex levels of intelligence. A layered architecture contains a set of interacting layers in which some are deliberative and others are reactive. In horizontal layering the sensors are directly connected to each of existing layers, which also might drive output directly.

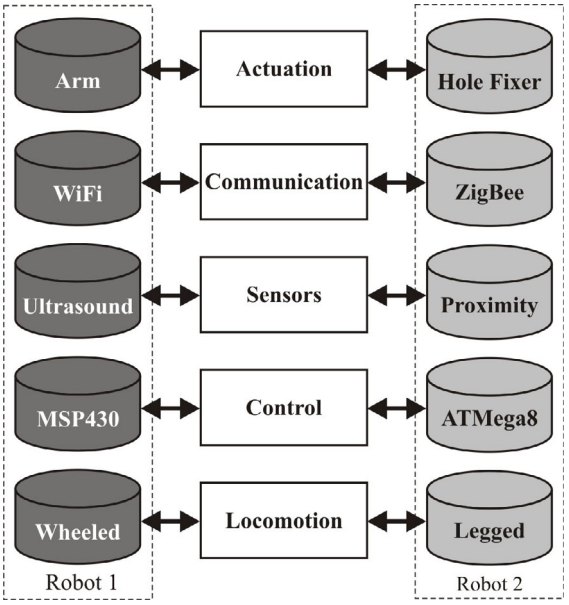

Figure 1. Flexible hardware structure.

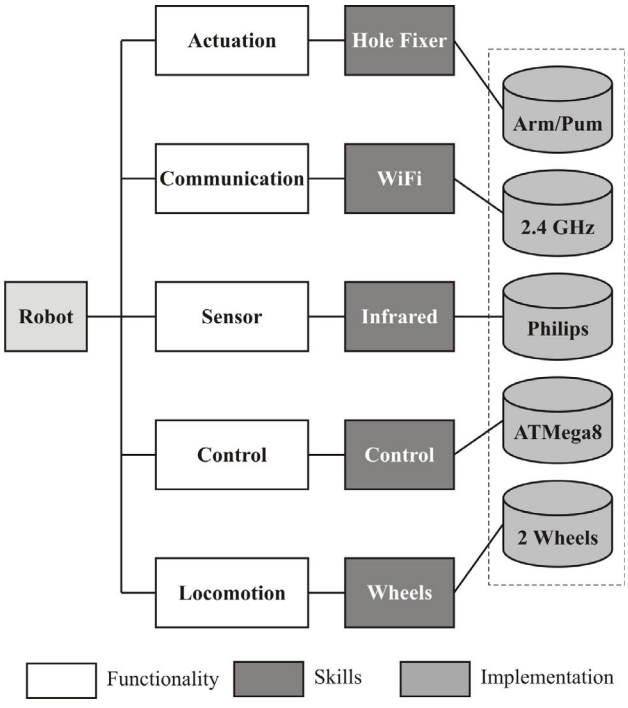

Figure 2. Flexible software structure.

The focus on this section moves from the architecture of the flexible mini robot to the architecture where a group of agents create a form. Individual agents are useless in the large majority of situations, because most scenarios involve several interacting agents. When dealing with multiagents the important aspects are to know how they communicate and how they interact. The communication and the interaction are the mechanisms that let the community of agents to have a more complex behavior than just the sum of their individual behaviors (see Fig. 3).

For the implementation of multiagent systems we will use JADE (Java Agent Development Environment) which is a middleware platform intended for the development of distributed multiagent applications based on peer-to-peer communication [17]. JADE includes Java classes to support the development of application agents and the runtime environment that provides the basic services for agents to execute. An instance of the JADE run-time is called a *container*, and the set of all containers is called the *platform*. These platforms provide the layers that hide from agents the complexity of the underlying execution system. This mechanism is depicted in Fig. 4.

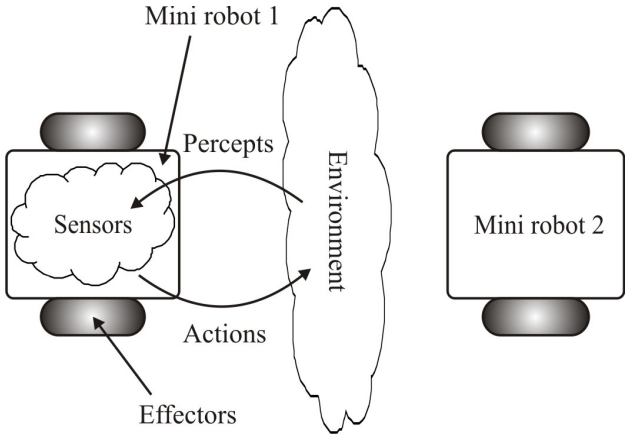

Figure 3. Agents interactions with the environment.

