Building Real Time Agents using Parallel Blackboards and its use for Mobile Robotics

Michel Occello [*], Yves Demazeau [†]
LIFIA/IMAG/CNRS
46, avenue Félix Viallet
38031 Grenoble Cedex FRANCE
Michel.Occello@imag.fr - Yves.Demazeau@imag.fr

Abstract : Vehicle intelligent control is a robotics real time application where elaborated reasoning process and reactive process work together and have to cooperate or more to be interdependent. Multi-agent systems are well suited to such complex systems specification, because of their software engineering, their reasoning capabilities (from a cognitive point of view) or their performances (from a reactive point of view). This paper proposes a model of agent integrating both reactive and deliberative capabilities adapted to real time context. The paper introduces and discusses the use of a parallel blackboard architecture to support the agent model in order to meet real time constraints. An illustration of the functioning of the agent architecture is given through a roadway traffic scenario emphasizing the main aspects of real time distributed decision making.

1 Introduction

Most real time systems can be seen as realising three steps : data acquisition, data processing and output providing to human-machine interfaces or physical devices. Real time intelligent systems for process control or operator assistance aim to supervise external physical devices with the help of computers. They have to work in relation with the real world. Agents systems are well suited to applications in this domain, because they offer modularity, flexibility, and concurrency. But in order to realise real time applications, agent systems have to be improved to take into account the evolution of the corresponding real environment.

Agents have to modify their behavior. They have to be able to adapt their plans according to the dynamic modification of the real world. To build real time multi-agent systems, we have to integrate in a same agent, cognitive capabilities (symbolic reasoning, social behavior) to ensure the best cooperation between agents, and reactive capabilities to follow the evolution of the environment as shown in [2].

We propose in this paper a model of an agent integrating both reactive and cognitive aspects. This agent architecture is supported by a generic parallel blackboard model which supplies a powerful architecture to implement agents. In the first part, we give an insight to some previous work that defines requirements for a cognitive /reactive agent and proposes elements for an agent model. Then we discuss this approach. In the second part, the use of parallel blackboard architecture for this type of agent is argued. The proposed architecture of cognitive/reactive agent is detailed in the third part. Finally, the activity of the resulting parallel blackboard model of agent is exposed in the last section, in a real time scenario of roadway traffic.

We conclude by some perspectives about the use of parallel and distributed blackboards to build agent systems and about the integration of such blackboard agents in a multi-agent integration platform.

2 Previous work

In [2], S. Bussman and Y. Demazeau have shown that especially in complex domains neither purely reactive nor purely cognitive approaches suffice to meet the requirements imposed by the environment.They extract the main requirements to integrate reactive and cognitive aspects in a same agent, and propose specifications for a reactive / cognitive agent. We emphasize in the first part the requirements they established and then we give an insight to the model they have proposed.

2.1 Requirements

Cognitive agents in a multi-agent world employ very general, and therefore powerful methods, but suffer of slow algorithms, due to their complexity. Furthermore,

[*]Maître de Conférences à l'Université Pierre Mendès-France (UPMF)-Grenoble II

[†]Chargé de Recherches au Centre National de la Recherche Scientifique (CNRS)

these algorithms, as they are designed, do not take into account unpredicted events, apart from execution failures which are only detected but not evaluated. These two severe drawbacks for cognitive agent systems are exactly the advantages of the other reactive agents in a multi-agent world : reactivity and fast action selection. On the contrary, it seems difficult to encode complex behaviour into a rule system that is only based on perception. Consequently, one would like to incorporate the advantages of both into a single model. In the following, the requirements stated for simple reactive-cognitive integration are listed.

Reactivity: An agent should be able to *react* upon *unpredicted* events by adapting its behaviour. The reactivity of an agent is located in the interpretation of its environment. Unpredicted events have to be recognized before one can react. The recognition is followed by an evaluation which determines the internal changes that the agent has to take in order to react to the new situation accordingly.

Adaptation of Behavioural Speed: An agent should be able to adapt its behavioural speed to the evolution of the system. This requirement covers two aspects. First, perception should be sufficiently fast in order to take a consistent picture of the system's state. Second, the computation and the execution of the appropriate action should be accomplished before ongoing changes make it obsolete. Reactive models show very fast behaviour by the use of rule systems. As stated in our requirement, we would like an agent not only to be fast in his actions but also adaptive to the rate of change of the system.

Symbolic Representation and Reasoning: An agent should be able to manipulate explicitly its knowledge about the universe. This offers us the classical techniques of AI, such as deduction, planning, and learning. These have been used by all cognitive models as a basis for an agent. But their advantage of generality is compensated by the absence of reactivity and of adaptation which we have shown to be indispensable requirements for multi-agent systems.

