A Layered Architecture using Schematic Plans for Controlling Mobile Robots

Mariano Tucat† Sebastian Gottifredi† Federico Vidaurreta
Alejandro J. García† Guillermo R. Simari

†Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET)
Laboratorio de Investigación y Desarrollo en Inteligencia Artificial (LIDIA)
Departamento de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur. Av. Alem 1253, (8000) Bahía Blanca, Argentina
{mt,sg,fv,ajg,grs}@cs.uns.edu.ar

ABSTRACT

Robotic soccer is a way of putting different developments in intelligent agents into practice, including not only problems such as multi-agent planning and coordination, but also physical problems related to vision and communication subsystems. In this work, we present the design used as the basis for a multi-agent system, implemented for controlling a team of robots, having as main goal to facilitate the testing of new theories developed on reasoning, knowledge representation, planning, agent communication, among others Artificial Intelligence techniques. The implementation of the system was carried out following a three-layer architecture which consists of a reactive layer, an executive layer and a deliberative layer, each of which is associated with a different level of abstraction. This layered design allows to construct a functional system with basic services that can be tested and refined progressively. We will focus our explanation on the *executive layer*, responsible for sensorial processing and the execution of *schematic plans.*

1 INTRODUCTION

In diverse research fields, systems are not always developed as a solution of a particular problem; sometimes, they are conceived in the form of testbeds for new theories, tools, and problem solving techniques. This methodology is common practice in the area of intelligent systems, in which problems with a rich structure are considered in order to address reasoning, belief revision, communication, learning, and autonomy, among other aspects. In the last few years, the game of soccer has drawn much attention in the area of multi-agent system research and development [7]. This is due to the fact that soccer is a complex and challenging domain that is useful in the evaluation of different kinds of developments that have been carried out in the field.

The game of soccer can be seen as a well defined system: the number and type of players, duration of play, allowed behaviors, and punishments (among other aspects of the game) are governed by a well defined set of rules that are known to all participants.

Partially supported by CONICET (PIP 5050) and SGCyT Universidad Nacional del Sur

However, the interaction among the players cannot be defined beforehand. Each team is composed of players that must cooperate in order to reach their goal of winning the game. They must also take into account the existence of the opposing team, which also has the goal of winning the game.

Robotic soccer is a way of putting different developments in intelligent agents into practice. This includes developments in autonomous, cooperative, competitive, reasoning, learning, and revision systems [18, 12]. Furthermore, it is useful in identifying problems related to aspects concerned with vision and communication. This type of problems cannot be all taken into account beforehand, and therefore demand that the system be designed in to be robust enough to recover from eventualities of this type. Robotic soccer is a complex domain, and it is necessary to take into account several aspects related to the robots. Each robot has sensors and effectors which are prone to failure. The environment is dynamic so there is no chance of knowing in advance the situations that can arise in a game. Therefore, it is necessary to be able to recover from adverse situations like sensorial or effectorial failure, and the decisions needed to carry out the recovery process have to be taken quickly.

In this work, we present the design used as the basis for a Multi-Agent System, implemented in order to control a team of robots playing soccer, having as main goal to facilitate the testing of new theories developed on reasoning, knowledge representation, planning, agent communication, among others Artificial Intelligence (AI) techniques. The implementation of the system was carried out following a three-layer architecture which consists of a *reactive layer*, an *executive layer* and a *deliberative layer*, each of which is associated with a different level of abstraction. We will focus our explanation on the *executive layer*, responsible for the plan execution and sensorial processing. That is, the layer in charge of controlling and determining the robots basic actions and movements, and also responsible for providing processed information about the environment.

The layered design allows the construction of a functional system with basic services that can be tested and refined progressively. As the low level service layers are implemented, the upper ones can be designed and tested using prototypes. This is also useful in testing different situations that may arise. For example, a prototype of the layers that offer services of sensorial and effectorial information can be used to evaluate and refine the design of these layers which implement the robot's behavior. The layers that are proposed as a basis for the architecture of a robotic soccer team offer a modular design. A team designed following this scheme could then be modified easily and reused in another robotic soccer league.