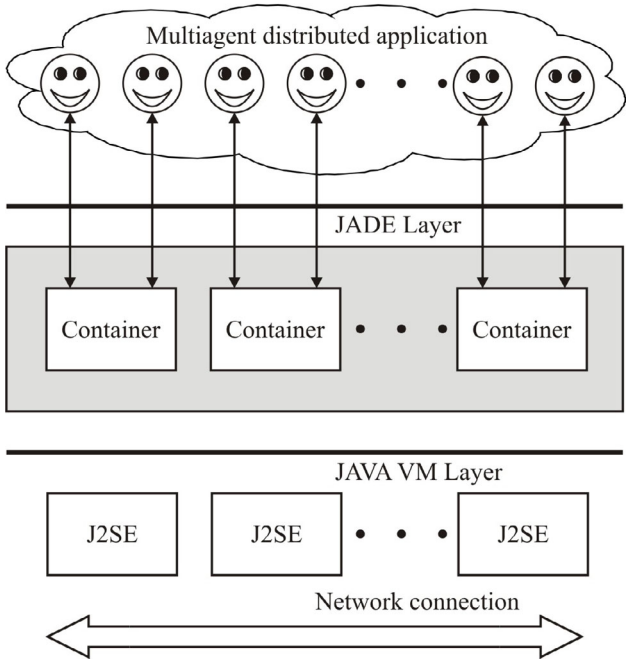

Figure 4. The JADE architecture.

III. THE MINI ROBOT

The RoboSmith's robots are flexible mini robots, which take the advantage of a layered design approach described in Section II. Even though there are five levels in the hardware and software architectures, the implementations (sub-modules) of the levels can be more than one. In addition, each level may also involve multiple closely related functionalities. The sub-modules are designed and manufactured at the Automation Laboratory. This section presents the mini robot and main sub-modules.

The locomotion module has a mechanical base and locomotion module hardware (sub-module). The base of the robot consists of an aluminum frame, two stepper motors, some gearing, two wheels and associated ball bearings, and the batteries. The base is designed by CAD tools and machined with high precision CNC machines. Fig. 5 shows the base with wheels, gears and motors. A legged version of the base is also in design process as an alternative locomotion to be used in different applications. The battery selected for the mini robot is an AA form factor NiMH rechargeable cell. Four of these cells connected in series are used in the system. The cells are nominally 1.2 volts each for 4.8 volts system voltage.

The two wheels module is very versatile and easily direction able. This mechanism allows the robot to make short turns by moving only a wheel and stopping the other. Another advantage is providing by the sensors. The time delay is not critical because, in this situation, the locomotion module has plenty of time to turn and the control module has plenty of time to make decisions.

The locomotion layer is implemented by first electronic module and the mechanical base described above. It's the most critical layer in the operation of the robot. It contains the circuit for the stepper controller, which provides the direction control for the motors and supplies the high current they require. This module is also manufactured in laboratory and includes common components. It also includes the power system, which consists of a DC-DC converter and some passive components. The power system provides +5 volts for the entire robot, and will accept from 1.5 - 15 volts on its input. This provides plenty of flexibility if a different battery system is put in place. This sub-

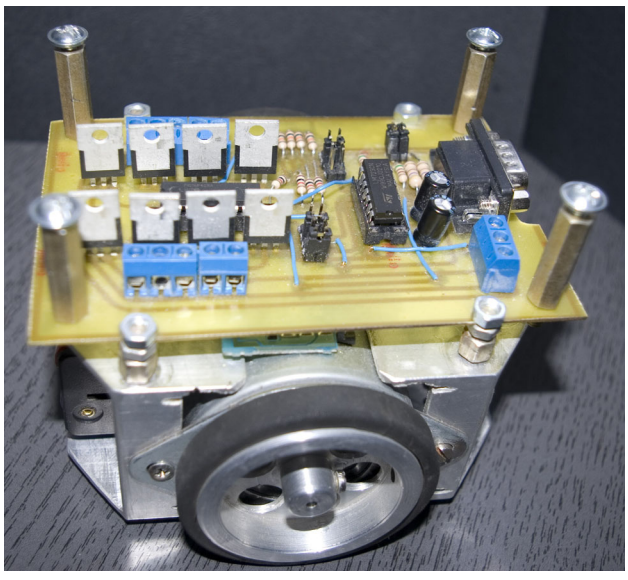

Figure 5. The base with wheels for flexible mini robot.

module also contains the charge circuit, which allows the battery to remain in the robot while it is recharged (see Fig. 6). Also by choosing the high capacity elements the robot has greater autonomy.