2.2 Elements for a model

It will be the goal of the model proposed to accomplish the integration of the three requirements formulated above into a consistent agent model.The task of an agent model is to describe how the input data is processed in order to determine the output of the agent (see figure 1). In the model the scope of the input data is restricted to digitized information which are called *sensory input*. Analogously, the output of an agent is called *actions*. From this point of view, the terms *perception* and *execution* refer to the processes necessary to interact with the environment.

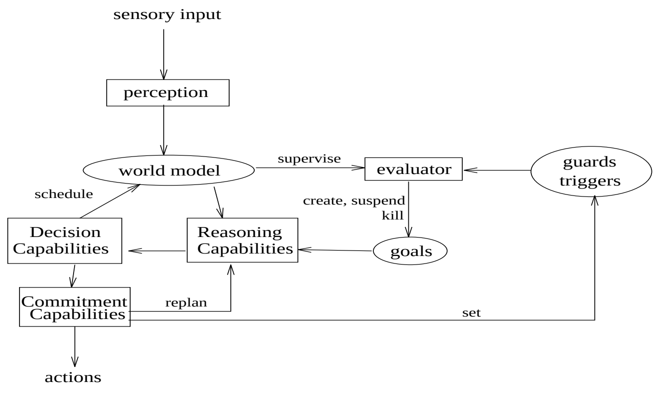

Figure 1: Agent Model

This previous model is the base of the agent model presented in our paper. Its main aspects are discussed in the next part where we propose improvements to obtain an adapted agent architecture.

2.3 Discussion

Adding communication. This work aims to describe how the input data is processed in order to determine the output of the agent. It restricts the scope of the input data to information called sensory input obtained from the environment. We have to take into account in the same way the information from the other agents. Analogously, actions are supposed to interact with the environment, the model has to consider the interactions with the agents too.

Real time agent control. Our objective is to extend this model to obtain a model really applicable to real time systems. A real time agent can be viewed as a process control system that aims to supervise a physical device in its environment. The state of the physical device is expressed through a set of measures and events obtained from sensors. This state is continuously modified, witness to the system evolution. The supervision of process consists in recording and maintaining a view of the controlled system (for an operator for example) according to the modifications of the state. If the control system has to pilot the physical device, it has to react to these modification by emitting orders to physical effectors. Response time must be viable with regard to the evolution of the controlled device.

Stressing the reactive behavior. As opposed to *transformational* program which have to provide results from input data and evolve at their own rythm, real time programs have to be *reactive* i.e. subject to the evolution of their environment to which they are related and which fix their cadence. Unfortunately, control systems (and specially knowledge based ones) cannot be totally situated in these classes, because they use transformational programs (algorithms) called by a reactive kernel.

Validation through the organisation. The previous model proposes a collection of transformational and reactive modules, but doesn't explicit which organisation between these modules can guarantee a satisfying global real time working. At this sense, a centralized control approach seems to be better adapted, it is the paradigm proposed by such systems. It needs to introduce a synchronous control unit which ensure reactivity in the decision making caused by an evolution of the environment [7].

Parallelisation and synchronisation. However, using agents which interact in real time with an environment pose problems of parallelism and synchronization in their structures. Parallelism is an imperative need for a real time system :

- either because the physical device consists of distributed equipments,

- either for security, reliability or efficiency.

In summary, we will propose in the following sections to work towards a complete model of agent, integrating :

- A symbolic representation of both the environment and the other agents,

- A sensing mechanism for both environment and agents events,

- A graduated adaptation of the behaviour in reaction to these events,

- An organisation for this agent supplying parallelism and reactivity mechanisms in order to satisfy real time constraints.

A parallel blackboard model is proposed to achieve this purpose. This choice is argued in the next section.

3 Why a parallel blackboard to support a real-time agent ?

This section presents a basic parallel blackboard model initially proposed in [7]. We will later discuss its adaptation to requirements for a real time agent.

A parallel blackboard model :

Classically, a blackboard system consists of a data structure called blackboard, several modules, a control mechanism.

A parallel blackboard architecture aims to really express the inherent parallelism of the conceptual blackboard model [3].

Modules react towards modifications of the blackboard, for their activations and inhibitions. They work on a local context which is a part of the blackboard data. *The blackboard contains domain data (used for problem resolution) and control data (summary of the state of the resolution). A control mechanism is in charge of the communications between modules and of control of the management of modules activity* (figure 2).

