22

Autonomous and Intelligent Mobile Systems based on Multi-Agent Systems

Adil Sayouti, Hicham Medromi and Fouad Moutaouakil
Laboratoire d'Informatique, Systèmes et Énergies Renouvelables
Team Architecture of Systems, ENSEM, Hassan II University
BP 8118, Oasis, Casablanca
Morocco

1. Introduction

The control of current robotic systems in manufacturing industry and the service sector has remained separate and independent. In other words, these robotic systems are isolated from one another by different environments and have no effective way to communicate. This has made the current robotic systems expensive and requiring a long developing cycle, which has in turn seriously hampered the day-to-day deployment of robot technology. Therefore it is crucial to develop an integrated network environment for robotic systems based on today's Internet technology. With the rapid growth of the Internet, more and more intelligent devices or systems have been embedded into it for service, security and entertainment, including distributed computer systems, surveillance cameras, telescopes, manipulators and mobile robots. Although the notion of Internet robotics or web-based robotics is relatively new and still in its infancy, it has captured the huge interest of many researchers worldwide. Except for operating in hazardous environments that are traditional telerobotic areas, Internet robotics has opened up a completely new range of real-world applications, for example in the following fields (Le parc & al., 2008):

1. Tele-teaching: a lot of universities are using robots to teach the basics of electrical engineering. The profitability of these robots is of course really poor because they are only used a few weeks a year. Why not developing common centers where students may have access to real robots without being close to them? One of the problems of e-learning is to make practical experiments. Why not using Internet technologies to let distant students to manipulate real systems?
2. Tele-maintenance: when a company is shipping systems all over the world, it needs to technicians when one of its systems has some failures. With Internet technologies, it is now possible to make some remote diagnostics, to solve and repair some problems, to prepare the right equipment to send, etc.
3. Tele-expertise: some specific operations on robotic systems can only be made by expert. In a close future, it will become possible for experts to operate from their office a machine located somewhere in the World, just using classic web technologies.
4. Tele-production: the remote access possibilities and taking control will make work easier for remote users and will allow the performances of more tasks in the future.

The use of Internet will reduce the costs of these activities. The increase of Internet abilities in term of speed and bandwidth in the future, let us also think that the quality of the remote control and the comfort of the user will also increase. But, when developing such applications, we have to think that these activities rely all the time on an unpredictable network and that we have to build them taking into account this parameter. The organization of a mobile system - or its control architecture - determines its capacities to achieve tasks and to react to events. In this chapter we propose a remote control architecture based on multi agents systems to take into account the lack of quality of services of Internet.
This chapter is presented as follows: on the next section, we present the multi-agent approach. Then we will describe some projects of remote control of robotic systems. In section 4, we propose a system architecture that allows such a control on Internet. The state of the art of control architecture is detailed in section 5. In the section 6, we present the proposed control architecture based on multi-agents systems. Then we will describe our remote control software architecture. In section 8, we present some applications of remote control of an autonomous Lego mobile robot as illustrative examples. Finally, some conclusions and future developments are presented in section 9.

2. Multi-agent systems for autonomous control

The organization of a system - or its control architecture - determines its capacities to achieve autonomous tasks and to react to events (Novales & al., 2006). The control architecture of an autonomous mobile system must have both decision-making and reactive capabilities: situations must be anticipated and the adequate actions decided by the mobile system accordingly, tasks must be instantiated and refined at execution time according to the actual context, and the mobile system must react in a timely fashion to events. This can be defined as a rational behavior, measured by the mobile system's effectiveness and robustness in carrying out tasks.
To meet this global requirement, the control system architecture should have the following properties (Alami & al., 1998):

1. Tele-teaching: a lot of universities are using robots to teach the basics of electrical engineering. The profitability of these robots is of course really poor because they are only used a few weeks a year. Why not developing common centers where students may have access to real robots without being close to them? One of the problems of e-learning is to make practical experiments. Why not using Internet technologies to let distant students to manipulate real systems?
2. Programmability: a useful mobile system cannot be designed for a single environment or task, programmed in detail. It should be able to achieve multiple tasks described at an abstract level. The functions should be easily combined according to the task to be executed.
3. Autonomy and intelligence: the mobile system should be able to carry out its actions and to refine or modify the task and its own behavior according to the current goal and execution context as perceived.
4. Reactivity: the mobile system has to take into account events with time bounds compatible with the correct and efficient achievement of its goals (including its own safety). Consistent behavior: the reactions of the mobile system to events must be guided by the objectives of is task.

5. Robustness: the control architecture should be able to exploit the redundancy of the processing functions. Robustness will require the control to be decentralized to some extent.
6. Extensibility: integration of new functions and definition of new tasks should be easy. Learning capabilities are important to consider here: the architecture should make learning possible.

We note an interesting link between the desirable properties of intelligent control architecture for autonomous mobile systems and the behavior of agent-based systems (Ferber 1999):

1. Agent-based approaches to software and algorithm development have received a great deal of research attention in recent years and are becoming widely utilised in the construction of complex systems.
2. Agents use their own localised knowledge for decision-making (see Figure 1), supplementing this with information gained by communication with ist environment and the other agents.
3. Remaining independent of any kind of centralised control while taking a local view of decisions gives rise to a tendency for robust behavior.
4. The distributed nature of such an approach also provides a degree of tolerance to faults, both those originating in the software/hardware system itself and in the wider environment.
5. Multi-agent systems can manifest self-organization and complex behaviors even when the individual strategies of all their agents are simple.
6. Agents can share knowledge using any agreed language, within the constraints of the system's communication protocol. Example languages are Knowledge Query Manipulation Language (KQML) or FIPA's Agent Communication Language (ACL).

It is for these reasons that we consider an agent-based system to be a suitable model on which to base an intelligent control architecture for complex systems requiring a large degree of autonomy.

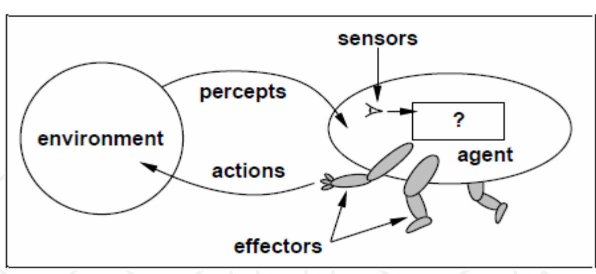


Fig. 1. Agents interact with environments through sensors and effectors

3. Remote control over Internet – state of the art

During the Nineties, several projects appeared of robotic systems control, using Internet as communication network (Goldberg & al., 2001) with various objectives.

The **Mercury Project** (Goldberg & al., 2000) is believed to be first system that allowed Internet users to remotely control robotics via Internet. This project launched the first system allows users to alter the real world. This project is initialized by an interdisciplinary team of anthropologists and computer scientists. They want to explore an area dubbed "Mercury site". Because nobody can work in this dangerous area, the remote robotics is a good choice.

The remote control of the robot is designed to excavate the surface with short burst of compressed air and then the surface is revealed and the relevant data can be collected by the anthropologists. After the success of the initial exploration, the site is open to all the researchers who are interested in having a remote control of the robot via Internet. The successes of Mercury Project is not only on its excavation purpose, but also showed the possibility of control the robot via Internet. This is the milestone on Internet telerobotics, more and more Internet telerobotics projects were launched in the later years.