Over on this sub-module (as seen in Fig. 7), it is the main controller, an ATMega8 microcontroller running at 16 MHz. The 8 kB flash memory is included on the chip and also 512 B RAM and 1 kB SRAM. All other components are soldered directly to the board. Having improvements of the memory architecture, the mini robots are able to run in a flexible architecture.

At the top, the sub-module for communication layer is based on an XBee hardware board (serial version) (which is very similar to familiar ZigBee modules) [19]. This is necessary for the agent's interactions and for reporting to the main server unit which is hosted on a PC (see Fig. 8.). Another XBee module (an USB version this time) is connected to a host PC and connects the mini robot to the control program. Also many mini robots (a maximum of 16 is recommended) equipped with XBee module can be interconnected in this manner. The control program will coordinate the messages.

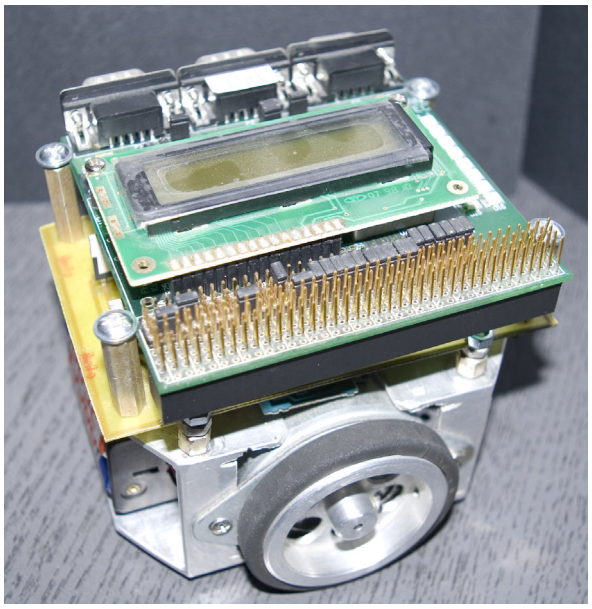

Figure 6. Implementation of locomotion layer of flexible mini robot.

IV. THE MULTIAGENT ROBOTIC SYSTEM

The RoboSmith architecture is based on a multiagent robotic system which coordinates the entire community of agents. This section presents the implementation of the RoboSmith architecture.

The RoboSmith is a networked organization of mini robots that have formed together a cooperative dynamic network to reach group benefits. RoboSmith is a society of agents (the mini robots) and therefore their interactions are at society level. The mini robots can be considered intelligent agents because they are proactive, reactive and have social ability. They are proactive since they have goal directed behavior that is seen when the layers involved participate in society with the best possible performance. Moreover, they can keep working even under the environmental coalitions. They are reactive in the sense that they react to changes in the external environment, which are "sensed" through messages. Although the RoboSmith agents are not purely reactive, the importance of messages in their behavior is so relevant that their architecture is more reactive than proactive. Finally, they have social ability because they are able to negotiate and to cooperate with the other mini robots.

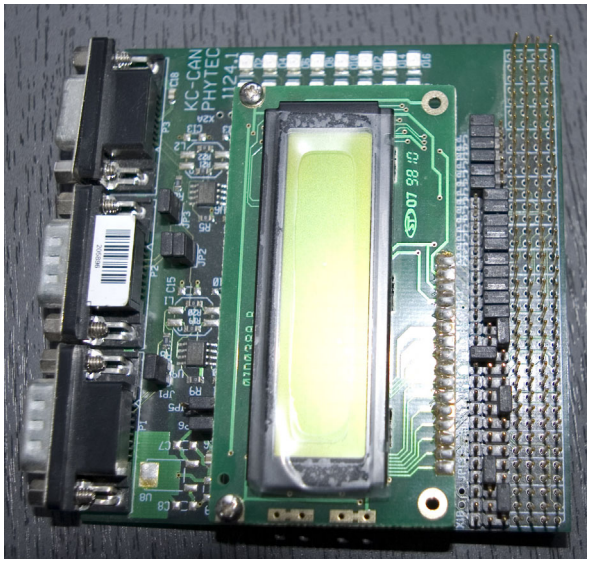

Figure 7. Mini robot's CPU.