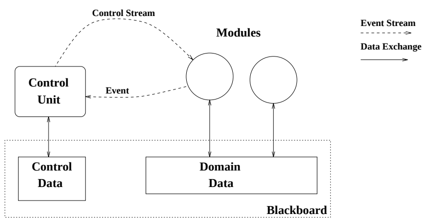

Figure 2: An Event Driven Parallel Blackboard System

Modules emit events to the control unit or modify the blackboard data. Main sorts of events are **modification** and **signal**. A modification event implies an access to the domain blackboard.With signals events, *modules communicate their state to the control unit.* Events only contain immediate information. This distinction between data flows and control flows is also made in non-blackboard agent architecture as [1]. The control unit receives events from modules and emits control signals to them. Common types of control signals are **activation signal**, **inhibition signal**. Modules which have all their conditions validated are activated by an activation flow. Inhibition signals trigger exceptions processing in the modules. The control unit is application independent. It is built and formalized as in [7].

Adequation with requirements :

A complete study of the use of blackboard systems for dynamic systems control [7] proposes a few privileged points to adopt the blackboard concept to organize a real time agent :

1. A real time system uses for many reasons a distributed architecture : it seems necessary to adopt *a parallel blackboard model.* A parallel blackboard system uses a functional decomposition of the control of systems where modules represents different tasks of organs of the physical device, as control functions of effectors or access to sensors constituting a heterogeneous architecture.

2. A blackboard system is no longer seen as a transformational system but as a reactive system : *decision making aims no longer to find the best solution in a lim-*

ited time but to respond to an event in a limited time which depends of the context.

3. The evolution of the state can question the activity of some modules : *modules must be sensitive to interruption signals.* The blackboard becomes a dynamic tool which coordinates and schedules modules and ensures synchronization.

4. Parallelism allows by the asynchronous execution of modules *to acquire data in real time through specialized modules that constitute the interface with the real world.* The software manages the activity of physical modules (directly related to process) and logical ones (for the system evolution) through the data structure representing the world. The structure represents the state of the environment and of the controlled system. Modules modify the state of the structure which evolves and offers at each time the supervision of the process.

5. The control mechanism must choose modules whose activity can be performed in the right delay, but *time-out tests must be provided to suppress modules whose duration are too expensive according to their deadline.* Separation between the blackboard control mechanism and the domain activity offers opportunities for action scheduling under temporal constraints.

6. Control complexity must be reduced by the *elimination of competing interpretations and reliable data filtering.* Recording the modification of the environment on the blackboard and the choice of action in function of the state of the blackboard provides an interesting mechanism for reactivity.

This management, control mechanism, must be transparent as in any programming systems. Such an interpretation of the concept is used for sensors fusion and multi-robots coordination.

4 A model of agent integrating cognitive and reactive aspects

The integration of cognitive and reactive capabilities is possible only with the use of parallelism in the structure of the agents. Separation between Decision/Reasoning and Perception/Communication tasks allows a continuous supervision of the evolution of the environment. The reasoning model of our agent is based on the

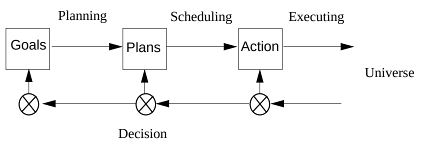

Figure 3: Control Process of the Agent Model

Perception/Decision/Reasoning/Action paradigm. The cognitive reasoning is so preserved. Predicted events contribute to the normal progress of the reasoning process. Decision modules evaluate the importance of the unpredicted events and the obligation to place new goals on the mental model. New goals imply the activation of reasoning modules to partially or totally replan according to the importance of the event. The agent control process can be explicited by the figure 3.

We describe now the different modules needed by a cognitive/reactive agent proposed by [2], organized according to the blackboard model presented below. The

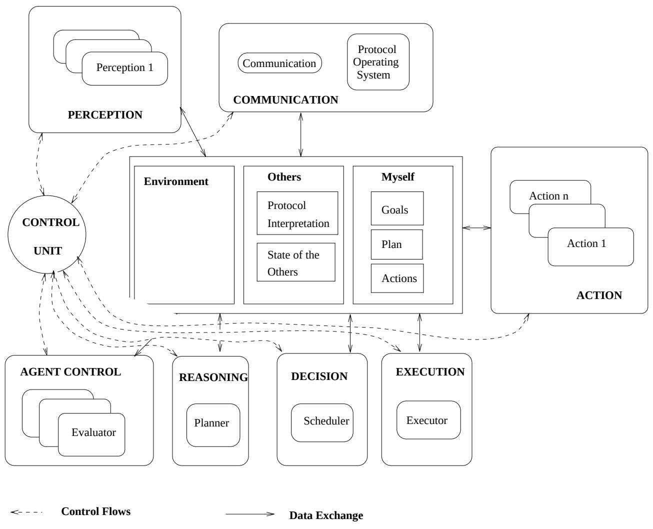

Figure 4: A cognitive/reactive parallel blackboard agent

centre of the agent is its *world model.* This model comprises its knowledge about **the environment, the mental states of other agents, and its own mental state**. The proper mental state includes in particular the plans that are being executed or which the agent takes into consideration. This model is maintained by an interpretation process of the sensory data. This model constitutes the **domain blackboard**. Evolving in a real world, each agent must integrate capabilities of perception realized by sensor devices. The knowledge about the environment is build by **PERCEPTION** modules.

