Agent-oriented design of human–computer interface: application to supervision of an urban transport network

Houcine Ezzedine, Christophe Kolski, André Péninou

A version of this paper has been published in: *Engineering Appl. of Artificial Intelligence, 18, pp. 255-270.*
The final version is available at: http://www.sciencedirect.com/

AGENT ORIENTED DESIGN OF HUMAN-COMPUTER INTERFACE. APPLICATION TO SUPERVISION OF AN URBAN TRANSPORT NETWORK

Houcine Ezzedine (*), Christophe Kolski (*), André Péninou (**)

(*) LAMIH-UMR CNRS 8530
Université de Valenciennes et du Hainaut-Cambrésis, Le Mont Houy
F-59313 Valenciennes Cedex 9, France
Phone: +33 (0)3 27 51 14 69, Fax: +33 (0)3 27 51 13 16
{houcine.ezzedine, christophe.kolski} @univ-valenciennes.fr

(**) Laboratoire LGC - EA 2043 – Université Paul Sabatier
IUT Paul Sabatier – 115, route de Narbonne
F-31077 Toulouse Cedex 4, France
Phone: +33 (0)5.62.25.88.86, Fax: +33 (0)5.62.25.88.85
peninou@iut-blagnac.fr

ABSTRACT:
At the present time, the approaches found in the design of interactive systems use a modular structure, with the aim of achieving a better understanding of reactivity, flexibility, maintainability and reuse at the human-machine interface level. However, most architecture models are far more concerned with user-controlled applications and they do not consider the specificities of supervision applications in which the human operator acts as the controller and commander of an independent dynamic process. A multi-agent approach is a possible answer to this type of situation. The agent oriented model put forward in this article is the subject of an application intended in the long term to supervise the user information system of an urban transport network.

Keywords: multi-agent systems, architecture, Human-Machine Interface, interactive systems, specification, design, modeling, supervision, urban transport systems.

1. INTRODUCTION

The design of interactive systems, especially the design of their architecture, has been the subject of active research for the past twenty years (Foley and Van Dam, 1982; Pfaff, 1985; Coutaz, 1987, 1990; Dix *et al*., 1993; Gram and Cockton G., 1993; Helander *et al*., 1997; Calvary *et al*., 1997; Bass *et al*., 1991, 1998; Coutaz and Nigay, 2001). Currently, the research as regards architectures is moving towards new interactive applications using new information and communication technologies and sciences: CSCW and multi-user approaches (Hill *et al*., 1994; Tarpin-Bernard and David, 1997; Dewan, 1999; Anderson *et al*., 2000), intelligent systems (Kolski *et al*., 1993; Kolski and Le Strugeon, 1998; Beka Be Nguema *et al*. 2000; Höök, 2000), multi-modality and multi-platform approaches (Thévenin and Coutaz, 1999; Calvary *et al*., 2001 and 2003; Paterno and Santoro, 2003). It is also worth mentioning the new agent oriented approaches applied to information research on the Internet (Keeble and Macredi, 2000; Klusch, 2001; Grislin *et al*., 2001).
On the other hand, very little research concentrates on the problem of the architecture of human-machine interfaces used in supervision type industrial contexts. These interfaces are characterized by the highly dynamic and complex nature of the application: operators in control rooms are faced with thousands of variables and must perform highly cognitive tasks in order to maintain control of the system and intervene in the event of malfunction; descriptions of the problems of human-machine interaction in the control room can be found in (Rasmussen, 1986; Sheridan, 1988; Millot, 1988; Gilmore *et al*., 1989; Kolski, 1997; Moray, 1997) for example. We concentrate on these kind of systems in our research project, studying the potential contribution of multi-agent systems towards the design of such interfaces.
In this article, we will begin by presenting our approach for the agent oriented design of Human-Machine Interfaces (HMI). The various characteristics involved in such a design will be dealt with, especially as regards their contribution in the different phases in the HMI specification cycles. We will then describe the context for the application which we used as a basis for the experimentation of these principles. For this, the process model, which is useful for the creation of a useable prototype, will be

presented. The specification of the interface will then be presented. Each module specified and associated to an agent will be described, along with its role in the interaction with the user and the supervision of the process. The usability characteristics of each window will be given as part of these presentations. Finally, we will present elements concerning the interests and limits of our research. The article ends with a conclusion and some research perspectives.