The communication and the interaction among individuals or within some domains can only take place if there is some of the conceptualisation of these domains. To guarantee a common semantic understanding, agents must use appropriate ontology to communicate with their partners. In the case of the RoboSmith, all the mini robots need to share some basic concepts, such as skills, requests, services and agent. Therefore all agents of the proposed architecture share a basic global ontology that patterns the basic referred concepts.

For our purposes, we have adopted the description of an agent as a software program with the capabilities of sensing, computing, and networking associated with the specific skills of the mini robots described above. This implementation is made in JADE because this development tool is very versatile and could be very well integrated with others development tools (like Protégé-2000 and Java [17, 18]). Also JADE is an open source FIPA compliant Java

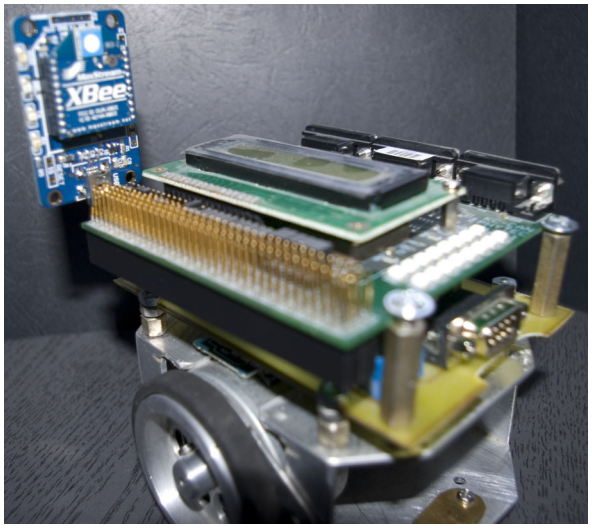

Figure 8. Implementation of hardware communication layer.

based software framework for the implementation of the multiagent systems. It simplifies the implementation of the agent communities by offering runtime and agent programming libraries, as well as tools to manage platform execution and monitoring and debugging activities. These supporting tools are themselves FIPA agents.

JADE offers simultaneously middleware for FIPA compliant multiagent systems, supporting application agents whenever they need to exploit some feature covered by the FIPA standard (message passing, agent life cycle, etc.), and a Java framework for developing FIPA compliant agent applications, making FIPA standard assets available to the programmer through Java object-oriented abstractions. The general management console for a JADE agent platform (RMA – Remote Monitoring Agent), like in Fig. 9, acquires the information about the platform and executes the GUI (Graphic User Interface) commands to modify the status of the platform (creating new agents, shutting down containers, etc) through the AMS (Agent Management System). The agent platform can be split between several hosts (provided that there is no firewall between them). The agents are implemented as one Java thread and Java events are used for effective and lightweight communication between agents on the same host. The parallel tasks can be still executed by one agent, and JADE schedules these tasks in a more efficient (and even simpler for the skilled programmer) way than the Java Virtual Machine (VM) does for threads. Several Java VM, called containers in JADE, can coexist in the same agent platform even though they are not running in the same host as the RMA agent. This means that a RMA can be used to manage a set of VMs distributed across various hosts. Each container provides a complete run time environment for the agent execution and allows several agents to concurrently execute on the same host. The DF (Directory Facilitator), AMS, and RMA agents coexist under the same container (main-container) together with the RoboSmith's agentified mini robots, as it is shown in Fig. 9.

To facilitate message reply, which, according to FIPA, must be formed taking into account a set of well-formed rules such as setting the appropriate value for the attributes *in-reply-to*, using the same *conversation-id*, etc., the method *createReply*() is defined in the class that defines the ACL (Agent Communication Language) message. Different types of primitives are also included to facilitate the implementation of content languages other than SL (Standard Languages), which is the default content language defined by FIPA for ACL messages. This facility is made with Protégé 2000 as depicted in Fig. 10.