To ensure the reactivity of the agent, an evaluator continuously examines this world model. **AGENT CONTROL modules** detect situations to which the agent needs to react, evaluates them, and decides to take the appropriate actions which may be of the form to create, suspend, or kill goals, i.e. to change the context of the planning and executing process. The continuous supervision of the agent's situation ensures that the agent can react to unpredicted events at any time. The same mechanism takes into account interaction with other agents using the interaction protocols proposed by [4] (**COMMUNICATION modules**). Furthermore, the planner has the possibility to include internal actions, such as replanning or the setting of *guards* and *triggers*, in order to be more adaptive at run time. Guards and triggers, supply information about the current situation during the execution phase of actions, they are stored in the domain blackboard.

Whenever a goal is created (or modified) a plan is searched that achieves the goal, this task is realized by **REASONING modules**. Plans relative to each known goal are stocked in the part of the blackboard concerning the mental state of the agent. The planner details the action in the order in which they will be performed. This process may be guided by hierarchical planning that attempts to infer the action sequence in a top-down fashion. In simple applications, we can assume that the agent have plans for all encountered goals and possesses all needed actions; the planning process is then reduced to a fast pattern-matching algorithm.

Constructed plans are scheduled consistently in the blackboard. Due to this technics, if the agent has to act fast, the scheduling and the execution of an incomplete plan can start before the planning process is finished. This ensures the adaptation of the agent to the evolution speed of his environment and is necessary if the agent pursues several goals at the same time. The committed actions are performed at the scheduled time by the **ACTION modules**.

All modules are managed by the control unit, the global model is presented in figure 4. This multi-modules approach allows a modular and independent description of each of the action and perception tasks in separate modules.

5 Application to mobile robotics

We illustrate in this section how runs the agent architecture by an application to mobile robotics agents. The next situation example emphasizes the agent organization and its behavior in front of traffic. They show the integration of deliberative and reactive aspect of the agent decision making according to the progressive decision process exposed in the previous section.

Let us assume a robotic agent on a road. In order to plan or to achieve symbolic decision making tasks, This agent needs cognitive capabilities. Thus it must have a representation of the environment. But as it must control a physical device (the vehicle) in a real time scenario, this representation must be dynamically maintained, to respond to unpredicted events.

A normal working mode consists in the planning of trajectory in function of a goal. A goal is expressed by a final position, speed and orientation for the robotics mobile device on the road map. A nominal path is calculated by a dedicated module and placed in a goal tree.

According to the model of agent proposed in the previous section, we can describe an agent structure for this application. The *Planner* Module is triggered by a new goal (i.e by the reception by the blackboard control unit of an *Event(New_Goal)* signifying a modification on the shared blackboard). Part of Reasoning modules, it aims to find a plan to satisfy a goal. Plans (list of actions) are stored on the blackboard.

The *Scheduler* Module chooses a plan, takes all actions of this plan and schedules them at the required place in the *BB:agenda of actions*. If the plan cannot be scheduled, a new Event(New_Goal) can propose to replan.

In the case of a normal moving, nominal path is produced by a *ACTION:Plan_Trajectory* Module. This module computes a nominal trajectory for a nominal speed-time profile from the current posture, the final goal and the static model of the environment stored in the *BB:environment* part of the blackboard. The *ACTION:Execute_Trajectory* Module generates commands to the physical device each sample period of time Δt. According to the potential field method [6], a command consists in the nominal path parameters modified by potential field parameters. These two actions modules are executed by the blackboard control unit by reaction to Event(Activate_Action_...) from the *Executor* Module. Once the execution phase is started, the agent is in a perception/action mode, since a plan is being performed. The perception modules react to dynamic modification of the environment.

