Advanced Robotics, Vol. 14, No. 1, pp. 75–86 (2000)

Short paper

A programming environment for real-time control of distributed multiple robotic systems

MAURIZIO PIAGGIO, ANTONIO SGORBISSA and RENATO ZACCARIA

DIST Laboratorium, University of Genoa, Via Opera Pia 13, 16145 Genoa, Italy

Received 2 February 1999; revised 1 June 1999; accepted 11 December 1999

Abstract—In recent years there has been great interest in robot software control architectures. However, although many interesting solutions have been presented, most of the research problems tackled related to a single robot perception, navigation and action in everyday environments. Instead, most of the practical applications of mobile robotics for service tasks in civilian environments consist of systems composed of multiple robots communicating with each other, with external sensing and actuating devices, and with external supervising workstations. RoboCup offers a great opportunity to deal with this problem. In fact the software architecture of a robot soccer player must allow successful intra-robot integration of the different activities (visual perception, path planning, strategy planning, motion control, etc.) spanning many different types of representation (raw sensor data, images, symbolic plans, etc.) and it must also guarantee successful inter-robot integration by supporting communication and cooperation. This paper focuses on this problem, presenting ETHNOS-IV — a programming environment for the design of a real-time control system composed of different robots, devices and external supervising or control stations — which has been successfully used within the Italian ART robot team in the RoboCup-99 competition. ETHNOS provides support from three main point of views which will be addressed in detail: inter-robot and intra-robot communication, real-time task scheduling, and software engineering and code reuse. Experimental results illustrating the advantages of this approach will also be presented.

Keywords: Heterogenous robotics; control architectures; distributed robotics; real-time scheduling; software engineering; RoboCup.

1. INTRODUCTION

In recent years there has been a great interest in robot software control architectures, witnessed by the many conferences [1], workshops [2] and journal special issues [3] dedicated to this problem. However, although many interesting solutions have been presented, most of the research tackled problems related to a single robot perceiving, navigating and acting in everyday environments. Instead, most of the practical applications of mobile robotics for service tasks in civilian environments consist of systems composed of multiple robots communicating with each other, with external sensing and actuating devices, and with external supervising workstations.

Only very recently, thanks to the progress in communication technology which has widened the horizon of distributed robotics [4], have research issues in autonomous robot architectures be addressed taking into account this multi-agent notion of robot autonomy. In this sense, the RoboCup [5] competition offers a great opportunity to deal with this problem. In fact, the software architecture of a robot soccer player must allow successful intra-robot integration of the different activities (visual perception, path planning, strategy planning, motion control, etc.) spanning many different types of representation (raw sensor data, images, symbolic plans, etc.), and must also guarantee successful inter-robot integration by supporting communication and cooperation.

These problems are particularly significant in ART — the Italian robot soccer team. ART is a national team made up of contributions from six research groups from different universities. Each of the research groups involved in the team is responsible for the development of its own robot which differs in terms of mechanics (robot base and kickers), hardware (processor type and speed) and control software (different methodologies for perception, reasoning and motion control).

For this reason the software architecture and development environment in ART assumes a fundamental role in the team integration, ranging from communication and coordination of heterogeneous robots to software exchange and re-use across different robot platforms.

This paper presents ETHNOS-IV — a programming environment for the design of a real-time control system composed of different robots, devices and external supervising or control stations — which focuses on these issues and which has been successfully used in the ART team in RoboCup-99 competition.

2. ETHNOS IV

ETHNOS IV (Expert Tribe in a Hybrid Network Operating System) is a programming environment for multi autonomous robot systems. It is composed of:

(1) A dedicated distributed real-time operating system supporting different representation, communication and execution requirements. These functionalities are built on top of a Posix compliant Linux RT kernel.

(2) A dedicated network communication protocol designed for both the single robot and the multi-robot environment, specifically taking noisy wireless communication into account.

(3) An object-oriented Application Programming Interface (API) based on the C++ language (and a subset based on Java).

(4) A set of additional development tools (a robot simulator, a Java applet template, etc.)

The reference architecture of a single robot and, consequently, of the ETHNOS operating system is entirely based on the concept of *expert*, a concurrent agent responsible for a specific deliberative or reactive behavior. The experts in ETHNOS can be organized in groups, possibly distributed in a computer network, depending on their computational characteristics: the type of cognitive activity carried out, timing constraints, type of data managed, duration, etc. However, these groups do not have a hierarchical organization but, on the contrary, are executed in a flat model in such a way that the real-time constraints are not violated.