Telegarden is the second Internet telerobotics from Goldberg and al. This Telegarden system allows Internet users to view and interact with a remote garden filled with living plants (see Figure 2). Users can plant, water, and monitor the progress of seedlings via the tender movements of an industrial robot arm.

Fig. 2. Telegarden

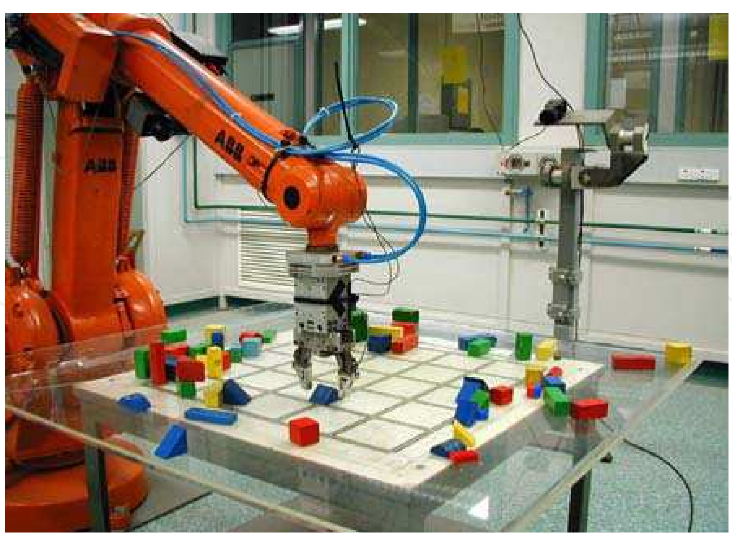


Fig. 3. Australian Telerobot on the Web

The **Australian Telerobot** on the Web project (Taylor & Taylor, 1997) is established by Taylor at The University of Western Australia. A six degree of freedom robotic arm is controlled through Taylor's Internet telerobotics system (see Figure 3). The robotic arm can

play the wooden block on a table. The user connects the server via Internet connection, and can log to the system with an identification check. This project is very successfully due to it's an interesting wooden block game as well as users can play it via Internet.
The launch of **RHINO project** (Schulz, 1997) indicates the possible potential of Internet robotics in daily life. RHINO project (see Figure 4) is initially launched for a museum of contemporary technology in Bonn Germany. Visitors of the "Deutsches Museum Bonn" will have the opportunity to be shown through the museum by a mobile robot "RHINO". RHINO can provide the user with the information they concerned as well as more information in deep upon request.

Fig. 4. RHINO robot

Fig. 5. PumaPaint interface

The research focus of **Xavier** (Simmons & Xavier, 1998) is to study the local intelligence of the robot as well as users' interface. The research team has considered the supervisory control which aims to give robot the command at a higher level. This scheme can reduce the influence of Internet time delay, but prevents more interaction between users and robot. In this situation, users can't interact with the robot immediately. Remote users need fast feedback (image) when controlling the robots on the web facing unpredictable Internet time

delay (limited bandwidth). The supervisory control can indeed reduce the bandwidth requirement, but at the same time reduces interactivity. These existed problems are highly concerned in the later years' research of Internet telerobotics.
The **PumaPaint project** (Stein, 1998) came out from the collaboration between University of Wisconsin and Wilkes University (see Figure 5).
The computer science department at UW needed a PUMA robot as undergraduate teaching resource. But if they develop and maintain the system themselves, it's money and time costly. So, they decide to share the installed one at Wilkes University. Students from University of Wisconsin can access the PUMA robot via Internet connection.
The system is developed in Java considering the cross platform advantage of java as well as the reusability of Java program. The Java Virtual Machines (JVM) is involved in the development, every machine need to install this before using the system, but the JVM is quite popular in most of web browsers.
The possibility of piloting a **Khepera robot** (Saucy & Mondada, 2000) was made available to the general public over the web in December 1996 (KhepOnTheWeb). By means of a WEB client (Netscape), the user can move forward or turn the robot and receive images of the remote environment (see Figure 6). He can also choose the point of view and receive images either from the robot's on-board camera or from a camera mounted on the ceiling.

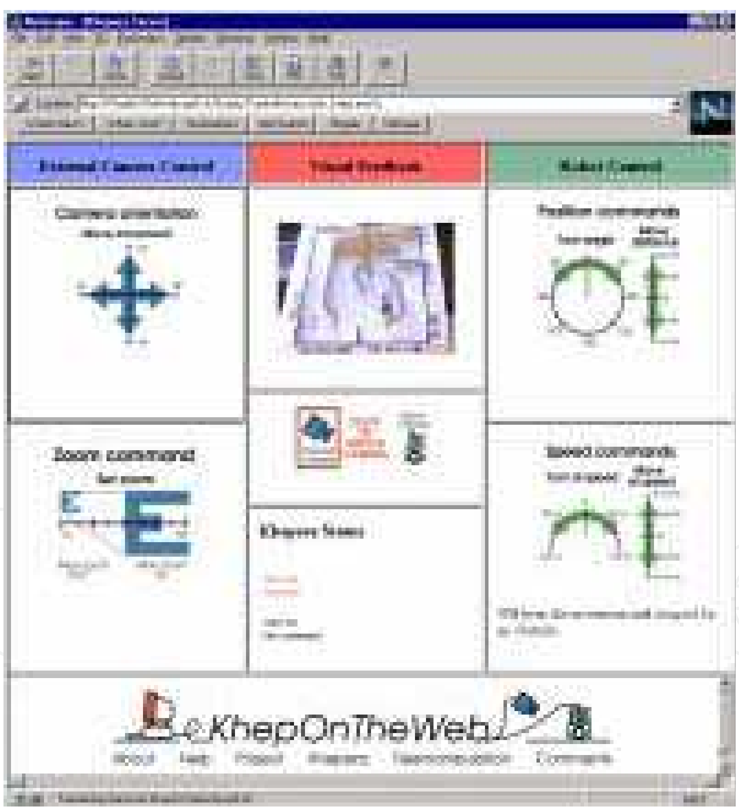


Fig. 6. The remote control interface

All these experiences are really interesting because they have treated the problem of remote control in different manners and in different contexts. To develop safe and evaluated web-based remote control, one has to take into account all of them. Nevertheless, we can notice that none of these works have tried to develop some control architecture and the unpredictable nature of the Internet is not really taking into account with all the consequences. In the next sections, we will propose a solution that consists in equipping the mobile systems with a high degree of local intelligence in order for them to autonomously handle the uncertainty in the real world and also the arbitrary network delay.