2 BACKGROUND AND MOTIVATION

Mobile robots involved in complex environments require high degree of intelligence or high level capabilities (such as reasoning, knowledge representation, planning, agent communication) integrated with lower level primitives (such as sensor management, basic movements, obstacle avoidance, navigation, etc.). Since 2004 we have been working in mobile robotics, specially in robotic soccer. Our previous researches were focused on those high and lower level areas. In particular, we have developed an obstacle avoiding system [15], researches on sensorial information and basic movements [14], and a multi-agent architecture to control the robots [11]. We also have developed a robotic soccer team that participated in the E-League held in Robocup 2004 [4].

We designed a new Multi-Agent System for implementing the control of the team. We developed the system following a three-layer architecture. The main goal of the design and implementation of this architecture was to encapsulate all the low level developments, including the ones mentioned above, in the lower layers of the architecture and allow an easier implementation of high level capabilities in the upper layer. Therefore, AI theories developed on reasoning, knowledge representation, planning, agent communication, among others, can be tested in this real scenario. In particular, we developed a team controlled by a BDI architecture to participate in the VI Argentine Championship of Robot Soccer (CAFR 2008 [5]).

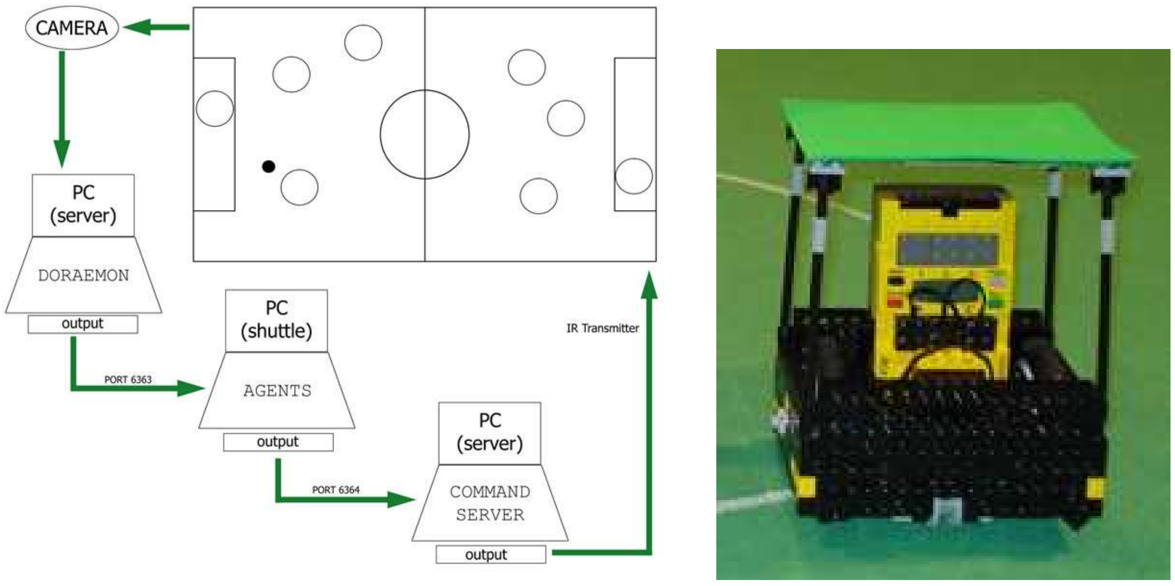


Figure 1: (a) League Setup (b) A robot built using Lego Mindstorms kits

The main goal of the leagues we participated in (E-League [6] and CAFR [5]) is to provide an environment where researchers, practitioners and students interact sharing knowledge and expertise while enjoying the games. The leagues provide common basic services to all of the participants, such as vision and communication. Teams can use low cost kits such as [3] and concentrate on the development and study of Artificial Intelligence techniques, as the ones mentioned above. The most important feature of these leagues is its simple and modular structure. There are only three basic components that must be available to obtain a functional team: a vision module that works as the robot's perception component, a communication module that allows actions to be communicated to the robots, and a control module that is implemented by agents that control the robots on the field of play.

Each team has one or more auxiliary computers in which the agents are executed. These agents communicate with the vision component in order to obtain information about what happens on the field, and send messages to the robots by means of a communication module (see Figure 1(a)). Even though the league does not define a standard platform for the construction of the robots, it does impose restrictions over the processing and memory capacity. This allows the use of low cost robotic kits, many of which fall under these restrictions. The system we developed was implemented using Lego Mindstorms kits [3], which are within the rules of the league (see Figure 1(b)). Each team is composed of three or four robots, one of the robots acting as a goalkeeper. There are restrictions over the size of the robots, their shape, and the components used in their construction. Even though the robots do not communicate amongst themselves, the processes that control them can do so.