This generality in the expert specification allows the use of ETHNOS with different architectural organizations without loosing real-time analysis, execution and inter-robot communication support. This aspect has been very important in ART in which:

- Homer and Bart robots (Padova research group) were developed using a subsumption-based model.
- TinoZoff goalkeeper (Parma research group) was developed using a central controller.
- Rullit robot (Milan research group) was based on fuzzy logic control.
- Relè (Genova research group) was based on a hybrid diagrammatic/reactive model.

Figure 1 illustrates this concept depicting the use of ETHNOS in the implementation of three different architectures [6–8]. The same characteristics apply to multi-robot

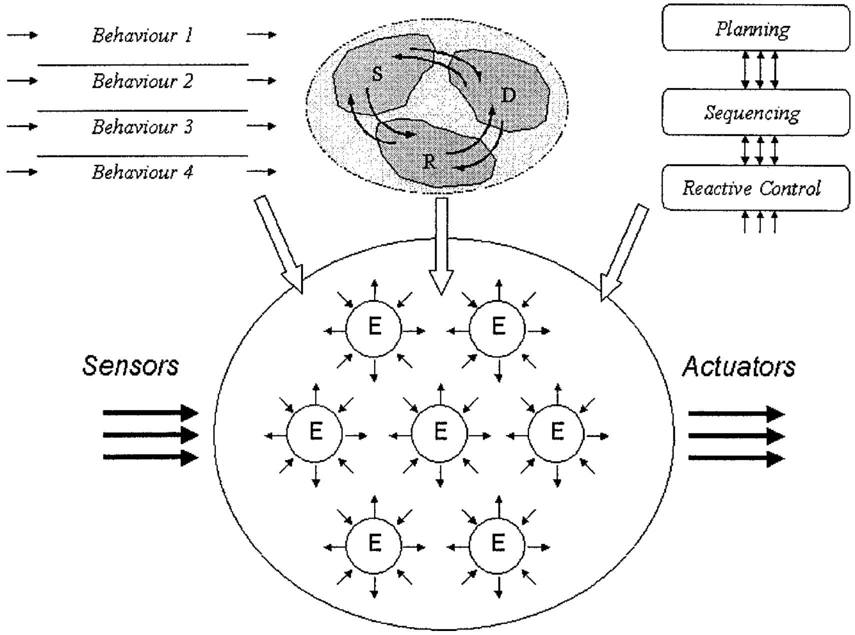

Figure 1. ETHNOS implementing HEIR (in the middle), a behavior-based subsumption-like model (on the left) and a three-layered model (on the right).

control in which ETHNOS could implement different coordination strategies as in [9, 10].

2.1. Inter-expert and inter-robot communication

One of goals of RoboCup is to encourage the comparison and exchange of methodologies, techniques and algorithms within the robotics and artificial intelligence community. A common programming environment which, without imposing significant constraints on the single components, allows us to easily put together the result of different researchers within the same group (or possibly also across different groups) is certainly a contribution in this direction. In harmony with this goal, the ETHNOS programming environment allows the robots to be programmed in such a way that the different experts can be integrated, during development but also at run-time, with little or no intervention in the software of the expert itself, thus facilitating both rapid prototyping and dynamic architectural reconfiguration. The first property facilitates the development of a robot application (from a set of components or behaviors) even by non-highly specialized programmers (for industrial purposes but also for didactic activity in student projects, particularly relevant in RoboCup); the second property allows the system to easily scale-up to a robot capable of different complex activities and thus to be able to switch at run-time from a configuration in which it is performing a certain task (e.g. in which the robot is attacking and thus the active behaviors are responsible for ball pushing, path planning and obstacle avoidance) to a different configuration (e.g. in which the robot is defending and the active behaviors are responsible for opponent tracking, ball interception, etc.).

These properties are achieved by exploiting a suitable inter-expert publish/subscribe-based message communication protocol (the EIEP — Expert Information Exchange Protocol [11] fully integrated with the expert scheduler. The EIEP encourages the system developer to de-couple the different experts in execution, to reach, as close as possible, the limit situation in which the single expert is not aware of what and how many other experts are running but only of the data being exchanged. In this way an expert can be added, removed or modified at run-time without altering the other components of the system.

