[10] Bertino, E., Mileo, A. and Provetti, A., 2003. User Preferences VS Minimality in PPDL. Proc. of *AGP03*. In press.

[11] Bertino, E., Mileo, A. and Provetti, A., 2003. Policy monitoring with User Preferences in PDL. Proc. of *NRAC03*. In press.

[12] Bertino, E., Provetti, A. and Salvetti, F., 2003. Local Closed-World Assumptions for reasoning about Semantic Web data. Proc. of AGP03, APPIA-GULP-PRODE. In press.

[13] Web location of the Milan-Messina Action Group: `mag.dsi.unimi.it`.

[14] Web location of the Psmodels solver: `www.tcs.hut.fi/Software/smodels/priority/`.

[15] Bertino, E., Cochinwala, M. and Mesiti, M., 2002. UCS-Router: A Policy Engine for Enforcing Message Routing Rules in a Universal Communication System. Mobile Data Management 2002: 8-

[16] Guha, R.V., McCool, R. and Miller, E., 2003. Semantic search. WWW 2003: 700-709.

[17] Papadimitriou, C., 2003. Lectire at Lipari Summer School. Manuscript.

[18] Web location of the most known ASP solvers.
Cmodels: *www.cs.utexas.edu/users/yuliya/*
aspps: *www.cs.uky.edu/ai/aspps/*
DLV: *www.dbai.tuwien.ac.at/proj/dlv/*
Smodels: *www.tcs.hut.fi/Software/smodels/*

[19] Virmani, J., Lobo, L. and Kohli, M., 2000. *NETMON: Network Management for the SARAS Softswitch* Proc. of IEEE/IFIP Network Operations and Management Symposium.

Author Information

Alessandra Mileo Dipartimento d'Informatica e Comunicazione Università degli studi di Milano. Milan, I-20135 Italy
`mileo@dico.unimi.it` □

ARTICLE

Multi-Agent Planning Architecture for Autonomous Robots

Author: **B. Innocenti, B. López, and J. Salvi**

Introduction

The challenge of developing autonomous robots involves several related problems as dynamical modeling of the world, task and path planning, planning and scheduling, etc..
Traditionally, each problem has been solved and implemented in a module based architecture, where the relationships among all components are established at the design time. This kind of architectures constrains in some way the possible outcome when the robot has to perform a task. This fact leads researchers either to focus on new, more flexible architectures, or to develop collections of autonomous robots that coordinate their activities to solve complex tasks. In the later, multi-agent architectures have been applied in order to have a global behavior of the agent population. An agent in such approach is equivalent to a robot.

Our proposal is the other way around and it is related to the last approach, that is, how to build robot architecture based on multi-agent system (MAS). In this proposal all the agents constitute a single robot. Agents have the same global goal: to control the robot and to do it intelligently, while fighting for resources. So, planning tasks and actions is required in order to maximize the robot performance. We believe our approach will allow to have a more robust, flexible, reusable, generic and reliable architecture that

can be easily modified and completed to permit social behavior among robots.

Background

At Girona University we apply MAS in the soccer robotic team, RoGi Team. In robotic soccer, teams are composed of 4 players and a goal-keeper. Position of robots (teammates and opponents) and the ball is obtained by a global vision system, with the camera put 3 meters above the field. This yields to have a relative global knowledge of the environment.

Figure 1: RoGI Team

As a MAS system, each robot is an intelligent physical agent with a role assigned, as in real soccer teams [1] [2] [3] [4]. Each role (goalkeeper, attacker, defense, etc.) has a set of possible actions to do (go for the ball, defense a zone, pass the ball to a teammate, etc.). In order to choose the best action to do as a whole team, the decision process of each agent takes two clearly defined steps: the first one consist in evaluate as an individual which of the possible actions is the most feasible (according to distance to the ball, opponents near robot, distance to the opponent goal, etc..) and the second one lies in the communication of this decision to teammates so as conflictive situations (as two robots going for the ball) can be avoid (based on the role of the agent in the team, position, etc.). As a result of this step one agent may have to change the selected action. Once the team reaches an agreement, each agent executes the selected action.