4. Proposed system architecture

With the rapid growth of the Internet there are many communications technologies available to execute requests in a networked environment. Currently the most widely used web browser is the Hypertext Transfer Protocol (HTTP). It can be executed with the Communication Gateway Interface (CGI) for remote control, which is one of methods used in many web-based telerobot systems (Kosuge & al., 1998). Through the Hyper Text Markup Language (HTML) form, a request can be passed from client to server to launch a process to perform some predetermined actions in the server. A dynamically generated HTML page will return the results to the client. But CGI has a number of limitations (Hu & al.) such as its slow response speed. Moreover, a complete HTML page must be generated with each request while the resulting page is still static. So it is not suitable for real-time remote control. In contrast, Java provides the capability to implement network connections and thus avoids the limitations of CGI. A Java applet can operate within the browser and hence is accessible by most computers on the Internet (Le parc & al., 2005). Rather than being static, a Java applet also enables an interface to dynamically change its content due to the fact that the Java applet is an executable within a web page.

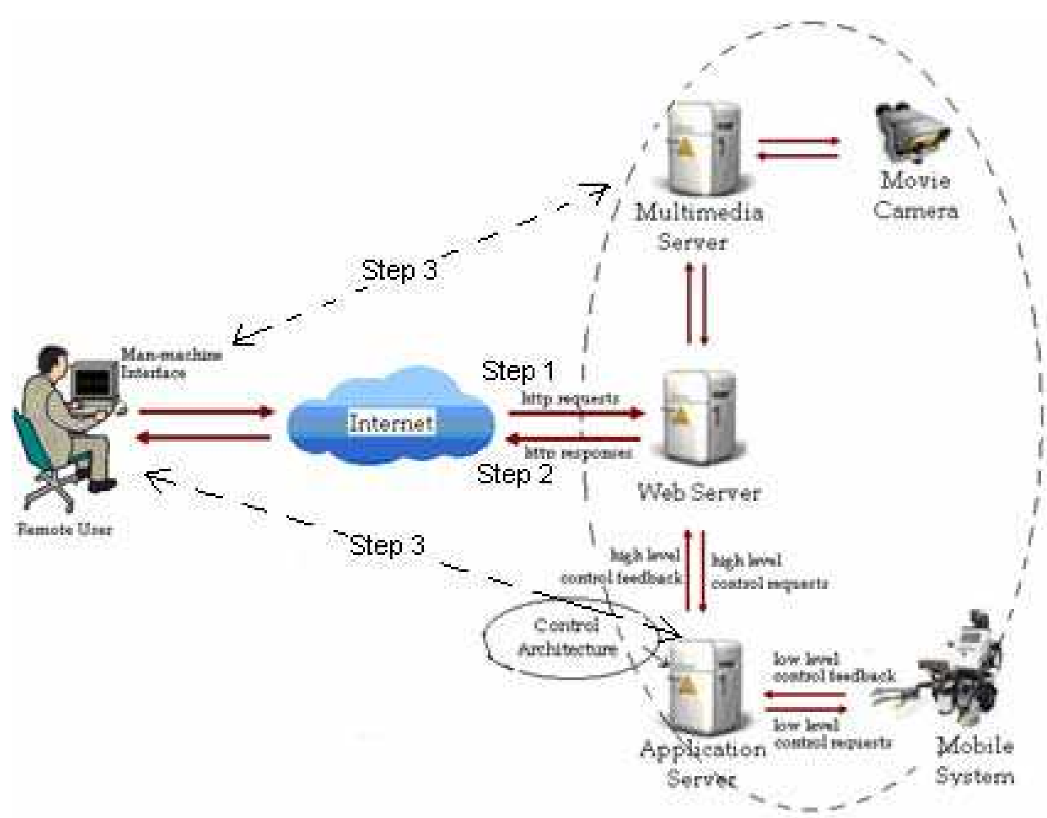


Fig. 7. Proposed System Architecture

From the study of the experiences made on Internet, a common frame (see Figure 7) can be described about the operational aspects of a remote control application (Sayouti & al., 2010).

The remote user, through his Internet navigator, addresses a request to a Web server (step 1) and downloads an application on his work station such as for example an applet Java (step 2). A connection is then established towards the server in charge of the management of the robot to control (step 3). The user is then able to take the remote control of it. In parallel to step 3, other connections are also established towards multi-media servers broadcasting signals (video, sound) of the system to be controlled.

5. Brief overviews of control architecture

One of the first authors who expressed the need for a control architecture was R.A. Brooks (Brooks, 1986). In 1986, he presented an architecture for autonomous robots called "subsumption architecture". It was made up of various levels which fulfil separately precise function, processing data from sensors in order to control the actuators with a notion of priority. It is a reactive architecture in the sense that there is a direct link between the sensors and the actuators (see Figure 8). This architecture has the advantage to be simple and thus easy to implement, nevertheless, the priorities given between the different actions to perform are fixed in time and do not allow an important flexibility.

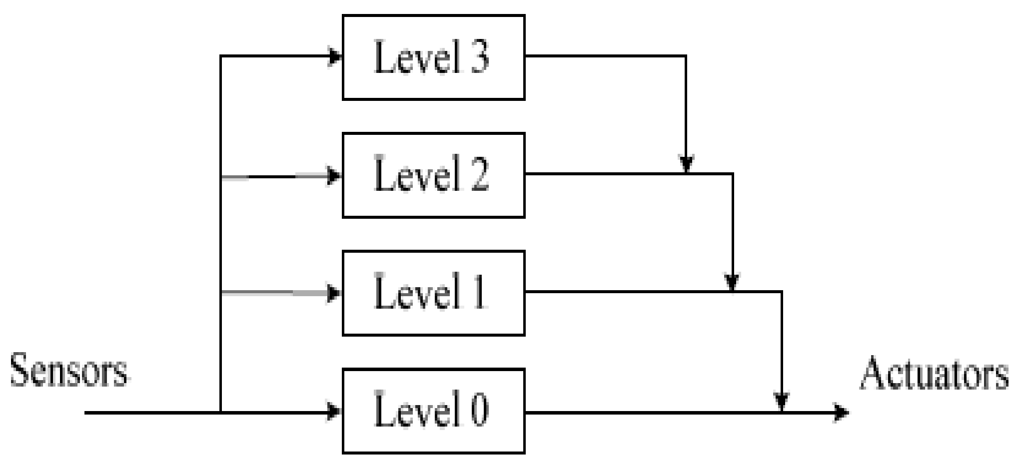


Fig. 8. Subsumption Architecture

Then other various architectures were developed based on different approaches, generally conditioned by the specific robot application that the architecture had to control.

The architecture 4-D/RCS developed by the Army Research Laboratory (Albus, 2002) has the main characteristic to be made up of multiple calculative nodes called RCS (Real time Control System). Each node contains four elements, performing the four following functionalities: Sensory Processing, World Modeling, Behavior Generation and Value Judgment. Some nodes contribute to the perception, others contribute to the planning and control. These nodes are structured in levels, in which one can find the influence of the reactive behaviors in the lower levels and of the deliberative behaviors in the higher levels.

The Jet Propulsion Laboratory developed in collaboration with NASA its own control architecture called CLARAty (Volpe, 2000). Its principal characteristic is to free itself from the traditional diagram on three levels (Functional, Executive, Path-Planner) and to develop a solution with only two levels which represent the functional level and the decisional level. A specific axis integrates the concept of granularity of the architecture for compensating the difficulties of understanding due to the reduction of the number of levels (see Figure 9). One of the interests of this representation is to work at the decisional level only on one model emanating from the functional level. The decomposition in objects of this functional level is described by UML formalism (Unified Modeling Language) that allows an easier realization of the decisional level.