Moreover, in ETHNOS the different experts are allowed to subscribe to *communication clubs*. For example, we may envisage a single club to which the different players belong or even different clubs, to which only players which are performing a cooperative activity belong. Message subscription and publication can thus be distinguished in internal, external in a specific club or external in all clubs. Again, it is the responsibility of the system to dynamically transparently distribute the messages to the appropriate receivers. Figure 2 shows an example configuration that we have tested. In particular, we allow the robots to communicate in a single club, i.e. the team club (to which all of them have subscribed), and with an external supervisor (the *coach*) which monitors the activity of all the players for displaying and debugging the team activity during a match.

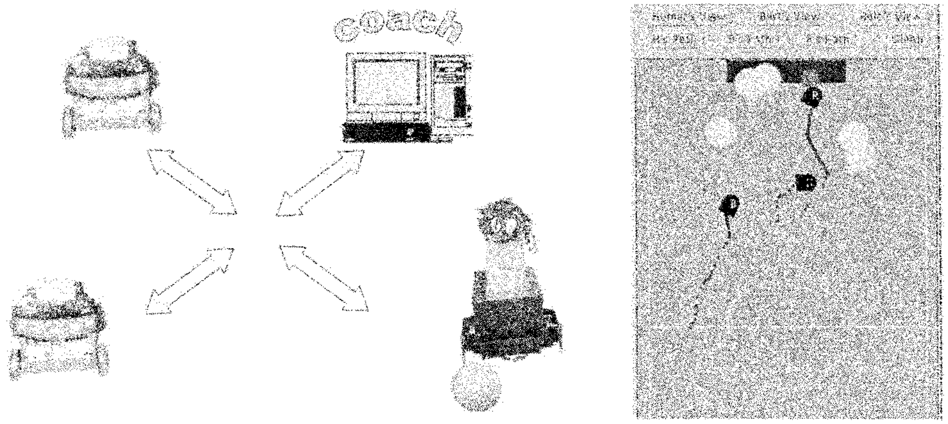

Figure 2. Left: example of multi-robot ETHNOS configuration. Right: coach monitoring the team activity.

It is worth noticing that, because of EIEP protocol, whenever we want to add (remove) a player to (from) the team, it is not necessary to explicitly inform each player about the modifications in the team's composition. In fact, the players just have to agree about the type of information they are ready to send and receive: it is ETHNOS which automatically updates the database of each club's members and consequently dynamically dispatches messages from senders to the appropriate receivers. This has been very important in ART in which there were more than four types of robots available, with different characteristics of play, and thus different team compositions were selected for the single matches and also modified during the matches to better contrast the opponent.

Moreover, since in RoboCup (and in general in mobile robotics) network communication is often wireless (i.e. radio link, Wavelan©, etc.), transmission packets are sometimes lost due to noise or interference. In this context, both TCP-IP- and UDP-IP- based communication cannot be used: the former because it is intrinsically not efficient in a noisy environment; the latter because it does not guarantee the arrival of a message nor any information on whether it has arrived or not. For this reason we have also designed a protocol for this type of application, called EEUDP (Ethnos Extended UDP) because, based on the UDP, it extends it with the necessary properties.

The EEUDP allows the transmission of messages with different priorities. The minimum priority corresponds to the basic UDP (there is no guarantee on the message arrival) and should be used for data of little importance or data that is frequently updated (e.g. the robot position in the environment that is periodically published). The maximum priority is similar to TCP because the message is sent until its reception is acknowledged. However, it differs because it does not guarantee that the order of arrival of the different messages is identical to the order in which they have been sent (irrelevant in ETHNOS applications because every message is independent of the others), which is the cause of the major TCP overhead. Different in-between priorities allow the re-transmission of a message until its reception is acknowledged for different time periods (i.e. 5, 10 ms, etc.).

2.2. Real-time expert scheduling

Another important aspect of the ETHNOS environment is the support in the real-time analysis and execution of the different experts. However, before dealing with these properties, let us first investigate the timing requirements of the different activities that are carried out by RoboCup players.