As mentioned above, mobile robots involved in complex environments, such as robotic soccer, require high degree of intelligence integrated with lower level capabilities. Robots should be able to react quickly in a highly dynamic environment and also to reason about strategies and robot behavior at a high level. Therefore, a robot may need to decide which action (move forward, rotate, etc.) to execute next, in order to progress in the desired direction. For example, it may decide to move forward if it is facing the desired location, or it may decide to rotate in order to end looking the goal (desired location). However, it may also need to decide among different strategies or possible high level behaviors. As an example, a robot may need to decide whether to try to go after the ball or to stay in a defensive position, among other possibilities.

The system designed and implemented for controlling the robots need to take into account the characteristics of the environment, and also reaction capabilities and robot behavior requirements. Therefore, the system controlling the robots need to be able to combine reactive response with high level behavior. In the following section we will describe a multi-agent system designed to cope with these requirements, also trying to facilitate the testing of new theories developed on reasoning, knowledge representation, planning, agent communication, among others AI techniques.

3 DESIGN OF THE AGENT SYSTEM

The proposed design considers the construction of the system based on an hybrid architecture combining reaction with deliberation [16]. The most popular hybrid architectures is the three layer architecture (see Figure 2), which consists of a *reactive layer*, an *executive layer* and a *deliberative layer*, each of which covers different levels of abstraction of the problem to be solved. The *reactive layer* provides low-level control of the robot. The *executive layer* serves as the glue between the *reactive* and the *deliberative layer*. It accepts directives by the *deliberative layer*, and sequences them for the *reactive layer*. The *deliberative layer* is responsible for controlling the robot behavior by taking high level decisions.

As will be explain next, each layer provides services to the upper layer, which are implemented using those provided by the lower layer. Therefore, this layered design allows local modifications. Thus, the implementation of some services can be modified without provoking many changes in other layers using these services.

3.1 Reactive layer

We associate with this layer the program running inside the robots and implementing the basic actions they need to be able to act in a dynamic environment such as robotic soccer. This layer also includes the basic hardware and software support that is provided by the league. This involves physical support, such as infrared transmitters, video camera, communication network, and common software. The software that is provided by the league includes video and command communication servers.

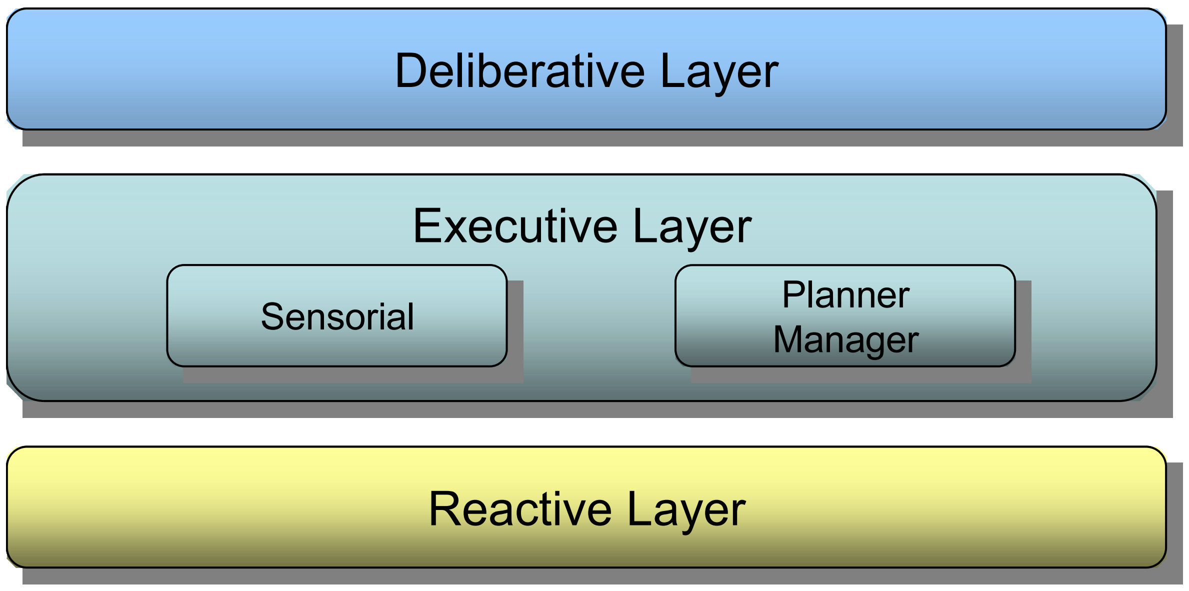


Figure 2: Architecture used for the implementation of the Matebots team.