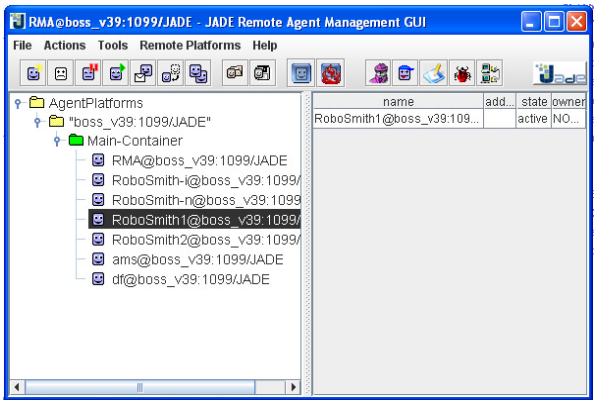

Figure 9. JADE Implementation of RoboSmith.

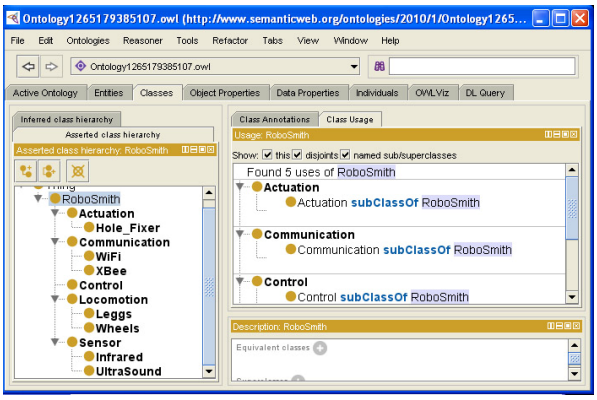

Figure 10. Defining ontologies for RoboSmith.

In Fig. 11 is presented a graphical view of ontology classes which facilitate understanding the fact that from the point of view of the programmer, a JADE agent is simply a Java class that extends the base *agent* class. It allows inheriting a basic hidden behavior (such as registration, configuration, remote management, etc.), and a basic set of methods that can be called to implement the application tasks of the agent (i.e. send/receive ACL messages).

Moreover, the user agents inherit from their Agent superclass some methods to manage agent behaviors. Also this diagram can be represented in UML (Unified Modeling Language) because behaviors are implemented as hierarchy of the classes.

The Protégé 2000 connects to RoboSmith JADE Agents by including a Protégé configuration file in the Java compiler. The RoboSmith ontology is divided in many concepts that follows the class hierarchy defined above.

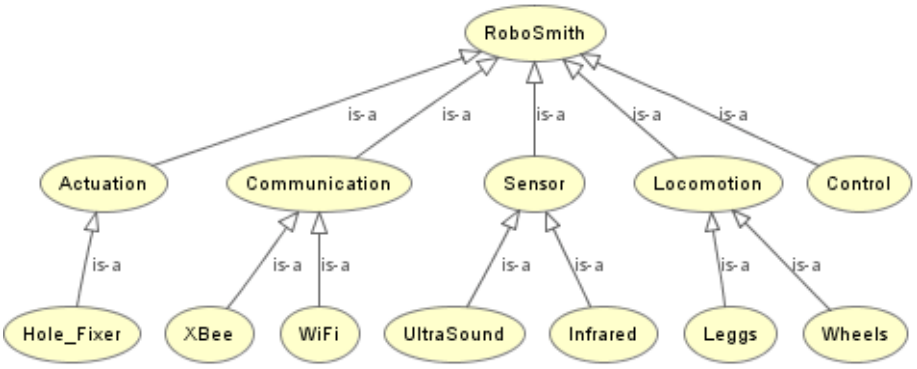

Figure 11. Classes hierarchy for RoboSmith architecture.

V. CONCLUSIONS

In this paper, the exploitation of the multiagent technology applied in flexible mini robotic system has been presented. A number of two robots were constructed and were implied in multiagent robotic system. It is possible to extend this number by adding similar robots.

The results presented confirmed the theoretical predictions made during the design phase. The proper function of the robotic system is important from a practical point of view since it provides a detailed framework about the design of the control structure and the behaviors tasks. This confirm that multiagent technology can be successfully implemented to control the robotic systems.