2. TOWARDS A MULTI-AGENT APPROACH FOR HUMAN-MACHINE INTERFACES

Our aim is to suggest a multi-agent model for the human-machine interface in the context of complex industrial applications. In the following part, we describe the motivation behind using this multi-agent approach for interface design. The main existing propositions for interface modeling are then given and the principles we suggest are presented. They will be treated in the subsequent section dealing with the application created for the supervision of an urban transport network.

2.1. Notion of agent and multi-agent system

2.1.1. Notion of agent

There are many definitions of the agent concept (Bradshaw, 1997; Ferber, 1995; Gasser, 1989; Franklin *et al*., 1996; Logan, 1998; Wooldridge and Jennings, 1995). Up to now, none of them have truly met with unanimous approval, because of the great variety of entities referred to as “agents” which are studied by the researchers. The definitions diverge mainly as regards the type of envisaged applications and the research issues. An agent represents "an entity which acts”, that is to say an entity capable of modifying its environment. We can distinguish approaches oriented towards problem resolution, and software engineering type approaches which are concerned with new agent-based software architectures. We are interested in the latter and we will therefore retain the definition given by (Wooldridge and Jennings, 1995) who identified four key characteristics of an agent:

1. autonomy as regards the user and the environment,
2. sociability in order to communicate with other agents ,
3. reactivity as regards the changes in its environment,
4. proactive behavior as regards individual goals.

There are three types of agent capacities, matching the three general steps in the performance of a task:

1. a *perception* step: the agent perceives its environment and updates its internal representations of this environment and of the other agents,
2. a *cognition* step (decision): the agent determines which task is to be performed and decides when and how to do it,
3. an *action* step: the agent "acts"; it actually performs the actions which have been chosen.

These three steps form a repetitive cycle according to the evolution of the environment in which the agent is situated. They also exist at the macro level above the agent, that is to say at the multi-agent system level which will now be presented.

2.1.2. Notion of multi-agent system

A multi-agent system is made up of a group of entities which co-operate and are capable of communicating and coordinating their behavior in order to reach a common goal (Bond and Gasser, 1988). Generally, an artificial world peopled by interacting processes is called a multi-agent system.
Whereas the description of an agent gives a local vision of the problem, the multi-agent description provides an overall external view of the problem. In order to obtain a complete view, an intermediate description must be added, concerning the “social” aspect (relationships) which is internal to the system and deals with the interactions between agents (Mandiau *et al*., 1999).
The particularity of each multi-agent system lies in the criteria applied to the distribution of capacities and knowledge (or data) between agents. The distribution criteria can be multiple:

- natural distribution of the problem (spatial, functional, temporal, …),
- distribution of knowledge: use of the group synergy and reduction of the complexity of the problem resolution through the cooperation of several independent systems (example: multi-expert systems having expertises which can overlap each other),
- epistemological distribution in order to implement co-operation and coordination models (social models, for example),

- software-based criteria in order to build modules with a low degree of coupling and a high degree of consistency which are easier to develop and maintain,
- technological criteria, for example concerning the interconnection of processors.

(Wooldridge *et al*., 00) have argued the possible contributions of an agent approach towards system design, based on the argument that the agent is a new design paradigm, in particular for distributed and/or cooperative applications. Our approach begins with the same postulates and hypotheses and goes on to suggest a new approach for the design of interactive systems, in particular in relation to applications for the supervision of dynamic systems.