The basic actions represent the minimal unit of change a robot may try to execute in order to modify its environment. The possible basic actions of any robot are directly related to its shape, design and capabilities. In our case, our robots have only two motors connected to their corresponding wheels, allowing them to describe different kind of movements. Next we will define the syntax of the possible basic actions a robot may be able to execute.

Definition 1 (basic actions)
Let $Anam$ be a set of action names each with a given arity. Let $Apar$ be a set of action parameters. Then, $a(p_1, ..., p_n)$ is a *basic action* with $a \in Anam$ and $p_1, ..., p_n \in Apar$ where n is the arity of a. The set of basic actions is denoted by $Actions$.

As we will explain in the following subsection, these basic actions are provided by this layer to the executive layer, in order to allow the construction of specific sequences of actions with a determined goal, such as going to the ball. The basic actions implemented inside the robots include three generic movements: moving forward and backward, rotating clockwise or counterclockwise and describing different kinds of arcs. These actions may also vary in the velocity of the wheels, allowing the robots to execute movements with different speeds and precision.

This layer also includes the basic perceptions. The basic perceptions represent the information any robot may obtain from the environment. This information may correspond to the location of the objects that are part of the environment. It may also represent the orientation and velocities of these objects. Next we will define the syntax of the possible basic perceptions a robot may obtain from the environment.

Definition 2 (basic perceptions)
Let $Pobj$ be a set of perception objects each with a given arity. Let $Ppar$ be a set of perception parameters. Then, $o(p_1, ..., p_n)$ is a *basic perception* with $o \in Pobj$ and

$p_1, ..., p_n \in Ppar$ where n is the arity of o. The set of basic actions is denoted by $Perceptions$.

The video server, called *Doraemon* [13, 8], is part of the software support provided by the league. A video camera covers the field of play and this server processes the images that it obtains, generating information packets that are then made available to be used by the agents that control both teams' robots. The packets that are generated by the video server provide information about the objects that are on the field of play. Information about the position, orientation, and speed of these objects (the robots and the ball) is transmitted. This information will be captured and processed by the executive layer, as we will explain in the following subsection.

The league also provides a command communication software called *Command Server* (abbreviated CS from now on), which allows the agents, who control the robots, to send messages to the field. As we have mentioned, the processing and memory capabilities of the robots is limited, and the control software must therefore reside and execute on auxiliary computers. In this way, the decision processes are carried out by these agents in the upper layers, and the decisions are then communicated to the robots through the CS. The decisions communicated to the robots are the actions to execute, which, as we mentioned above, are implemented in the program running inside the robots. The frequency by which an agent can send actions to the robots through the CS is limited by the physical characteristics of the transmission method used by the robots, in our case Lego Mindstorms kits [3] using IR.

3.2 Executive Layer

This layer serves as the glue between the *reactive* and the *deliberative layer*. It accepts directives by the *deliberative layer*, and sequences them for the *reactive layer*. It also provides perception information about the environment to the *deliberative layer* in order to allow it to reason/decide about the robot behavior. Therefore, this layer is divided in two sub-layers, the *planner manager sub-layer* and the *sensorial sub-layer*.

The *planner manager sub-layer* provides a set of implemented *schematic plans* allowing the *deliberative layer* to control the robot behavior without worrying about low level details. Thus, this sub-layer provides the ability to select and execute actions by planning, and performing such actions as a matter of plan execution. However, this *planner manager* does not decide which *schematic plan* to execute, it only obtains the actual *schematic plan* and selects the best action to perform next in order to accomplish the desired goal of the plan.

A *schematic plan* represents a plan in a highly dynamic environment, in which the sequence of actions needed to achieve the desired goal may vary at the moment of executing the plan. Therefore, these *schematic plans* are divided in several atomic actions, in particular, the basic actions described in the *reactive layer*, and each of these actions depend on the state of the field at the moment immediate before of been executed. Next we will define the structure of the *schematic plans*.