REFERENCES

[1] J. Barata, and L.M. Camarinha-Matos, "Multiagent Coalitions of Manufacturing Components for Shop Floor Agility – The CoBaSA Architecture", *Int. J. Networking and Virtual Organisation*, 2(1), pp.50-77, 2003. doi:10.1504/IJNVO.2003.003518

[2] B.S. Choi, J.J. Lee, "Localization for a mobile robot based on an ultrasonic sensor using dynamic obstacles", *J. Artificial Life and Robotics*, vol. 12, Springer Japan, pp. 280-283, March, 2008.

[3] V. Denneberg and P. Fromm, "OSCAR. An open software concept for autonomous robots", *Proc. of the 24th Annual Conference of the IEEE Industrial Electronics Society*, pp. 1192-1197, sep. 1998.

[4] L. Feng, J. Borenstein, and H. R. Everett, "*Where am I? Sensors and Methods for Autonomous Mobile Robot Positioning*", vol. 3, The University of Michigan, 1994, pp. 9-10.

[5] D., Floroian, *Multiagent Systems*, Cluj-Napoca (Romania), Ed. Albastra (In Romanian), 2009.

[6] W. Jueyao, Z. Xiaorui, T. Fude, Z. Tao, et al., "Design of a Modular Robotic System for Archaeological Exploration", *Int. Conf. on Robotics and Automation*, Kobe, Japan, pp. 1435-1440, 2009. doi:10.1109/ROBOT.2009.5152364

[7] T. Komatsu and N. Kuki, "Investigating the Contributing Factors to Make Users React Toward an On-screen Agent as if They are Reacting Toward a Robotic Agent", *18th IEEE Int. Symp. Robot and Human Interactive Communication*, Toyama, Japan, Sept. 2009. doi:10.1109/ROMAN.2009.5326350

[8] D., Lee, W., Chung, "Discrete-status-based Localization for Indoor Service Robots", *IEEE Trans. Ind. Electron.* Vol. 53, No. 5, Oct. 2006, pg.:1737–1746. doi:10.1109/TIE.2006.881949

[9] W.H. Lee and A.C. Sanderson, "Dynamics and distributed control of modular robotic systems", *Proc. of the 26th Annual Conference of the IEEE Industrial Electronics Society*, pp. 2479-2484, sep. 2000.

[10] W.J. Opp and F. Sahin, "An Artificial Immune System approach to Mobile Sensor Networks and Mine Detection", *IEEE SMC 2004, the Hauge*, Netherlands, oct. 2004.

[11] E., Petriu, T., Whalen, R., Abielmona, A., Stewart, "Robotic Sensor Agents: A New Generation of Intelligent Agents for Complex Environment Monitoring", *IEEE Instrum. Meas. Mag.*, Vol. 7, No.3, Sep. 2004, pg.:46–51. doi:10.1109/MIM.2004.1337913

[12] F. Sahin, "GroundScouts: Architecture for a Modular Micro Robotic Platform for Swarm Intelligence and Cooperative Robotics", *Int Conf. Syst., Man & Cyb*., 2004, pp.929-934.

[13] D. Weyns, H. V. D. Parunak, F. Michel, T. Holvoet, and J. Ferber, "Environments for Multiagent Systems State-of-the-Art and Research Challenges", *E4MAS* 2004 *Springer-Verlag*, Berlin, pp. 1–47, 2005.

[14] M., Wooldridge, N.R., Jennings, "Agent Theories, Architectures, and Languages: A Survey", Proc. *ECAI-Workshop on Agent Theories, Architectures and Languages*, Amsterdam (The Netherlands), Aug. 1994, pg.:145–160.

[15] J., Zheng, P., Lorenz, P., Dini, "Guest Editorial: Wireless Sensor Networking", *IEEE Netw.*, Vol. 20, No. 3, May/Jun. 2006, pg.:4–5. doi:10.1109/MNET.2006.1637925

[16] S.A. Suboting and Al. A. Oleinik, "Multiagent Optimization Based on the Bee-Colony Method", *Cybernetics and Systems Analysis*, Vol. 45, No. 2, 2009.

[17] ***, Java Agent Development Framework – JADE, ´http://jade.tilab.com/

[18] ***, Protégé 2000, http://protege.stanford.edu/

[19] ***, XBee, http://www.digi.com

AUTHORS

D. Floroian, D. Ursutiu, F. Moldoveanu and L. Floroian are with Transilvania University of Brasov, Romania.

Submitted October 4th, 2010. Published as resubmitted by the authors October 17th, 2010.