Even if it seems intuitive that a mainly reactive approach is best suited for tackling with the highly dynamic environment of a RoboCup match (there is probably not enough time for symbolic planning before acting if you want to reach the ball, avoid adversaries or score a goal in most situations), some kind of more deliberative activity could be carried on in the background to increase the performances of the single player and thus of the whole team, e.g. see game strategy and team mate coordination (currently mostly investigated in simulated and small size RoboCup [12]. Given such premises the concurrent tasks that the robot has to perform have different timing characteristics and requirements: while most of the tasks are periodic and sporadic ones with upper-bounded and very short computational time and have hard/soft real-time requirements (i.e. reading sensors, locally choosing the best trajectory to reach the ball, piloting actuators, etc.), the deliberative activities are carried on by tasks whose execution time probably cannot be upper-bounded or which is very long with respect to the reactive ones. An overall architecture for the development of RoboCup players should, in our view, take these aspects into account and thus permit the integration of reactive planning and control activities (whose execution is critical for the system and which therefore have real time requirements) with deliberative ones (whose execution is not critical for the good functioning of the system but which can increase performance in many situations).

Non-real-time tasks, since they are not critical to the behavior of the system, can be executed in a low-priority process. They will run in the background and communicate their results to reactive components whenever the information they produce can be used to guide the reactive behavior of the system. It should be outlined that, given the non-time-predictable nature of these activities, such communication happens very seldom with respect to the continuous exchange of information between the real-time tasks.

Taking these consideration into account ETHNOS provides different possible solutions to this scheduling problem ranging from two extremes in which:

(1) There are two processes: a higher priority process, which non-pre-emptively schedules the tasks with real-time responsiveness requirements (as stated in the non-pre-emptive EDF [13]), and a lower priority process, which executes in the background (when the higher level process is not running) the deliberative tasks required for coordination activities (whose execution time cannot be bound and whose requirements are not real-time).

(2) Real-time experts are mapped, one to one, into different Posix threads and scheduled as stated by the Rate Monotonic algorithm [14] with a Deferrable Periodic Server [15] where as background experts can be assigned a lower priority.

It is worth noticing that as well as providing transparent real-time execution (automatic computation of the required priorities) ETHNOS also includes a tool for analyzing the feasibility of the scheduling task. In fact, in a preliminary phase, in both the Rate Monotonic and the EDF scheduler cases, ETHNOS performs an approximate worst execution time analysis (because of the soft real-time requirements of such applications, the worst analysis computation does not have to be extremely precise).

2.3. The programming interface

ETHNOS-IV has been designed to facilitate the development of complex distributed real-time applications even by non-highly specialized users. This is particularly true in the RoboCup context, in which the exchange of research results between attending teams assumes a relevant interest.

For this reason we have developed a dedicated ETHNOS programming environment for the development of robotics applications, providing an object-oriented programming interface [Ethnos Programming Interface, (EPI)] for the software implementation of the different components of the architecture. The EPI consists of a library of abstract C++ classes related to all the elements of the architecture (experts, timers, messages, expert activation and termination requests, etc.), encapsulating their properties and their common behaviors.

Thus, in ETHNOS, object orientation in the expert definition, expert de-coupling based on the EIE protocol and flat real-time execution are jointly exploited to allow the user to rapidly put together different experts and build a prototype of a real-time application. Notice that this can be carried out automatically without having to explicitly modify the experts for any new configuration.

3. EXPERIMENTAL RESULTS

ETHNOS has been extensively tested and evaluated in RoboCup in the Italian middle size robot team — ART. In this section we will describe the advantages of its use in this domain from the different points of view that have been mentioned throughout the paper; in particular, from the communication perspective and from the software engineering perspective. Let us examine these aspects in detail.

3.1. Communication

From the communication perspective the support provided by ETHNOS is at two different levels: network communication and message distribution.