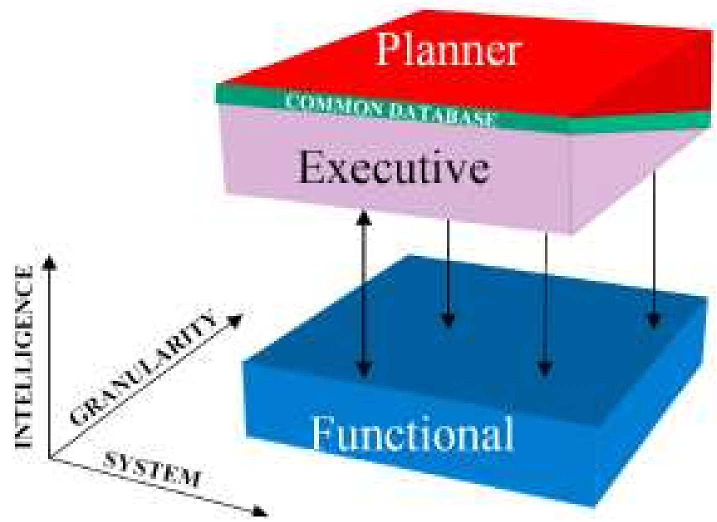


Fig. 9. Two level architecture

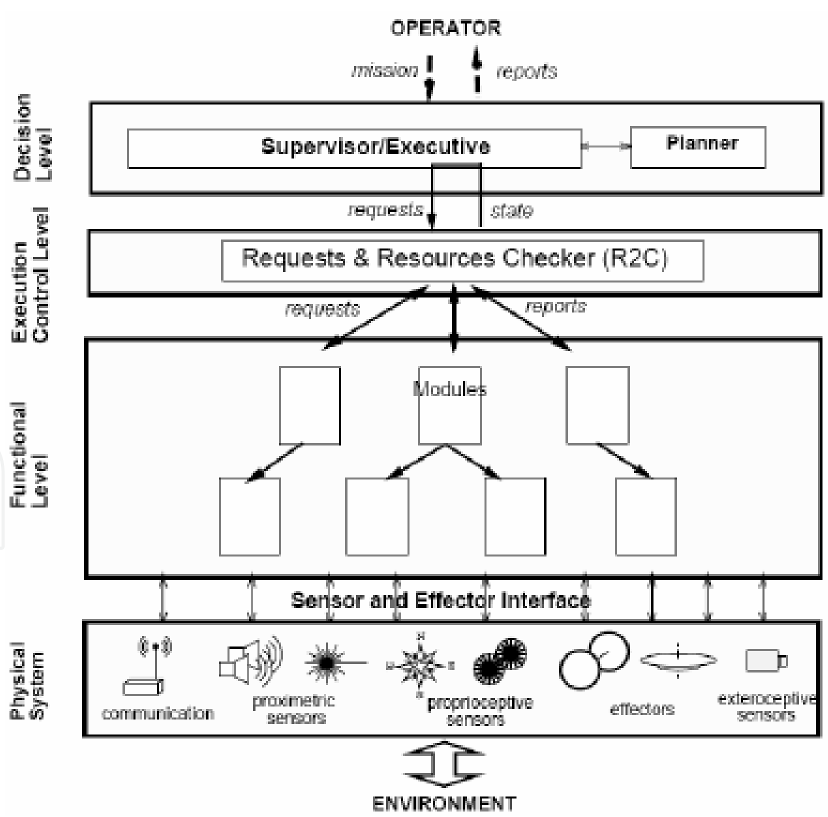


Fig. 10. LAAS Architecture

The LAAS architecture (Laas Architecture for Autonomous System) (Alami, 1998) is made up of three levels: decisional, executive and functional (see Figure 10). Its goal is to homogenize the whole mobile robotics developments and to be able to re-use already designed modules. All the modules of the functional level are encapsulated in a module automatically generated by GenoM. These have to interact directly with the actuators and other modules of the functional level. The higher level is a controller of execution (Request & Ressources Checker). Its main function is to manage the various requests emitted by the functional level or the decisional level. The operator acts only at the decisional level by emitting missions which depend on the information incoming from the lower levels. This architecture has an important modularity even if the final behavior is related to the programming of the controller of execution.

A. Dalgalarrondo (Dalgalarrondo, 2001) from the DGA/CTA proposed another architecture. It presents a hybrid control architecture including four modules: perception, action, attention manager and behavior selector (see Figure 11). It is based on sensor based behaviors chosen by a behavior selector. The "perception" module carries out models using processing which are activated or inhibited by the "attention manager" module. The "action" module consists of a set of behaviors controlling the robot effectors. A loop is carried out with the information collected by the perception part. This is particularly necessary for low level actions. The "attention manager" module is the organizer of the control architecture: it checks the validity of the models, the occurrence of new facts in the environment, the various processing in progress and finally the use of the processing resources. The "behaviour selector" module must choose the robot behavior according to all information available and necessary to this choice: the fixed goal, the action in progress, representations available as well as the temporal validity of information.

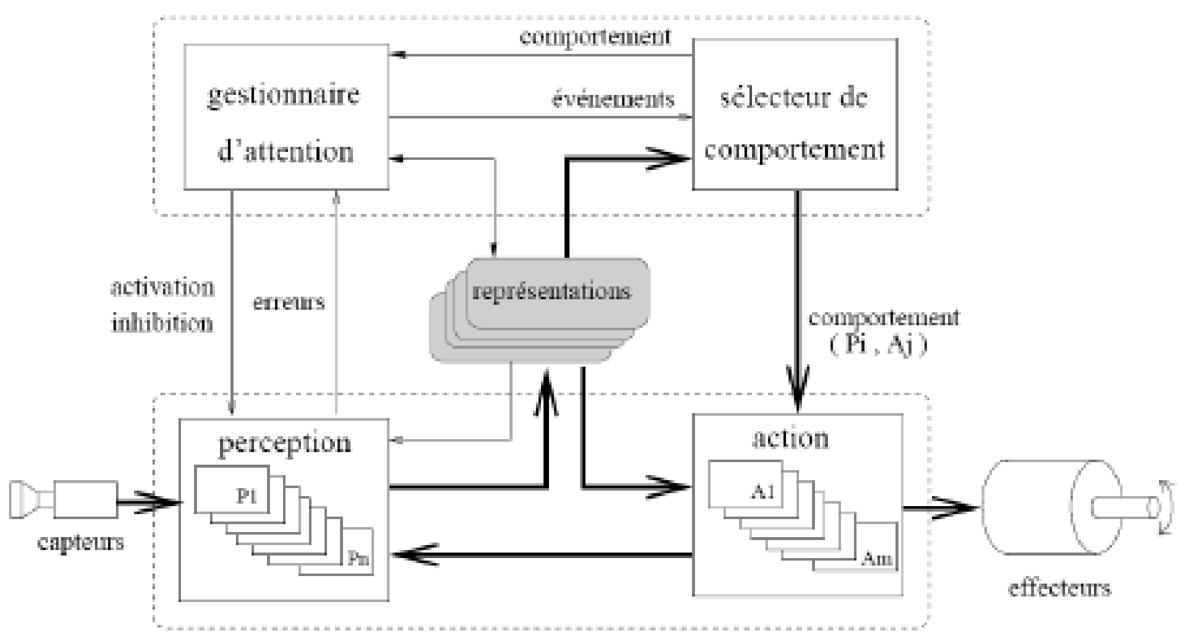


Fig. 11. DGA Architecture

The DAMN architecture (Distributed Architecture for Mobile Navigation) results from work undertaken at the Carnegie Mellon University (see Figure 12). Its development was a response to navigation problems.