2.2. Interest of a multi-agent design of human-machine interfaces - approaches found in research work

One of the most important elements in HMI engineering is the choice and description of an HMI architecture (Coutaz, 1993; Coutaz and Nigay, 2001). An architecture supplies the designer with a generic structure from which he/she can build an interactive application. The generic structure usually provides a component-based view of the interactive system. Several architecture models have been developed by researchers over the past twenty years. Two main categories of architecture can be singled out: functional component architectures and structural component architectures.

The most well known models are to be found in the first category: the Language model (Foley and Van Dam, 1982), the Seeheim model (Pfaff, 1985), its development into the ARCH model (Bass *et al*., 1991). These models suggest splitting an interactive application into several functional components. For example, the Seeheim model is made up of three logical components (Fig. 1): (1) the *presentation* is the part which can be seen by the user and which manages the input and output; (2) the *dialogue controller* is responsible for the structure of the dialogue and dialogue control between the user and the interactive system; according to (Coutaz, 1990), syntactically correct phrases correspond to the requests and data a user wishes to transmit to the application, and in the other direction the dialogue controller receives abstract output phrases which are sent towards elements specialized in the *presentation*; (3) *the interface with the application* is responsible for the communication of data between the interface and the application whilst respecting the semantics of the data and the dialogue.

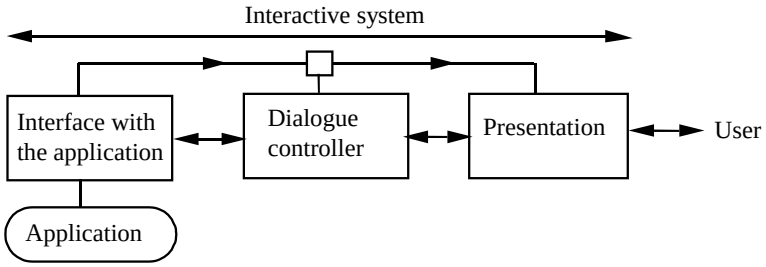

Figure 1. Seeheim model (Pfaff, 1985)

In the second category of architecture, we find agent models such as PAC and PAC-Amodeus (Coutaz, 1987; Nigay and Coutaz, 1991), object models (Palanque and Bastide, 1995; 1997) including the well-known MVC (Model, View, Controller) model (Goldberg, 1984) and its recent variations (such as MVC2). In such models, the system is divided up in a structural way according to the rules of composition and communication between structure elements. The agent model is therefore not completely new in HMI architecture. The PAC model (Coutaz, 1987) for example, makes it possible to introduce the notion of agent, and thus of state management, into the architecture of an interactive system. This model defines agents using three facets (Fig. 2) (Coutaz, 1987):

- P: the Presentation, which links the agent to the input/output devices,
- A: the Abstraction, which connects the agent to the functional core of the application,
- C: the Control, which is responsible within the agent for translation and communication functions between the two other facets, for a control between the two elements, as well as for a communication function with other PAC agents.

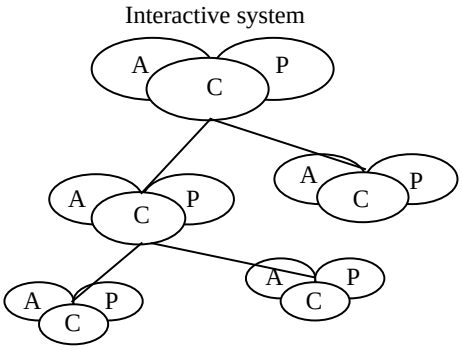

Figure 2. Example of a PAC model of an interactive system (Coutaz, 1987)

An architecture built according to the PAC agent model is a hierarchy of PAC agents, some of which control lower-level agents. The application architecture is built therefore from a single structure element, the PAC agent, and not from a functional division of the application.
The contribution of agent approach has been to encourage a component based design independent from any programming language and to make it possible to deal with complex dialogues, for example multi-wire dialogues. In particular, the breakdown of an interactive application into a hierarchy of agents according to simple and established composition rules is a powerful modeling tool. One possible drawback is the model of relations between PAC agents: the links between PAC agents are communication or composition links only. Although the composition links between PAC agents were studied closely to obtain rules for composition between agents in order to build coherent "clusters" of agents (Nigay and Coutaz 1991; Nigay 1994), the communication links between agent "clusters" still have to be established by the designer. Moreover, these two aggregation modes can prove to be insufficient when used in the context of supervision applications in which part of the interaction is not controlled by the user but by an independent software module.
It should be noted that even though these first models attempt to exploit the notion of agent for the design of interactive systems, they are still a long way from the definitions of agents and multi-agent systems given in section 2.1. Our aim is to take a step further towards these definitions.