Definition 3 (schematic plans structure)
Let β be a well-formed formula. Let $Actions$ be a set of basic actions. Then the structure of a schematic plan, denoted by $PlanS$ is defined as follows:

- $Actions \subseteq PlanS$
- if $\pi, \pi' \in PlanS$, then π**;**$\pi' \in PlanS$
- if $\pi \in PlanS$, then **while** β **do** $\pi \in PlanS$
- if $\pi, \pi' \in PlanS$, then **if** β **then** π **else** $\pi' \in PlanS$

where **;** is a sequential operator, the **while-do** construct is an iteration operator, and the **if-then-else** construct is a conditional choice operator.

This definition shows all the possible *schematic plans* and the syntax used in their definition. Therefore, any *schematic plan* will be specified following the schematic plan structure defined above. Next we will show an example of a *schematic plan* whose goal is to take possession of the ball. In this *schematic plan*, the robot may need to rotate in order to be oriented in the direction of the ball, and then move forward to grab the ball. However, the ball may change its location, and the robot may need to reorientate itself. Note that this *schematic plan* does not consider the possibility that another robot may interfere in the path towards the ball, thus, it will not avoid any obstacle.

Example of a schematic plan: goto(ball)

```
while (not have_ball(robot)) do
      while (not oriented_to_ball(robot)) do
            if (orientation(robot) > desired_orientation) then
               rotate_right
            else
               rotate_left
      ;
      move_forward
```

The current *schematic plans* implemented for the domain of robotic soccer are:

- go to a given object, such as another robot or the ball,
- pass the ball to a teammate,
- go to a defensive position,
- kick the ball,
- dribble to a given location, and
- clear the ball out of the defensive zone.

The *planner manager sub-layer* contains a queue of *schematic plans* in execution and provides a set of services that allow the *deliberative layer* to handle this queue. Essentially, it allows the upper layer to queue *schematic plans* and also to remove all the queued *schematic plans*. This sub-layer is responsible for obtaining the actual *schematic plan* and executing the actions chosen by the selected *schematic plan*. Whenever the *schematic plan* accomplish its goal, the *planner manager sub-layer* removes the *schematic plan* from the top of the queue, in order to continue executing the next *schematic plan* in the queue. Note that, in the case that the upper layer removes the current *schematic plan* in execution (by removing all the queued *schematic plans*), it will be automatically destroyed and the *planner manager sub-layer* will wait until the upper layer adds a new *schematic plan*.

In the *sensorial sub-layer*, the visual information is processed and translated into information that express states of the world. This information is divided in basic perception and inferred information. The basic perception, defined in the previous subsection, corresponds to the coordinates, orientation and speeds of the robots and the ball. In the case of the inferred information, it is composed by elements obtained using inference rules over the basic perception.

The inferred information represent high level knowledge the upper layer may use in order to decide the behavior of the robots. The goal of this kind of information is to allow the *deliberative layer* to obtain processed information, facilitating the use of declarative languages in the implementation of the agents controlling the robots. Examples of inferred information provided by the *sensorial sub-layer* are:

- The robots' and the ball's locations relative to the field,
- player and/or team that is closest to the ball,
- distances between different objects on the field, between players, rival players, etc.,
- whether the ball is moving and in which estimated direction,
- whether any robot is blocked, unable to move towards it goal.

These information is provided to the *deliberative layer* as PROLOG predicates, and they are implemented by querying and analyzing the basic perceptions obtained from the video server. Then, the situations modeled through these predicates are used in the upper layer to model and implement the team's game strategy.

The *executive layer* is implemented as an interface between the *reactive layer* and the *deliberative layer*. As shown in Figure 3, the *sensorial sub-layer* obtains basic information from the *reactive layer* and provides inferred information to the *deliberative layer*, and the *planner manager sub-layer* allows the *deliberative layer* to handle the *schematic plans* queue, transforming these *schematic plans* in basic actions for the *reactive layer*.