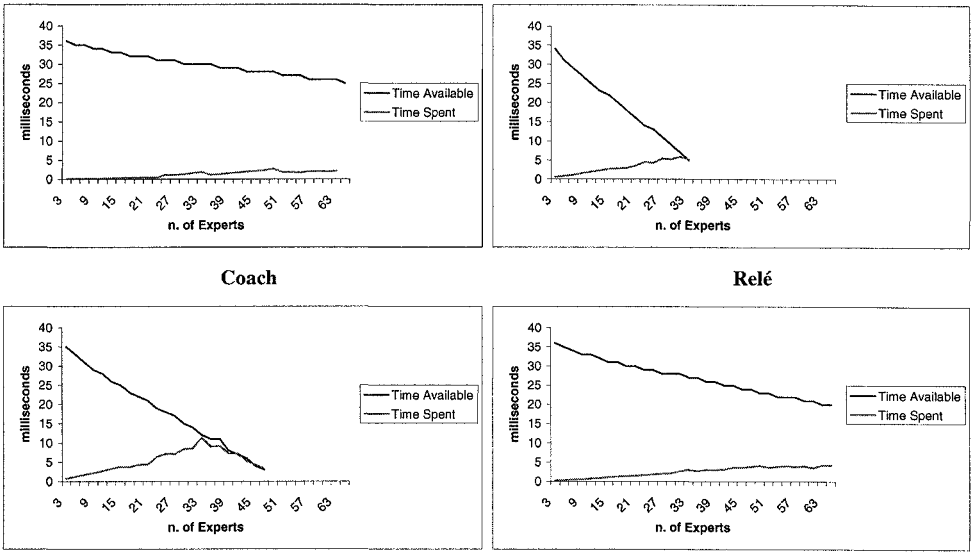

Figure 3. Network communication in ETHNOS.

For network communication the system allocates a maximum guaranteed and dedicated time. This value is computed automatically on the basis of the schedulability analysis so that the real-time execution of the user experts is also guaranteed. Thus clearly the value depends on the computational load of the experts in execution as well as the processor speed.

In Fig. 3, the four graphs represent different machines (with different processing power) corresponding to three robots (Relé, Homer and Bart) and a coach, connected using Wavelans®. The top line in the graph indicates the calculated time available each 100 ms for communication purposes. Clearly this value decreases as the number of experts in execution increase (and so the computational load). The bottom line indicates the time spent in communication which also increases with the number of experts (this is because for this experiment we have assumed that the activity of every expert involves either transmission or reception of messages).

In this way it is always possible to determine *a priori* whether the system is capable of both communicating information and executing in real-time. For example, if we consider Relé, the limit situations in which the two lines converge is also the limit beyond which the schedulability analysis fails. Instead, if we consider Bart, real-time scheduling is possible also beyond the intersecting point but only if we accept communication degradation.

An example exploiting the ETHNOS communication protocol to coordinate two robots for the achievement of a common goal is depicted in Fig. 4. In the example, Bart is provided with a camera but its actuators are switched off so that it cannot move. On the contrary, Homer does not have a camera but it is free to wander about. Homer uses the vision messages sent by Bart to locate the ball and, if it is sufficiently well positioned in the field, it pushes the ball towards the opponent

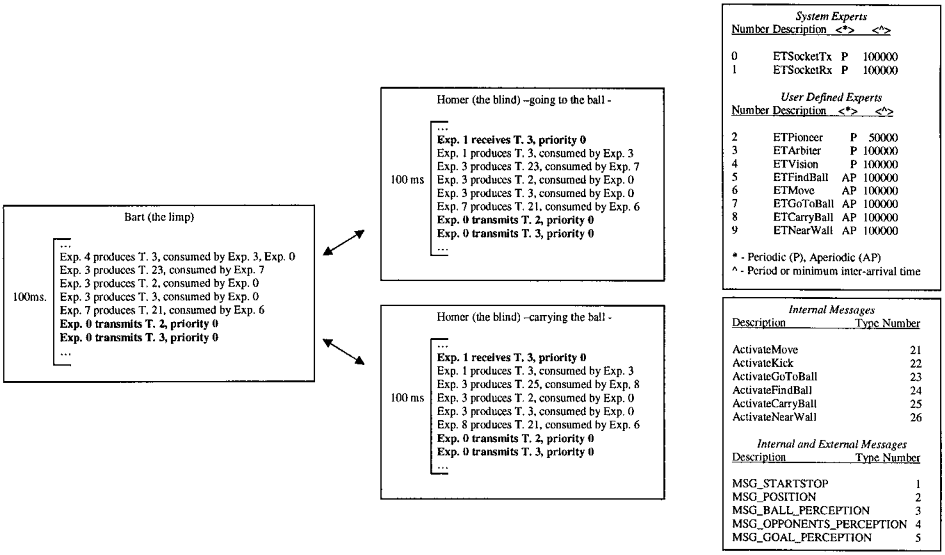

Figure 4. Message communication for robot coordination.

goal. In Fig. 4 we have emphasized the messages produced by the experts running in Homer and Bart during a 100 ms interval in two different situations: Homer going to the ball and Homer carrying the ball. On the right of Fig. 4 the experts in execution and the type of messages distributed are listed. A part of these messages is shared in broadcast (implicitly on the basis of the EIE protocol) on the local network (lines in bold in Fig. 4) and can therefore be received by the other robots or by a supervision workstation. Notice that only Bart's expert no. 4 (ETVision) produces vision messages (MSG_BALL_PERCEPTION). Such messages are received by Homer and, in particular, by the expert no. 3 (ETArbiter). In the absence of analogous locally produces messages, ETArbiter uses them to activate different behaviors (experts ETGoToBall, ETCarryBall, ETNearWall, etc.) depending on the situation in which the robot is in.