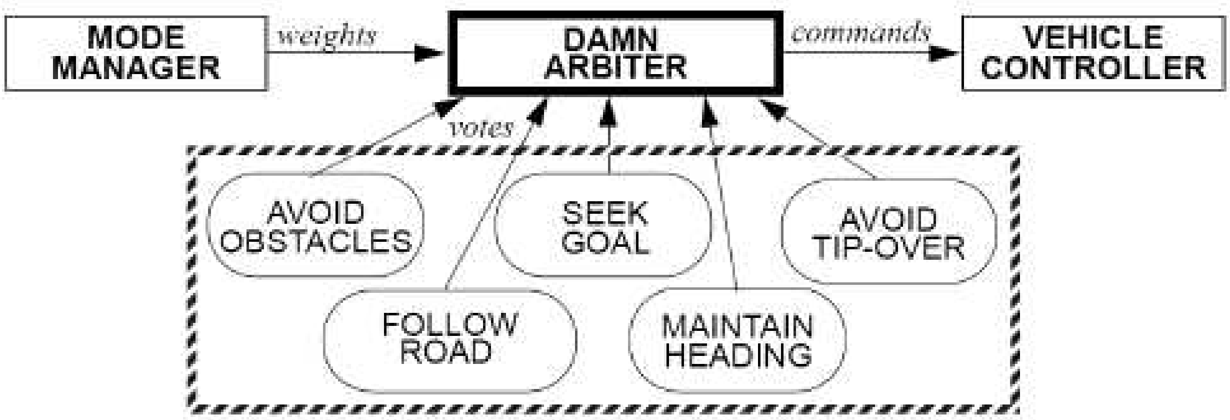


Fig. 12. DAMN Architecture

The principle of DAMN architecture is as follows: multiple modules share simultaneously the robot control by sending votes which are combined according to a system of weight attribution. Then, the architecture makes a choice of controls to send the robot, by a fusion of the possible solutions (Rosenblatt, 1995).

6. Proposed control architecture

Our architecture is based on the same architectures principles that have been presented in the previous section. It relies on the concept of levels initially developed by R. Brooks and which appear in architectures proposed by AuRA or LAAS. We propose a hybrid control architecture which combines aspects of classic control and behavior-based control. Our architecture called EAAS for EAS Architecture for Autonomous system (Sayouti & al., 2008b), including a deliberative part (Actions Selection Agent) and a reactive part. It is made up of two parts, each using distinct method to solve problems (see Figure 13). The deliberative part which uses methods of artificial intelligence contains a path planner, a navigator and a pilot. The reactive part is based on direct link between the sensors (Perception Agent) and the effectors (Action Agent).

Fundamental capacities of our architecture encompass autonomy, intelligence, modularity, encapsulation, scability and parallel execution. To fulfil these requirements, we decided to use a multi agents formalism that fits naturally our needs.

The Multi-Agent System paradigm is one of the most promising approaches to create autonomous, open and dynamic systems, where heterogeneous entities are naturally represented as interacting autonomous agents, which can enter or leave the system at will. Interaction among autonomous agents is fundamental to the dynamic of multi-agent systems. Agents need to interact and coordinate their activity to carry out their common global goal. The development of Multi agents systems is mainly due to its interactions with different scientific domains, in particular with biology. Biology, and especially ethology, inspired the first architecture and several distributed algorithms (Drogoul & Ferber, 1992)(Ferber, 1999).

The communication between agents in our architecture is realized by messages. Object oriented language is therefore absolutely suited for programming agents (we chose java). We use threads to obtain parallelism (each agent is represented by a thread in the overall process).