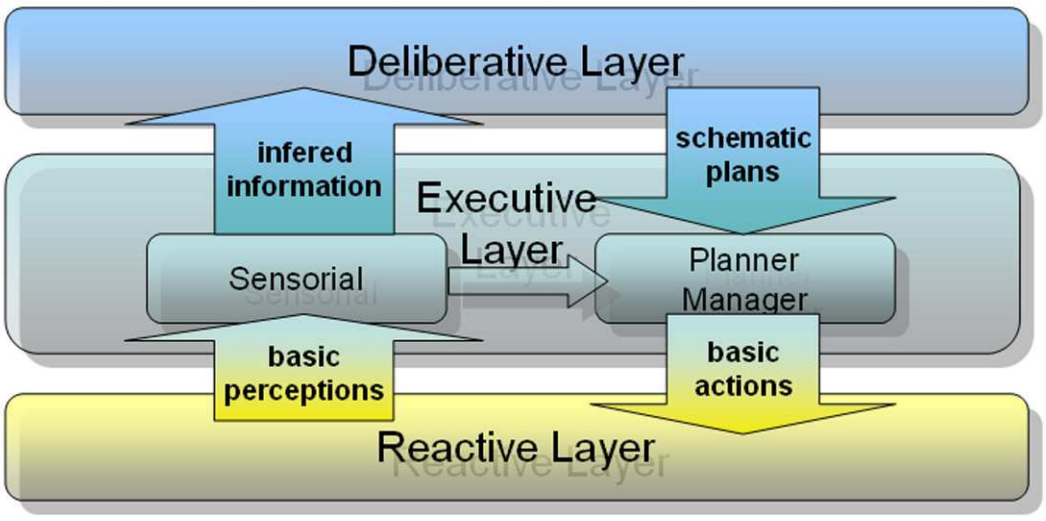


Figure 3: Interaction between the executive layer and the upper and lower layers.

The entire layer has been developed in the *C++* programming language providing PROLOG predicates. This decision has several advantages. As we mentioned before, the environment is highly dynamic, which causes the states of the world to change quickly. Therefore, it is necessary for the robots to be able to react accordingly to this dynamism.

Moreover, the information obtained by the robots can be wrong due to sensorial failure; after recovering from such a failure, the current situation could be completely different from the one previously perceived. In these cases, the system has to be able to analyze new situations, and quickly decide which actions should be taken by the robots.

The existence of this layer allow us to disregard the physical structure of the environment in which the team of robots is embedded. For example, it is possible to implement the services of this layer based on other environments, even simulated ones like the *FIRA SimuroSot*. If the interface of the services offered by this layer remain unchanged, then the *deliberative layer* can also remain unchanged.

3.3 Deliberative Layer

This layer is responsible for the design and implementation of the agents that control the robots behavior. It allows the use of different knowledge representation and reasoning systems. The lower layers provide high level, processed information to this layer, allowing it to decide the behavior of the robots, and control them. Therefore, this layer can be constructed using a simple behavior cycle, as shown in Figure 4, in which the agent sense the environment, decide which plan to perform and execute the chosen plan. Another alternatives corresponds to use more sophisticated agent architectures.

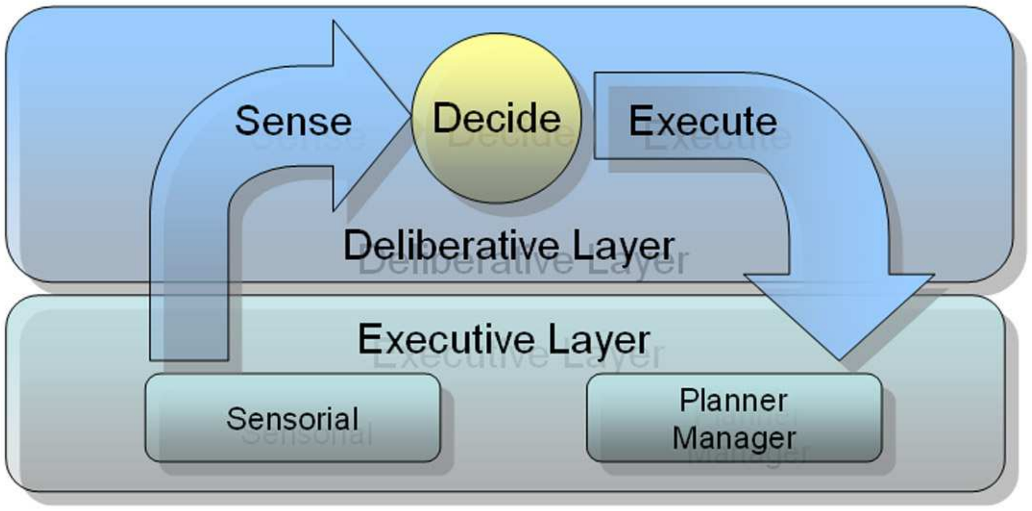


Figure 4: Perception/action loop.