Our MAS lacks of team or individual planning, agents are purely reactive, due to, among others, the speed of the game; robots are extremely fast and decisions must be made in milliseconds. Soccer is a very dynamic environment.

Figure 1 shows the robots that form RoGi Soccer Team.

In spite of the soccer environment, we believe that the planning component is a key issue for any robot that acts autonomously. So, we start analyzing the robot architecture to provide such functionality.

Ongoing Research

The main feature of this research is the multi-agent architecture for controlling one autonomous robot. This MAS is composed of some specific agents namely the task planner agent, the reactive agent, and the monitor agent. These specific agents can be, at the same time, multi-agent systems. Finally, the resulting architecture is formed by several abstractions levels of MAS (see Figure 2).

First, the task planner agent is in charge of providing a plan according to some given goal or set of goals. It is, in fact, a multi-agent system form by several planner agents and a coordinator agent.

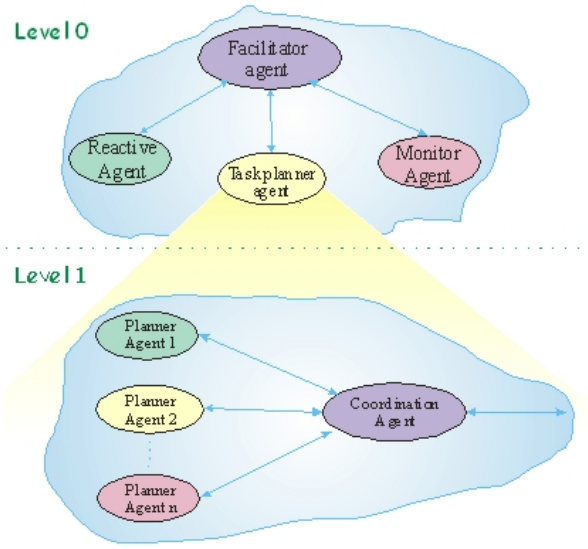

Figure 2: Task planner multi-agent system

The key agent in the task planner agent MAS is the Coordinator Agent, which asks to the planner agents for plan or a sub-plan. The idea is to have several existing planners that can be suitable for different situation (in some cases some of them can may not provide a solution or the solution may not be the best) or that can provide the solution at different time. The Coordinator Agent knows which planner is appropriate for each situation and asks it for a plan. In case of an unexpected event, it asks for re-planning or even it can ask to some planners to give a sub-plan for specific parts of the whole plan.

Figure 3: Pioneer Robot

Second, the reactive agent deals with obstacle avoidance and similar issues. Finally, the monitor agent tracks the execution of the current plan in order to detect on time, possible failures.

So as to avoid ad-hoc MAS platform, we have adopted Open Agent Architecture (OAA). This platform is developed and maintained by the SRI team [5] cite6. It presents some advantages over other multi-agent platforms including that agents can be programmed in C++, an important aspect to consider if we want to re-use some of the existing code. Another characteristic is that OAA has been integrated with Saphira. We use Saphira to develop the reactive agent and Saphira (also from de SRI team) is the software that comes with the pioneer mobile robot, the one used in this project [7] (see Figure 3).

OAA has some particular agents to guarantee a correct functioning of the platform, as can be the Facilitator agent, which provides the agent community with a number of services for routing and delegating tasks and information among agents. The role of this agent is important because it is where, upon connection, each agent registers its functional capabilities and specifications of its public data. Moreover when a request is send to the agent community specifying at a high level the description of the task along with optional constraints and advice on how the task should be resolved, the Facilitator agent distributes subparts of the task among agents and coordinates their global activity. So, Facilitator agent is the one who will have the knowledge of the different agents that form the proposed architecture, where they are, and the capabilities of each one.

Example

To clarify some ideas we propose an example. Let's consider a surveillance robot that has to move around in a building while taking pictures of some specific rooms. The map of the building is shown in Figure 4.