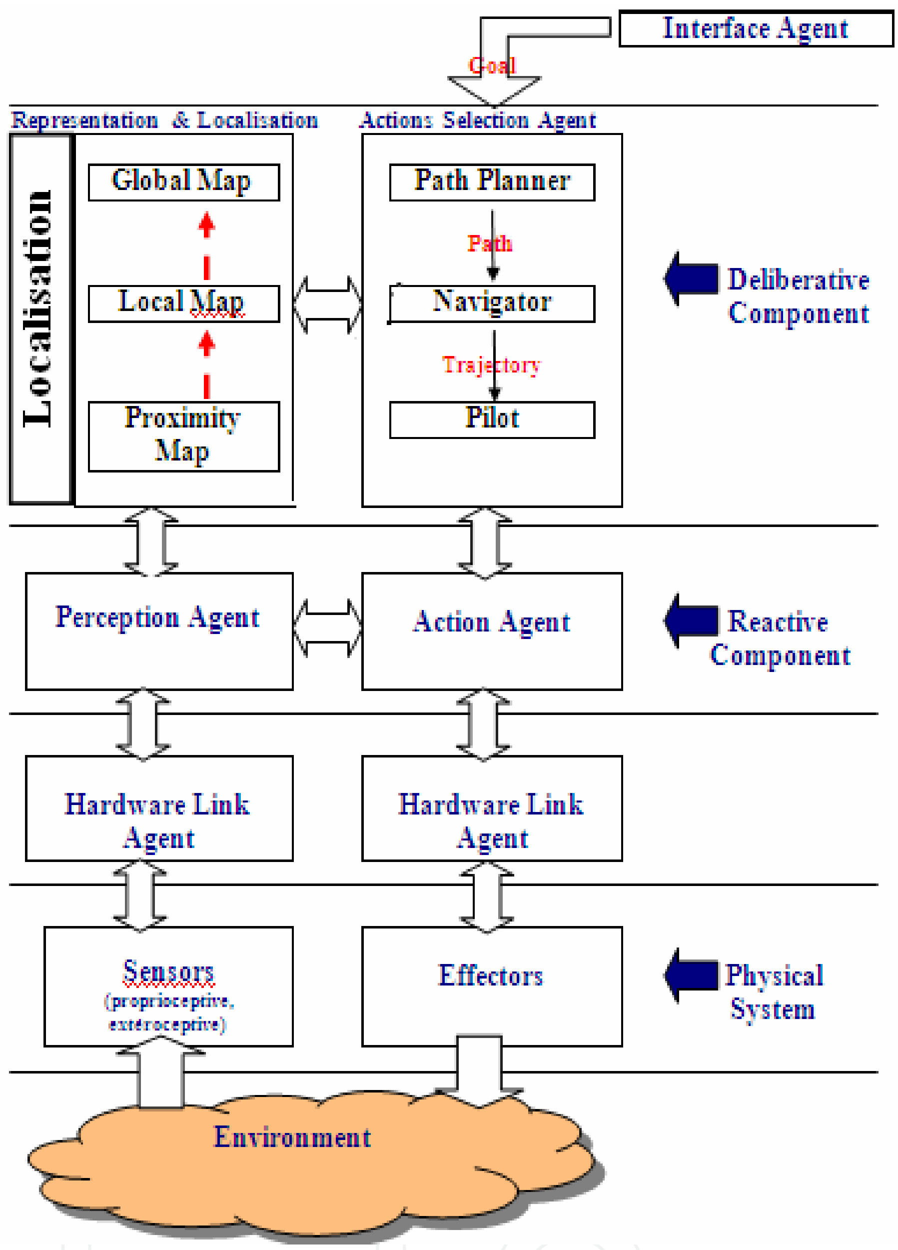


Fig. 13. EAAS Architecture

EAAS architecture consists in five agents: interface agent, actions selection agent, perception agent, action agent and hardware link agent. The interface agent is the high level of our control architecture. It must generate a succession of goal, or missions for the actions selection agent, according to the general mission of the robot. It is the "ultimate" robot autonomy concept: the robot generates itself its own attitudes and its own actions by using its own decisions. The perception agent manages the processing of incoming data (the sensor measurements) and creates representations of the environment. The actions selection agent must choose the robot behavior according to all information available and necessary to this choice: the fixed goal, representations and the robot localization. The actions selection agent contains a path planner, a navigator and a pilot. The path planner may take a goal as input and give a path for achieving the goal as output. The Navigator must translate a path into a trajectory for the pilot. The path does not take into account physical constraints of the robot, but the trajectory that it delivers must integrate them. The function of the pilot is to convert this trajectory into orders to be performed by the action agent. The action agent consists of a set of behaviors controlling the robot effectors. The hardware link agent is an interface between the software architecture and real robot. Changing the real robot require the use of a specific agent but no change in the overall architecture.

7. Remote control software architecture

A Software architecture has been defined to make remote control of mobile systems possible (Sayouti & al. 2008a) (Sayouti, 2009). Our software architecture is based on a set of independent agents running in parallel.
On the left side of figure 14, the server side is represented. It is basically composed of three main agents: "Connection Manager" which manages the different connected clients according to a Control Algorithm. This one is chosen by the designer of the system depending on the application: master/slave, priority, timeout... The "Media" agent communicates with the camera in order to broadcast signals (video, images) of the mobile system in its environment. The "SMA EAAS" (EAS Architecture for Autonomous Systems) which represents our control architecture. EAAS architecture is a hybrid control architecture including a deliberative part (Actions Selection Agent) and a reactive part. The reactive part is based on direct link between the sensors (Perception Agent) and the effectors (Action Agent).

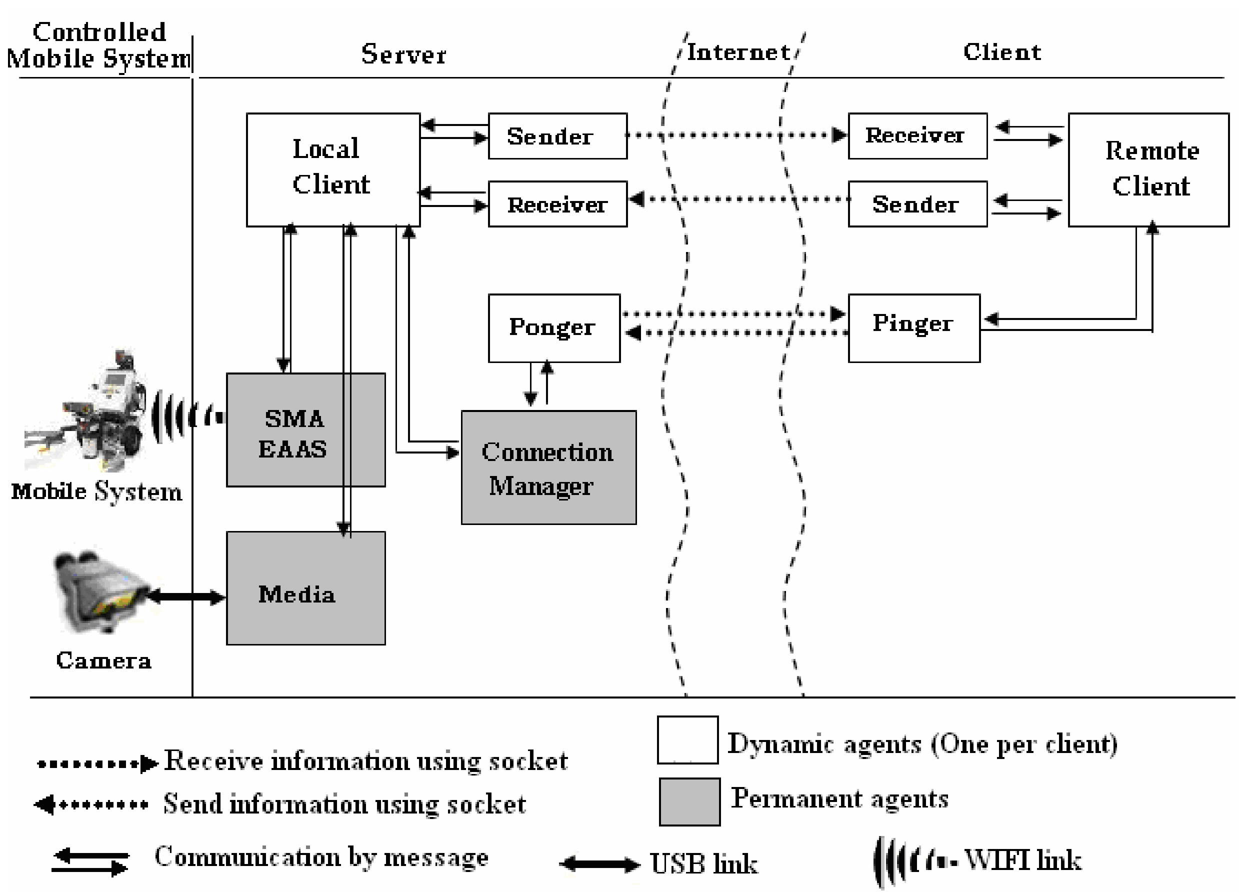


Fig. 14. Remote control software architecture proposed

The right side of the figure represents the client side. Agents are loaded in a web navigator. The "Remote Client" corresponds to a graphical user interface which allows the user to send orders to the mobile system and receive information about the environment. "Sender" and "Receiver" agents are used to allow the communication between the client and the server. "Pinger" and "Ponger" agents are used to observe dynamically the network. If the connexion is accepted, the "Connection Manager" will inform the "Local Client" agent which achieves the interface with the "SMA EAAS" to transmit orders transmission to the mobile system.

8. Applications

Different applications have been developed, in our laboratory, using concepts explained in the previous sections (Sayouti & al, 2009) .

The mobile system used in our realizations is a Lego robot. Lego Mindstorms (LEGO Company, 1997) is a development kit for manufacturing a robot using Lego blocks, and is gaining widespread acceptance in the field of technical education. By using a Mindstorm, a robot can be manufactured for various purposes and functions. It is beginning to be considered as a component of experimental equipment in robotics research.