In particular, we developed a team controlled by a Belief-Desires-Intentions (BDI) model [10] to implement the agents controlling the robots, to participate in the VI Argentine Championship of Robot Soccer (CAFR 2008 [5]). In this case, as in any other agent system, the agents perceive information about the environment and reason/decide about the *schematic plans* to take. In particular, in this proposal each agent has a *set of* *beliefs*, representing the perceived and inferred information obtained from the *executive layer*; a *set of desires* or goals, representing what the agent is trying to achieve; and a *set of intentions*, representing the *schematic plans* that the agent have to achieve its desires, implemented in the *executive layer*. These sets are used by the agent to determine the best course of action to achieve its desires.

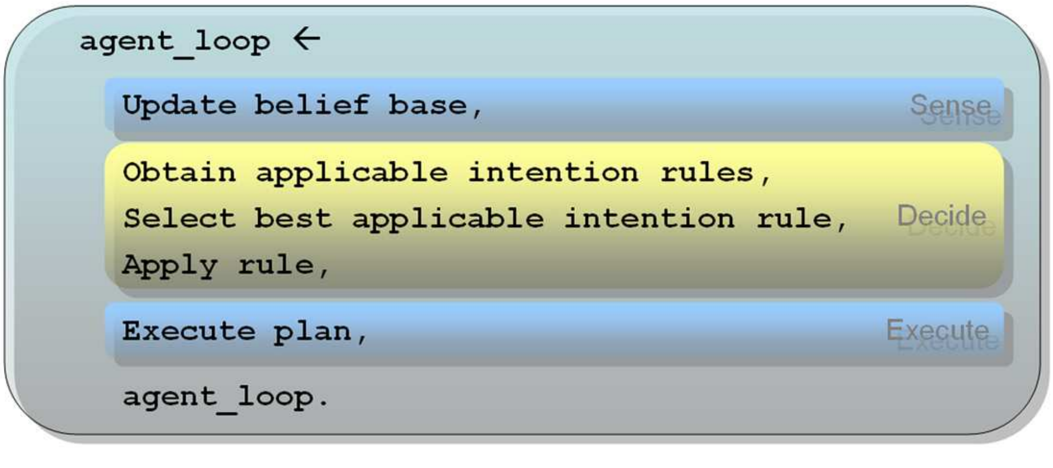
The *belief set*, or belief base contains all the information that the agent obtains from the field each deliberative cycle provided by the *executive layer*. As explained in previous subsections it is divided in basic information (elements directly perceived from the field like X,Y coordinates) and inferred information (elements obtained using inference rules that contain the basic information, like distance between to objects in the field).

The *desire set*, or desire base contains the goals/desires that the agent wants to achieve. This set will change among the life span of the agent. For instance desires can be *Defend*, *Attack* or *Score A Goal*. The elements of this set can have a priority attached, in order to help the agent to determine which one to achieve first.

The *intention set* is divided in two sub sets: the intention rule base which is a set of rules representing the intentional model of the agent and the currently executing intention which is the set of *schematic plans* that the agent is currently executing. In the intention rule base, each rule has a desire, a set of preconditions representing the trigger to fire the rule, and a set of *schematic plans* that will be executed if the rules is fired in order to achieve the desire.

For instance, one rule to achieve the desire *Score A Goal*, can have the following preconditions: carrying the ball, the opponent goal keeper is away from the goal and the running *schematic plan* is *dribble to the opponent penalty area*. Therefore the associated *schematic plan* will be: *Shoot for goal!*. Thus, when preconditions of this rule are met, it will be *applicable*. When the agent applies a rule, the plan of the rule will be executed.

In order to do this, each agent is modeled through a perception/action loop of the form:



```
agent_loop ←
    Update belief base,
    Obtain applicable intention rules,
    Select best applicable intention rule,
    Apply rule,
    Execute plan,
    agent_loop.
```

The BDI architecture represents one alternative cognitive system to implement the robot behavior. However, there exists another alternatives, such as KARO [17] (Knowledge, Abilities, Results, and Opportunity). The core logic consists of dynamic logic, used to model abilities and opportunities, and S5 logic, used to model knowledge. Another possibilities corresponds to use reasoning, planning, agent communication, among others AI techniques to improve and enhance the intelligence of the agent controlling the robots.