In the situation represented by figure 5, the agent follows a vehicle with a lower-speed. The vehicle is detected by the perception modules whose results trigger a *AGENT CONTROL:Evaluator* Module. The trigger Same_way_vehicle_present is validated, and implies to Acquire_vehicle_parameters. This tasks is done by the *ACTION:Agent_Interaction* Module to ask the other agent its speed, direction, etc. The *COMMUNICATION* Modules manage the mail-

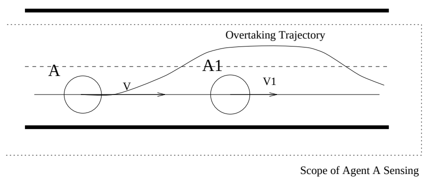

Figure 5: Situation of Perception-Communication/ Decision/ Action : Lower-Speed Vehicle on the Road

box and emit Event(New_Messages). A *AGENT CONTROL:Evaluator* evaluates consequences of the message reception (eventually according to a protocol). It modify the *BB:Others:vehicle_parameters* with exchanged values, update the *BB:New_Adaptations* fields and emit Event(New_Adaptations) resulting of the evaluation of *BB:communication_triggers* and *BB:communication_guards.* The *DECISION:Scheduler* Module is triggered by a necessary adaptation, it schedules the action stored in the blackboard with an immediate date of activation, so that the executor emit immediately the *Event(Activate_Action...).* The action can be *ACTION:Slow_Down* or *ACTION:Overtaking* which will modify the potential parameters of the commands stored in the *BB:Goal_Tree.* This behavior follows the Perception/Decision/Action process without replanning.

This situation illustrates, so, the functioning of the parallel blackboard architecture of agent presented in this paper. Both perception of static environment and interaction with other agents are taken into account. All stages of the decision process are proposed, allowing a really efficient achievement of the integration of reactive and deliberative capabilities in a real time context.

6 Conclusion and Perspectives

Viewing Parallel and Distributed Blackboards as Support Architectures for Multi-Agents Systems Blackboard Approach brings software flexibility and modularity the implementation of Action and Perception/Communication tasks. Furthermore, the independence between modules allows the coupling of both simulated and physical control modes of activity.
The study of a parallel blackboard model specially adapted to reactivity brings a powerful support for problems involving both centralized representation of data (as cognitive agents) and reaction to unpredicted events (reactive aspects).
A multi-agent system will be in fact implanted as a distributed hybrid blackboard system, using a set of parallel blackboard systems in a distribution close to the DVMT structure [5].
Integration in the MAGMA platform A generic tool has been developed in C++ using UNIX communication libraries [8]. It offers a graphical interface to build parallel blackboard systems and so reactive/cognitive agents. This software is now under development using DPSK+P libraries of active objects. The model of agent will be so integrated with its tool into the MAGMA platform of development and simulation of multi-agent systems, currently studied in our group.

7 Acknowledgements

The author would like to thank Stephan Bussmann for his study on the integration of cognitive and reactive aspects in a agent structure, which is partly at the origin of this agent model.

References

[1] O. Boissier and Demazeau Y. A multi-agent control architecture for studying the control of an integrated vision system. In *IEEE Int Conf. on Multi-Sensor Fusion and Integration for Intelligent Systems - MFI'94*, Las Vegas, USA, october 1994.

[2] S. Bussmann and Y. Demazeau. An agent model combining reactive and cognitive capabilities. In *To appear in Proccedings of IEEE International Conference on Intelligent Robots and Systems - IROS'94*, Munchen, September 1994.

[3] D.D. Corkill. Advanced Architectures: Concurrency and Parallelism. In V. Jagannathan, R. Dodhiawala, and L.S. Baum, editors, *Blackboard Architectures and Applications*, chapter II, pages 77–83. Academic Press, 1989.

[4] Y. Demazeau, O. Boissier, and J.L. Koning. Interaction protocols in distributed artificial intelligence and their use to control robot vision systems. In *IEEE Int. Conf. on Systems, Man and Cybernetics*, San Antonio,Texas, USA, october 1994.

[5] E.H. Durfee. *Coordination of Distributed Problem Solvers.* Kluwer Int. Series in Engineering and Computer Science, 1988.

[6] T. Fraichard and Laugier C. On line reactive planning for a non holonomic mobile in a dynamic world. In *Proc. of IEEE International Conference on Robotics and Automation*, pages 432–437, Sacramento, Ca., april 1991.

[7] M. Occello. Distributed and Parallel Blackboards : Application to Dynamic Systems Control in robotics and Computer Musics. *PhD Thesis Report, University of Nice - Sophia Antipolis,* january 1993 (In French).

[8] M. Occello and M. C. Thomas. A Parallel Blackboard Generic Tool for Intelligent Robotics. In *Proc. of IEEE Tools for Artificial Intelligence*, Arlington, VA, USA, november 1992. IEEE Press.