2.3 Proposition for the modeling of Human-Machine Interfaces in process supervision

Agent approaches provide potentially rich possibilities for links between agents (Grislin-Le Strugeon and Péninou, 2001): coordination, cooperation, and communication links. We therefore suggest using this wealth of possibilities to develop interactive system architectures adapted to the context of supervision systems.
Given the types of architecture presented in the previous section, our approach is intended to be intermediate as its principles borrow from both types of model; it is both functional and structural. We suggest using a separation into three functional components which we will call: *Interface* (or *Presentation*, according to the terminology specific to the field, as this component is in direct contact with the user), *Dialogue Controller*, and *Interface with the Application* (connected to the application). These functional elements can be clearly identified and supply a decomposition into three sub-problems, each requiring a differentiated and relatively independent resolution (Pfaff, 1985).
However, we suggest that each of these three functional components can be broken down further in a structural approach in the form of agents (Grislin-Le Strugeon *et al*., 2001; Cartegnie *et al*., 2002). This approach encourages a true multi-agent structure for human-machine interfaces, that is to say a multi-agent organization which defines task distribution modes along with established and specified cooperation and communication protocols.
We will therefore break down an interactive application according to the three functional components recommended in the Seeheim model (Pfaff, 1985). Each component is then built like a multi-agent system, figure 3. The three multi-agent systems (Interface with the Application, Dialogue Controller, Presentation) are considered as working in parallel, at a theoretical point of view at least.

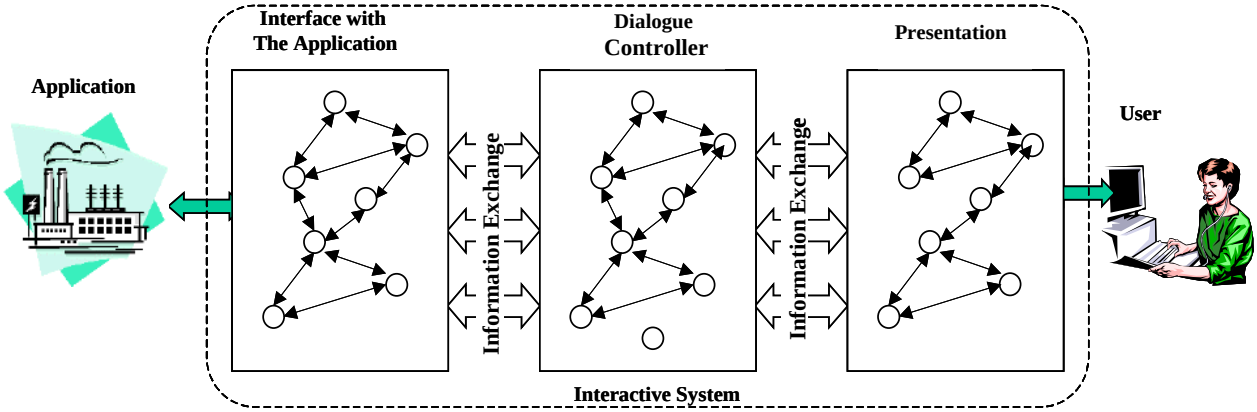

Figure 3. Suggested HMI model

Three types of agent are located in this architecture:

- *Application agents* in the *Interface with Application* component which manipulate the concepts of the application and cannot be accessed directly by the user. They guarantee the correct functioning of the application and the real time transmission of the information necessary for the other agents' work; the application agents therefore form a model of the real application;
- *Interactive agents* in the *Presentation* component; unlike the application agents, these agents are in direct relation with the user (who can "see" them). They coordinate between themselves in order to capture user commands and compose a presentation which will provide the user with an overall comprehension of the current situation of the application. Thus, a window can be considered as being an interactive agent in its own right; its specification describes its presentation and the services it provides. The interactive agents coordinate between themselves to guarantee the lexical and syntactical consistency of the interaction;
- *The control agents* in the *Dialogue Controller* component; these agents provide an intermediate representation between the application and the interface in order to give overall consistency to the dialogue. Their role, in particular, is to link the two other components together by distributing the user commands to the application agents concerned, and by distributing the application feedback towards the interactive agents concerned.

3. APPLICATION OF THE MODEL

3.1. Industrial Context

Our approach is currently being applied as part of a project which involves an industrial partner, the SEMURVAL company, which will run the future urban transport network in Valenciennes (Tram/Bus), along with several research laboratories (LAIL, I3D LAMIH and INRETS) (Ezzedine *et al.*, 01a). The project has the financial support of the Nord-Pas de Calais regional authority and the FEDER (Fonds Européen de Développement Régional - European Fund for Regional Development). Our research work is made up of the specification, design and assessment of a human-machine supervision interface to be used by the human controllers responsible for passenger information on the public transport system in Valenciennes (referred to as the Information Assistance System, or IAS).

As the tram lines are still under construction, at the moment the project team has to simulate the existence of the Tram/Bus network. The transport network will include information screens (or message makers) intended for the users. The screens are situated in the stations and the vehicles. The information displayed on the screens concerns the timetables of the vehicles and the connection information (connection timetables). It is calculated automatically by an exploitation assistance system (EAS) which is aware of the position and state of each vehicle in real time.

The project is to create a traffic regulation assistance system for the Valenciennes urban transport network. This regulation system is made up of three sub-systems: an Exploitation Assistance System (EAS), a Decision Assistance System (DAS) and an Information Assistance System (IAS), figure 4.

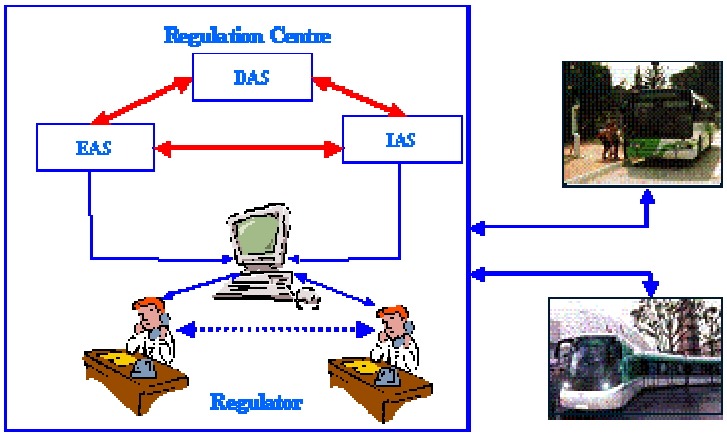

Figure 4. Description of the regulation assistance system

Each of the three sub-systems has its own role to perform:

- the EAS: centralizes the information concerning the exploitation (timetables, delays, vehicles ahead of schedule, messages, alerts, …) and makes it possible for this information to be managed;
- the DAS: develops, and assesses regulation strategies using information given by the EAS and suggests them to the regulator (Mesghouni *et al*., 98; Chihaib *et al*., 00). The regulator's task is therefore lightened, which should improve the quality of regulation. The DAS is not intended to replace the regulator, but it must provide him/her with assistance for decision-making (Lévine and Pomerol, 95).
- The IAS is the interactive system which forms the subject of our research work; it must present information to the regulators in the control room and allow them to transmit relevant information to the passengers.