4 CONCLUSIONS AND FUTURE WORK

In this work, we have presented a Multi-agent system for controlling a robotic soccer team, implemented following an hybrid three-layered architecture combining reaction with deliberation. This design allows the abstraction and modularization of the different aspects of the complex domain that robotic soccer represents. The main goal of the design and implementation of this architecture was to encapsulate all the low level developments in the lower layers of the architecture and allow an easier implementation of high level capabilities in the upper layer.

The hybrid architecture used is composed by three layers consisting of a *reactive layer*, an *executive layer* and a *deliberative layer*, each of which covers different levels of abstraction of the problem solved. In this work, we focused our explanation on the *executive layer*, responsible for the plan execution. This layer provides the upper layers the capability of obtaining high level, processed information and also the ability of controlling the robots through the execution of *schematic plans*.

In the chosen architecture, the *deliberative layer* is responsible for controlling the robot behavior by taking high level decisions. This layer allows the use of different knowledge representation and reasoning systems. In particular, we developed a team controlled by a BDI architecture in order to participate in CAFR 2008. We are currently working on the development of more complex agents that incorporate a wide range of AI techniques developed within LIDIA from the fields of planning, argumentation, learning, belief revision, and game theory and decision theory, among others.

References

[1] http://brickos.sourceforge.net/. First open-source operating system for the Lego Mindstorms RCX Controller.

[2] http://cs.uns.edu.ar/∼gis/robocup-tdp.htm. English version of the official Matebots E-League team webpage.

[3] http://www.legomindstorms.com. Lego Mindstorms robots and RCX controllers.

[4] http://www.robocup2004.pt/. Official webpage of Robocup 2004. Lisboa, Portugal.

[5] http://www.uncoma.edu.ar/cafr2008/. Official webpage of the VI Argentine Championship of Robot Soccer.

[6] Official E-League webpage. http://agents.cs.columbia.edu/eleague/.

[7] S. Achim, P. Stone, and M. Veloso. Building a dedicated robotic soccer system, 1996. In Proceedings of the IROS-96 Workshop on RoboCup.

[8] John Anderson and Jacky Baltes. Doraemon user's manual. http://sourceforge.net/projects/robocup-video.

[9] John Anderson, Jacky Baltes, David Livingston, and Elizabeth Sklar. Toward an undergraduate league for robocup. In *Proceedings of the RoboCup Symposium*, 2003.

[10] M. Bratman, D. Israel, and M. Pollack. Plans and resource-bounded practical reasoning. In *Philosophy and AI: Essays at the Interface.*

[11] Alejandro J. García, Gerardo I. Simari, and Telma Delladio. Designing an agent system for controlling a robotic soccer team. In *Proceedings of X Argentine Congress of Computer Science*, page 227, Buenos Aires, Argentina, 2004. Universidad Nacional de la Matanza.

[12] Kwun Han and Manuela Veloso. Automated robot behavior recognition applied to robotic soccer. Robotics Research: the Ninth International Symposium, pages 199–204. Springer-Verlag, London, 2000. Also in the Proceedings of IJCAI-99 Workshop on Team Behaviors and Plan Recognition.

[13] Benn Vosseteig Jacky Baltes and John Anderson. Robocup e-league video server. http://sourceforge.net/projects/robocup-video.

[14] Fernando Martín, Mariano Tucat, and Alejandro J. García. Soluciones a problemas de percepción y acción en el dominio de un equipo de fútbol de robots. In *Proceedings of X Argentine Congress of Computer Science*, pages 1895–1906, Buenos Aires, Argentina, 2004. Universidad Nacional de la Matanza.

[15] Nicolás Rotstein and Alejandro J. García. Evasión de obstáculos con bajo costo computacional para un equipo de fútbol de robots. In *Proceedings of X Argentine Congress of Computer Science*, Buenos Aires, Argentina, 2004. Universidad Nacional de la Matanza.

[16] S. Russel and P. Norvig. *Artificial Intelligence: a Modern Approach.* Prentice-Hall, 1995.

[17] B. van Linder, J.-J. Ch. Meyer, and W. van der Hoek. Formalizing motivational attitudes of agents using the KARO framework, February 27 2001.

[18] Gerhard Weiss. Learning to coordinate actions in multi-agent systems. In *Reading in Agents*, pages 481–486. Morgan Kaufmann, 1998.