3.2. Software engineering

From the software engineering perspective ETHNOS has been widely exploited in the ART team. In fact the different research groups (belonging to different Universities or Research Institutes) involved in the RoboCup project conducted separate research activity and applied different scientific solutions. Thus expert de-coupling and object orientation have been exploited both within a research group to put together components developed by different people as well across groups to exchange components to be used in different architectures. Moreover each group also differed in the robot base used which was either an ActiveMedia® Pioneer robot personalized with additional sensors and actuators or the Mo2ro developed and used by the Politecnico of Milan.

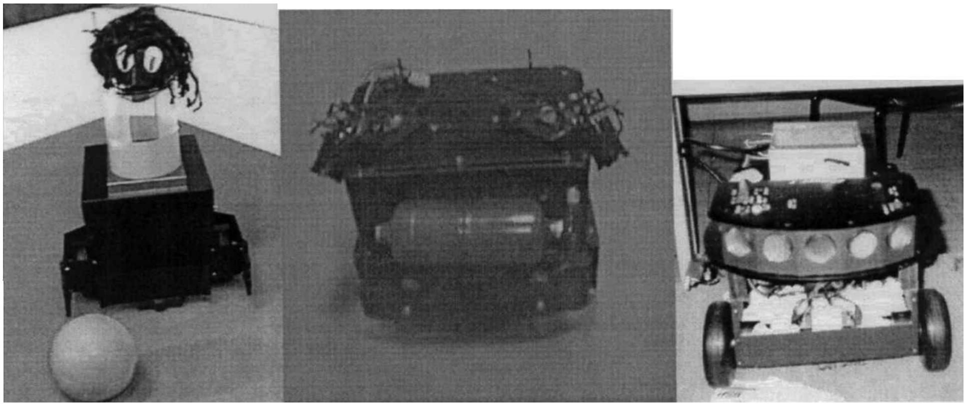

Figure 5. Left: Mo2ro Robot developed by Politecnico di Milano. Middle: TinoZoff developed by the University of Parma. Right: ActiveMedia® Pioneer used by University of Padua and Genoa.

In synthesis, ETHNOS in ART was applied to:

- Different robots: an ActiveMedia® Pioneer robot equipped with a frontal camera, a kicker and a compass, an ActiveMedia® Pioneer goalkeeper equipped with two cameras, and Mo2ro equipped with a different kicker and an omnidirectional vision sensor. Figure 5 depicts the robots used.
- Different architectures: purely behavior-based control architecture developed by the University of Padua, fuzzy logic-based architecture developed by the Politecnico di Milano and potential field based architecture developed in our laboratory.

It is worth mentioning that, despite all these differences in robots and architectures and the relatively large number of people involved (around 40), experts were successfully shared across all the different groups or subsets. Examples are the experts for robot re-localization, experts for robot coordination and experts for robot control. Moreover, inter-expert and inter-robot communication was dealt with entirely by the ETHNOS system, allowing the robots to communicate efficiently and transparently.

4. CONCLUSIONS

This paper presented ETHNOS, a programming environment for real-time control of distributed multiple robotic systems. Its properties have been described as well as its influences on the development of a soccer team for RoboCup. This year's competition in Stockholm has shown the importance of ETHNOS in the coordination of an heterogeneous team of robots independently developed. In particular, it is worth mentioning the ART managed to reach the finals, and show effective and efficient coordination despite the fact that the robots had played together less than a handful of times before the competition.

Acknowledgements

The authors wish to thank all the members of the ART Team for their significant contribution in the experimentation of ETHNOS and for some of the photos in this paper. This work has been partially funded by the Padova Ricerche Consortium in the ART Team RoboCup project.