As in most supervision problems, it is a question of permanently monitoring the state of a process and sending regulation commands for this same process. The process has its own individual behavior which is linked to some variables that the human operator can neither control nor modify and to some variables that the human operator can adjust. In our case, traffic conditions influence the times at which buses arrive at stations, but the regulator cannot influence the traffic conditions. On the other hand, the regulator can modify the state of the transport system and consequently regulate the information transmitted to the passengers. Our approach aims to meet the requirements of the regulator in terms of information needed so that he/she is able to perform his/her task of regulating the passenger information.

The regulator must be able to:

1. monitor the evolution of the passenger information which is automatically generated and transmitted by the IAS over the network in the stations and in the vehicles,
2. regulate this information if necessary, according to the company's objectives (reaction in the event of long delays, for example) or according to exceptional situations (temporary relocation of stops, maneuvers in the event of vehicle breakdown, etc.).

We have applied our agent oriented model of HMI to the specification and prototyping of the IAS.

3.2. Multi-agent specification of the IAS

3.2.1. Specification of the *Interface with the Application* component

The *Interface with the Application* component is a simulated multi-agent model of the real process, and at the same time it is directly linked with the real process. It is intended to manage the passenger information in buses and in stations, and to calculate the information to be displayed in normal conditions (delays, timetable modifications and scheduled runs, etc.). Given the natural geographical and structural distribution of the process, the vehicles and the stations are the relevant agents to the management of passenger information. They can communicate between themselves using the continuous availability of information concerning the line, and they can receive commands from the

regulator (via the *Dialogue Controller* component). Each agent has individual rules enabling it to make decisions as regards:

1. how to cooperate with the other agents,
2. which information should be displayed for the passengers ?

The different behavior of each agent takes into account the variety of possible situations such as: lines with heavy traffic and delays on certain sections only, stations with connections, lines with frequent or infrequent buses, lines with constant frequency or timetables. According to the traffic context, each agent has rules which enable it to act correctly in its environment: to display passenger information, to propagate relevant information. Finally, in the medium term, this approach makes it possible to deal easily with the evolution of the network over time: creation of temporary lines, modification of lines, etc.

The agent model selected is a reactive type in the sense given by (Ferber, 1995): according to the information received from the environment, the passenger information is never calculated in a global manner but only according to local rules.

The *Interface with the Application* component is a multi-agent system formally defined by a triplet (E, A, Op):

E: the environment made up of information coming from the EAS and the regulator (via the *Dialogue Controller* component),

A: the set of agents formed by the union of two sets:

Av: the set of vehicle agents which are autonomous and can communicate freely with other station agents and which receive information coming from the regulation room or the EAS,

As: the set of station agents which are autonomous and can communicate freely with the vehicle and station agents and which receive information coming from the regulation room or the EAS,

- Op: the set of the agents' possible operations.

A *vehicle* agent av belonging to Av is formally defined by a quintuplet (E, Ac, C, fa, fc), figure 5:

- E: set of the states of the environment which are perceived by the agent:
    - Eeas: States of information coming from the EAS,
    - Es: States of information coming from the station agents,
    - Edc: States of information coming from agents in the *Dialogue Controller* component,
- Ac: set of the agent's possible actions, sub-set of Op:
    - Acdo: set of display operations possible (display of information for the user),
    - Acco: set of communication operations with the station agents,
- C: knowledge of the agent:
    - Cacq: knowledge concerning the agent's acquaintances (station agents on the line),
    - Cchar: knowledge concerning the vehicle characteristics (journey to perform and itineraries, regulation actions in progress regarding the vehicle),
    - Cdisp: knowledge concerning the display implemented by the agent (global display parameters, local display rules),
- fa: action function: Edc x Eeas x Es x Cdisp → Acdo,
- fc: communication function: Edc x Eeas x Es x Cacq x Cchar → Acco.