The Lego mobile robot (see Figure 15) is powered by three reversible motors coupled to wheels and equipped with four sensors: sonar sensor, sound sensor, light sensor and touch sensor. The data produced by these sensors are used by perception agent to build a global map of the Lego robot environment's. This global map, the goal and the Lego robot localisation are used by the actions selection agent to define a plan of actions to achieve its mission. The Lego robot is equipped with Bluetooth connection that permits the communication with the application server and facilitates its displacement in the environment in order to reach its objective.

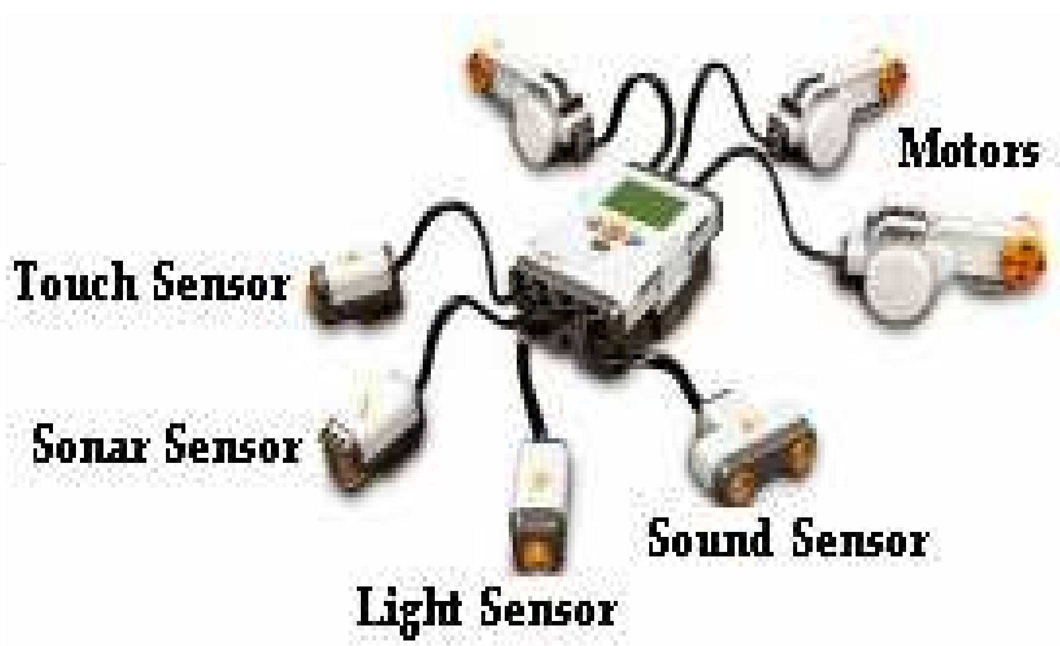


Fig. 15. General Architecture

During the project development, different configurations were tested in different environments. The aim is to develop a more reliable system architecture that can be used in the real world.

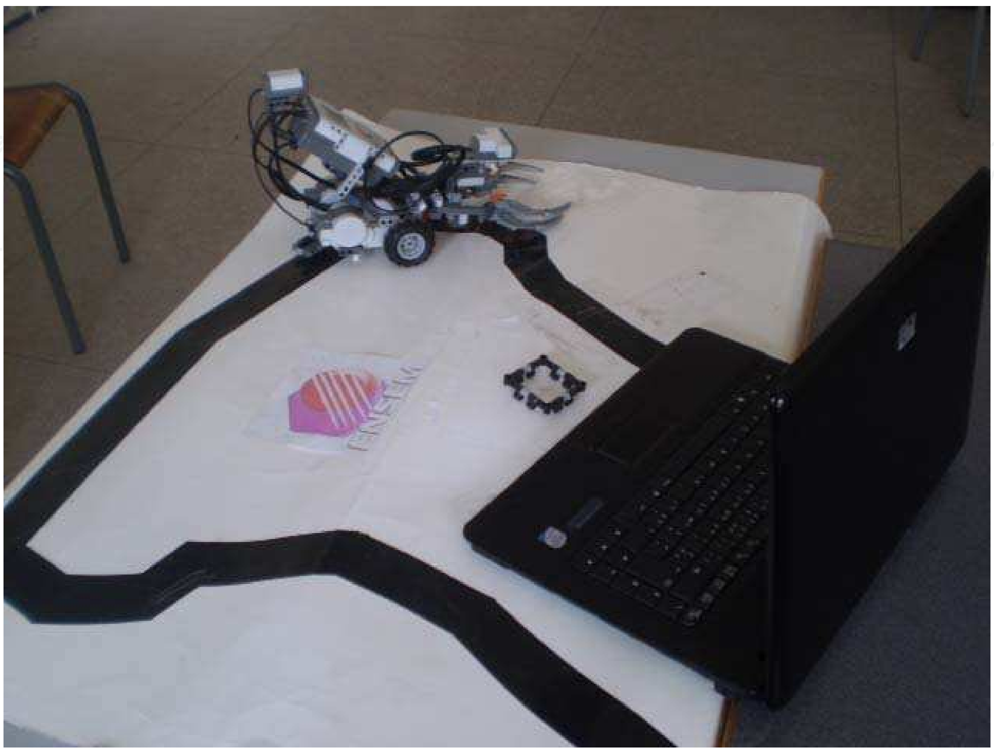


Fig. 16. Lego robot in its environment

Our Lego mobile robot was controlled to explore the laboratory while avoiding several static obstacles. In another test, the Lego mobile robot was controlled to push a ball to the goal.
The Web interface of our application (see Figure 17) is designed with the intention of making the remote control easy for researchers and students in order to interact with the Lego mobile robot. A simple interface is designed to provide as much information as possible for remote control. This user interface consists of several Java Applets as shown in Figure 8. It can work on any web browser.
The link between client(s) and server is based on two standard Internet protocols. First, the communication is based on the HTTP protocol, through a home page linking the applets used to move the mobile system. These applets are interpreted by the JVM of the browser. Then, the applets downloaded on the client communicate with the application server through the TCP/IP protocol.
The start button permits to start our application. The remote operator is invited to test the connection using the statistical or the dynamical way, before or during taking the control by clicking on the buttons labelled statistic or dynamic test respectively.
The remote users can pass online tests of knowledge in order to follow a formation answering to their needs. Different modules of formation (Cursus link) are available on our web site, to know: remote control, multi-agents systems, systems architecture, control architecture and autonomous mobile system. We have also set to the remote users a discussion forum within our application to interchange their ideas over the remote control subject.

Fig. 17. Web Interfaces

With this simple web interface, one user can control the mobile robot from the web browser with the visual feed back. The other users only have the visual feedback at the same time, and have to wait in queue until the first user logout at this stage.

9. Conclusion

In this chapter, we have presented a Web-based remote control application so that Internet users, especially researchers and students, can control the mobile robot to explore a dynamic environment remotely from their home and share this unique robotic system with us. In the first part, an analysis of the existing control architectures and the approaches for their development has guided us to design a hybrid control architecture. It is called EAAS for EAS Architecture for Autonomous System. The proposed generic architecture consists in associating a deliberative approach for the high level and a reactive approach for the low level. The deliberative level allows decision-making and actions planning thanks to the use of the agent selection of actions. The reactive level, based on couples of agents perception / action, allows the mobile system to react facing the unforeseen events. Then, the software implementation of our architecture was presented. It is achieved under the shape of a multi-agents system by reason of its autonomy, intelligence, flexibility and the various possibilities of evolution. In the second part, in order to validate the choice of our architecture, we presented two applications achieved by the system architecture team of the ENSEM.