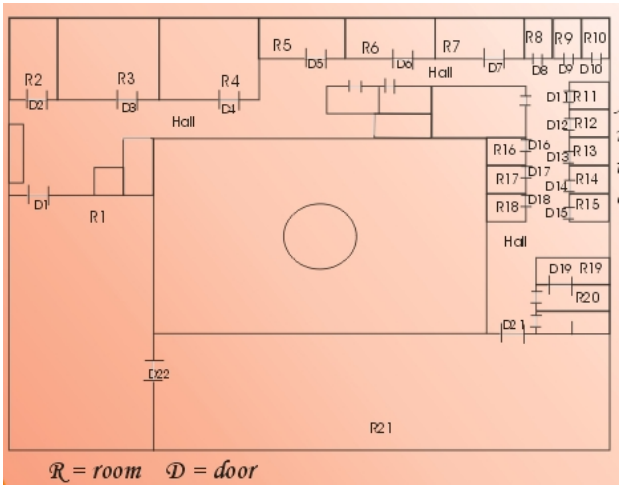

Figure 4: Floor of the building of the surveillance robot

The actions that the robot can do are: move(from,to), check-alarm and take-photo. Let's supposed that the robot is in Room R2 and must do the following plan:

```
{move(R2,Hall),move(Hall,R1),take-photo,
move(R1,Hall),move(Hall,R11),take-photo,
move(R11,Hall)}
```

The robot starts to execute the plan and when it is in Room R1 taking the picture, there is an alarm in Room R21. At this moment, the Coordination Agent asks to two proper planner agents, for a sub-plan that considers going to R21 to check the alarm. The answers are:
Planner Agent 1:

```
{move(R1,R21),check-alarm,move(R21,R1),
move(R1,Hall)}
```

Planner Agent 2:

```
{move(R1,R21),check-alarm,
move(R21,Hall)}
```

The proposed sub-plan of Planner Agent 2 may be the best because it puts the robot near the next objective that is Room 11. Under this belief the Coordination Agent can modify the original plan to be:
Final Plan (Coordination Agent): move(R1,R21),check-alarm,move(R21,Hall),move(Hall,R11),take-photo,move(R11,Hall)
And the robot continues its execution.

Conclusions

The ideas proposed in this paper are a preliminary study. At the moment, within OAA we have implemented three planner agents, by agentifying the existing planners Prodigy, Temporal Graphplan and Sensory Graphplan. Next, we will focus on the coordination agent.

Bibliography

[1] *Examples of Dynamical Physical Agents for Multi Robot Control*. Innocenti, B., de la Rosa et al. 2. Workshop Hispano-Luso de Agentes Físicos Madrid 2001. Spain.

[2] *An Example of Dynamical Physical Agents*. De la Rosa, J.Ll., Innocenti, B., Oller, A.et al. European Robocup Workshop. 2000. Holland.

[3] *Rational Dynamical Physical Agents*. De la Rosa, J.Ll., Garcia, R., Innocenti, B., et al. 3rd Robocup Workshop. Vol. on RoboCup-99: Robot Soccer World Cup III, Lecture Notes in AI No.1395, Eds. Springer-Verlag.

[4] *Rogi Team Real: Research on Physical Agents*. De la Rosa, J. Ll., García, R., Innocenti, B., et al. 3rd Robocup Workshop. Vol. on RoboCup-99: Robot Soccer World Cup III, Lecture Notes in AI No.1395, Eds. Springer-Verlag.

[5] *Building Distributed Software Systems with the Open Agent Architecture*. Martin, D.L., Cheyer A.J. and Moran D.B., in Proc. of the Third International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology, Blackpool, Lancashire, UK, 1998.

[6] *The Open Agent Architecture: a framework for building distributed software systems*. Martin D., Cheyer A.J., and Moran D.B., Applied Artificial Intelligence 13, pp 91-128, 1999.

[7] *Many Robots Make Short Work*. Guzzoni, D., Cheyer A., Julia L., and Konolige K., AI Magazine 18(1), pp 55-64, 1997.

Author Information

Bianca Innocenti Badano, Beatriz López, and Joaquim Salvi Agents Research Laboratory / Computer Vision and Robotics Group, Institut d'Informàtica i Aplicacions, Universitat de Girona
{bianca,blopez,qsalvi}@eia.udg.es □