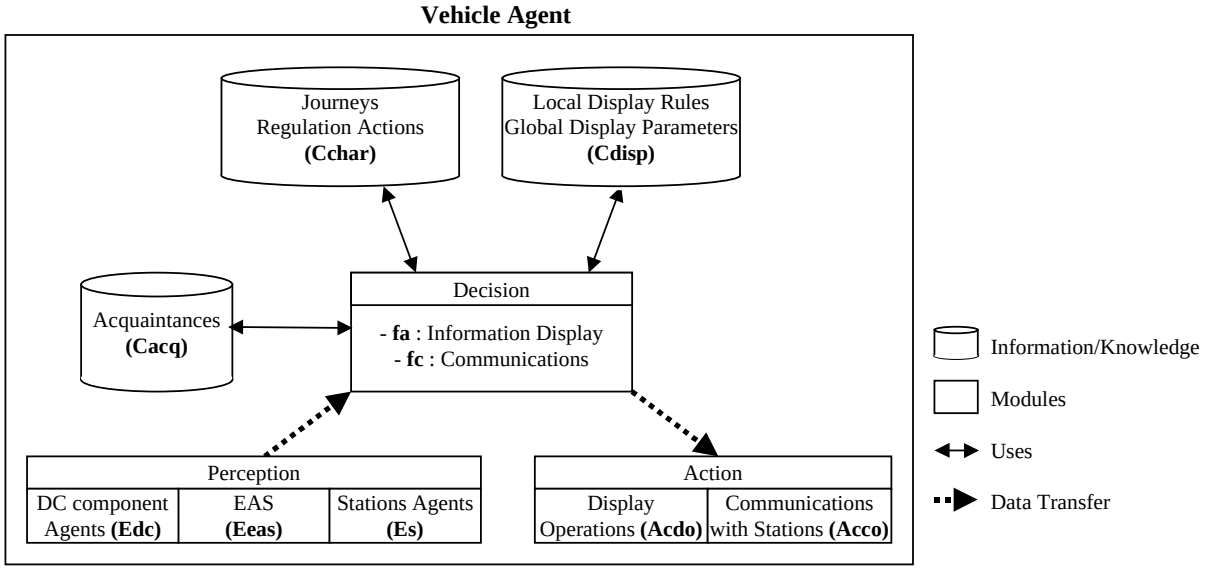

Figure 5. Definition of a vehicle agent.

A *station* agent as belonging to As is formally defined by a quintuplet (E, Ac, C, fa, fc), figure 6:

- E: set of the states of the environment which are perceived by the agent:
  - Eeas: States of information coming from the EAS,
  - Ev: States of information coming from the moving vehicle agents,
  - Es: States of information coming from station agents,
  - Edc: States of information coming from agents in the *Dialogue Controller* component,
- Ac: set of the agent's possible actions, sub-set of Op:
  - Acdo: set of possible display operations (display of information for the user),
  - Acco: set of communication operations with station and vehicle agents,
- C: knowledge of the agent:
  - Cacq: knowledge concerning the agent's acquaintances (preceding and subsequent station agents, expected bus agents),
  - Cchar: knowledge concerning the characteristics of the station (theoretical schedule, line route and position on the line),
  - Cdisp: knowledge concerning the display implemented by the agent (global display parameters, local display rules),
- fa: action function: Eeas x Edc x Ev x Cdisp → Acdo,
- fc: communication function: Eeas x Edc x Ev x Es x Cacq x Cchar → Acco.