The long-term goal of our research is towards real-world applications such as tele-teaching, tele-maintenance, tele-expertise and tele- production.

In future work we hope to increase the intelligence of the agents in our control architecture to provide a telerobot with a high degree of local intelligence to handle restricted bandwidth and transmission delay of the network and to integrate multiple mobile robots into a telerobotics system to achieve redundancy and robustness. This will pave the way for the remote exploration of an unknown and complex environment through the Internet.

10. References

Alami, R., Chatila, R., Fleury, S. Ghallab, M. & Ingrand, F. (1998). An architecture for autonomy. *The International Journal of Robotics Research, Special Issue on Integrated Architectures for Robot Control and Programming*, vol. 17, No 4, pp. 315-337

Albus, J. S. (2002). 4D/RCS A Reference Model Architecture for Intelligent Unmanned Ground Vehicules, *Proceedings of the SPIE 16th Annual International Symposium on Aerospace/Defense Sensing, Simulation and Controls*, 2002, Orlando

Brooks, R.A. (1986). A robust layered control system for a mobile robot, *IEEE Journal of Robotics and Automation*, vol. 2, n. 1, pp. 14–23

Dalgalarrondo, A. (2001). Intégration de la fonction perception dans une architecture de contrôle de robot mobile autonome, *Ph.D. Thesis, Université Paris-Sud*, 2001, Orsay, France

Drogoul, A., Ferber, J. (1992). Multi-agent simulation as tool for modelling societies: Application to social differentiation in ant colonies, *Proceedings of the MAAMAW'92*, 1992, Viterbo

Ferber, J. (1999). Multi-agents systems, an introduction to distributed artificial intelligence *Addison-Wesley*

Goldberg, K., Siegwart, R. (2001) Beyond Webcams: an introduction to online robots, *The MIT Press*

Goldberg, K., Gentner, S., Sutter, C., Wiegley, J. (2000). The mercury project: a feasibility study for internet robotics, *IEEE Robotics & Automantion Magazine*, pp. 35-40

Hu, H. YU, L. Tsui, P. Zhou, Internet-based Robotic Systems for Teleoperation, *International Journal of Assembly Automation*, Vol. 21, No. 2.

Kosuge, K., Kikuchi, J., Takeo, K. (1998). A Teleoperation system via computer Network, *Proceedings of the IROS'98: Workshop on Web Robots*, 1998, pp. 61-66, Victoria, Canada

Le Parc, P., Vareille, J. & Marce, L. (2004). E-productique ou contrôle et supervision distante de systèmes mécaniques sur l'internet, *Journal européen des systèmes automatisés (JESA)*, Vol.38, No. 5, p. 525-558.

Le Parc, P., Vareille, J., Marcé, L. (2005). Long distance remote control over Internet: a reliability challenge, *Proceedings of the XVI Workshop on Supervising and Diagnostic of Machining Systems*., 2005, Karpacs, Poland

LEGO Company, *Lego mindstorms official page*, http://mindstorms.lego.com, 1997.

Novales, C., Mourioux, G. & Poisson, G. (2006). A multi-level architecture controlling robots from autonomy to teleoperation. *First National Workshop on Control Architectures of Robots*, April 6, 7, 2006, Montpellier

Rosenblatt, J. (1995) DAMN: A Distributed Architecture for Mobile Navigation, *Proceedings of the 1995 AAAI Spring Symposium on Lessons Learned from Implemented Software Architectures for Physical Agents,* AAAI Press, Menlo Park, CA

Saucy, P., & F. Mondada (2000). Khepontheweb: Open access to a mobile robot on the internet, *IEEE robotics and automation magazine*, pp. 41-47, March 2000.

Sayouti, A. (2009). Conception et Réalisation d'une Architecture de Contrôle à Distance Via Internet à Base des Systèmes Multi-Agents. Phd. Thesis, ENSEM, Hassan II University, 2009.

Sayouti, A., Lebbat, A., Medromi, H. & Qrichi Aniba, F. (2010). Remote Laboratory for Teaching Mobile Systems". *International Journal of Computer and Network Security (IJCNS)*, Vol. 2, No. 3, pp. 28-32

Sayouti, A., Qrichi Aniba, F. & Medrom, H. (2008a). Remote Control Architecture over Internet Based on Multi agents Systems. *International Review on Computers and Software (I.RE.CO.S)*, Vol 3, No. 6, pp. 666 – 671

Sayouti, A., Qrichi Aniba, F., Medrom, H. & Radoui, M. (2008b). Tele-robotic over Internet Based on Multi-agents System, *Proceedings of the Seventh International Conference on Distributed Computing and Applications for Business, Engineering and Sciences (DCABES)*, 2008, Dalian, China

Sayouti, A., Qrichi Aniba, F., Medrom, H. (2009). Modeling Autonomous Mobile System with an Agent Oriented Approach". *International Journal of Computer Science and Network Security (IJCSNS)*, Vol. 9, No. 9, pp. 316 – 321

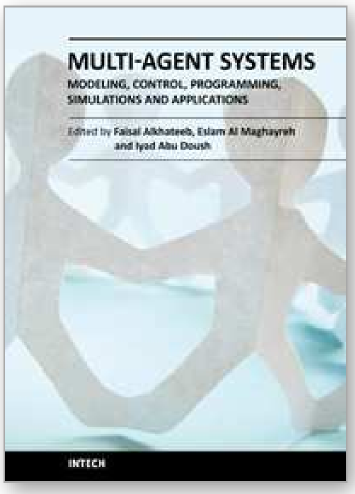
Schulz, D., Burgard, W., Cremers, A. B. (1998). Predictive simulation of autonomous robots for tele-operation systems using the world wide web, *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and System*, 1998, Victoria, B.C.

Simmons, R., Xavier. (1998). An autonomous mobile robot on the web, *Proceedings of the International Workshop On Intelligent Robots and Systems (IROS)*, 1998, Victoria, Canada

Stein, M.R., Painting on the world wide web: the pumapaint project, Proceedings of the IEEE IROS'98 Workshop on Robots on the Web, 1998, Victoria, Canada

Taylor, K., Dalton, B. (1997). Issues in internet telerobotics, *Proceedings of the FSR'97 International Conference on Field and Service Robots*.

Volpe, R. (2000). CLARAty: Coupled Layer Architecture for Robotic Autonomy (*JPL Technical Report D-19975*



Multi-Agent Systems - Modeling, Control, Programming, Simulations and Applications
Edited by Dr. Faisal Alkhateeb

ISBN 978-953-307-174-9
Hard cover, 522 pages
Publisher InTech
Published online 02, February, 2011
Published in print edition February, 2011

A multi-agent system (MAS) 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. Agent systems are open and extensible systems that allow for the deployment of autonomous and proactive software components. Multi-agent systems have been brought up and used in several application domains.