REFERENCES

1. Special Session on Software Integration and Cross-Platform Validation in Robotics, in: *Proc. IEEE Int. Symp. on Computational Intelligence in Robotics and Automation* (1998).
2. *Proc. AAAI Spring Symp. on Lessons Learned from Implemented Software Architectures for Physical Agents*, Stanford University (1995).
3. Special Issue, *J. Exp. Theort. Artificial Intelligence* **9** (1–2) (1997).
4. Special Issue on Robotics and Communication, *Advanced Robotics* **12** (8) (1998).
5. H. Kitano, M. Asada and Y. Kuniyoshi, RoboCup: a challenge problem for AI, *AI Magazine* **18** (1997).
6. Brooks,.
7. M. Piaggio, HEIR — a non-hierarchical hybrid architecture for intelligent robots, in: *Proc. ATAL: Workshop on Agent Theories Architectures and Languages*, Paris (1998).
8. E. Gat, Integrating planning and reacting in a heterogeneous asynchronous architecture for controlling real-world mobile robots, in: *Proc. National Conf. on Artificial Intelligence (AAAI)* (1992).
9. L. E. Parker, Multi-robot team design for real-world applications, in: *Distributed Autonomous Robotic Systems 2*, H. Asama *et al.* (Eds). Springer-Verlag, Tokyo (1996).
10. A. Matsumoto and H. Nagai, The modeling of the team strategy of robotic soccer as a multi-agent robot team, in: *Distributed Autonomous Robotic Systems 2*, H. Asama *et al.* (Eds). Springer-Verlag, Tokyo (1996).
11. M. Piaggio and R. Zaccaria, An information exchange protocol in a multi-layer distributed architecture, in: *Proc. IEEE Int. Conf. on Complex Systems*, Hawaii (1997).
12. P. Stone and M. Veloso, Using decision tree confidence factors for multiagent control, in: *Proc. 1st Int. Workshop on RoboCup*, Nagoya (19970.
13. K. Jeffay, D. F. Stanat, and C. U. Martel, On non-preemptive scheduling of periodic and sporadic tasks, in: *IEEE Real Time Systems Symp.* (1991).
14. C. L. Liu and J. W. Layland, Scheduling algorithms for multiprogramming in a hard real time environment, *J. ACM* **20** (1), 46–61 (1973).
15. J. K. Strosnider, J. P. Lehoczky and L. Sha, The deferrable server algorithm for enhanced aperiodic responsiveness in hard real time environments, *IEEE Trans. Comp.* **44** (1), 73–91 (1995).
16. M. Piaggio and R. Zaccaria, Distributing a robotic system on a network: the ETHNOS approach, *Advanced Robotics* **12** (8) (1998).

ABOUT THE AUTHORS

Maurizio Piaggio was born in Genova, Italy on 1 April, 1970. He received the MS degree in Electronical Engineering from the University of Genova in 1993. In that year he worked at the LIFIA in Grenoble, France on robot manipulation and compliant control. In 1998 he received his PhD in Computer Science in Genova in the field of robot software architectures. He is currently working at the Department of Communication, Computer and System Sciences of the University of Genoa. He is the coordinator in Genoa of the Italian RoboCup Team and member of the AI*IA (Italian Association for Artificial Intelligence). His current research interests are on robot architectures, mobile robots, navigation and planning.

Antonio Sgorbissa was born in Genova, Italy on December 20, 1970. He received the MS degree in Electronical Engineering from the University of Genova in 1997. He is currently a PhD Student and Research Assistant at the Department of Communication, Computer and System Sciences of the University of Genoa, and member of the AI*IA (Italian Association for Artificial Intelligence). His current research interests are on mobile robots navigation and localization, and real-time systems.

Renato Zaccaria was born in 1949. He received a 5 years degree in Electronic Engineering from the University of Genoa in 1974. He has been Professor of 'Programming Languages' and 'Artificial Intelligence' at the Faculty of Engineering of Genoa University. Since 1991 he has been Professor of 'Robotics' at the same Faculty and Professor of 'Computer Science' at the Faculty of Humanities. He is member of the Department of Communication, Computer and System Sciences of the University of Genoa. His research interests lie in the field of advanced robotics and artificial intelligence (in particular concerning action representation, planning, navigation and autonomous agents). He is responsible for research grants from the Italian National Research Council, the Italian Ministry of Research, the EEC and private companies. He is also member of the AI*IA (Italian Association for Artificial Intelligence).