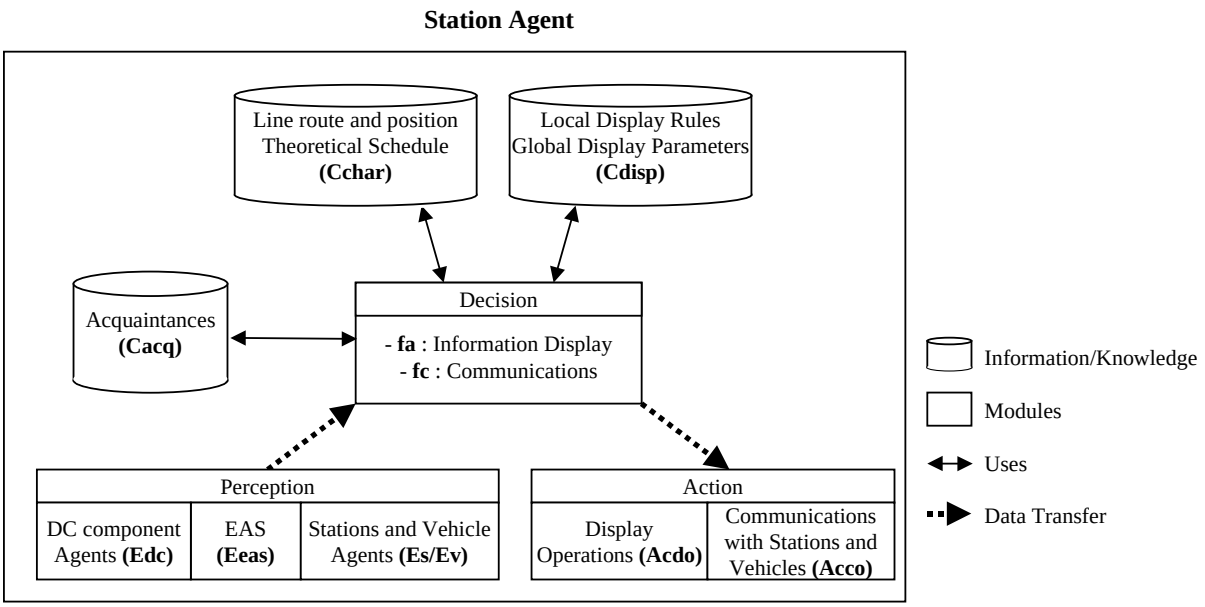

Figure 6. Definition of a Station agent.

For each vehicle or station agent, the action and communication functions implement specific rules for reaction linked to the spatial context, for example: the position of the station agent on the line, the distance as regards the vehicle, the number of stations in relation to the vehicle. Other contextual parameters can also be taken into consideration: rush hours and slack periods, traffic conditions, ...

3.2.2. Specification of the *Presentation* component

The *Presentation* component is made up of agents which manage the display of the process monitoring and the commands sent by the regulator. The agents group together various basic interactive elements logically in order to form relevant applicative views for the user. The applicative views linked to process control are distinguished from those linked to process command. The applicative views related to process control are: the synopsis of a line, the passenger information in stations, the passenger information on vehicles. The applicative views related to process command are: the passenger information modification functions. Each agent describes the state of a user interface object in relation to input/output functionalities: request for possible process modifications from the *Dialogue Controller*, transmission of the regulator's commands to the *Dialogue Controller*, etc. In particular, each agent manages a state of its current display (displayed or not). In the event of an alert, for example, an agent can react to prompt its display, even if this has not been explicitly requested by the user (one of the basic properties of an agent). It can also prompt for the display of other agents in order to give the user a coherent view of the context. This point is directly linked to supervision.
The agent model selected is a reactive type in the sense given by (Ferber, 1995). The *Presentation* component is a multi-agent system formally defined by a triplet (E, A, Op):
E: the environment made up of information coming from the *Dialogue Controller* component and from the user,
A: the set of agents formed by the union of two sets:
Ap: the set of applicative view presentation agents (one agent per view of the application),
Aco: the set of command agents (agents taking the regulator's commands into account to "regulate " the passenger information),
Op: the set of the agents' possible operations.
A *Presentation* agent ap belonging to Ap is formally defined by a quintuplet (E, Ac, C, fn, fa), figure 7:

- E: set of the states of the environment which are perceived by the agent:
  - Ei: States of information coming from agents in the *Dialogue Controller* component,
  - Ea: States of information concerning alerts coming from agents in the *Dialogue Controller* component,
  - Eu: States of information coming from the user commands,

- Ac: set of agent's possible actions at the interface level, sub-set of Op,
- C: agent's knowledge concerning the state of the interface,
- fn: display function in normal mode: Ei x Eu x C → Ac,
- fa: display function in the event of an alert: Ea x C → Ac.

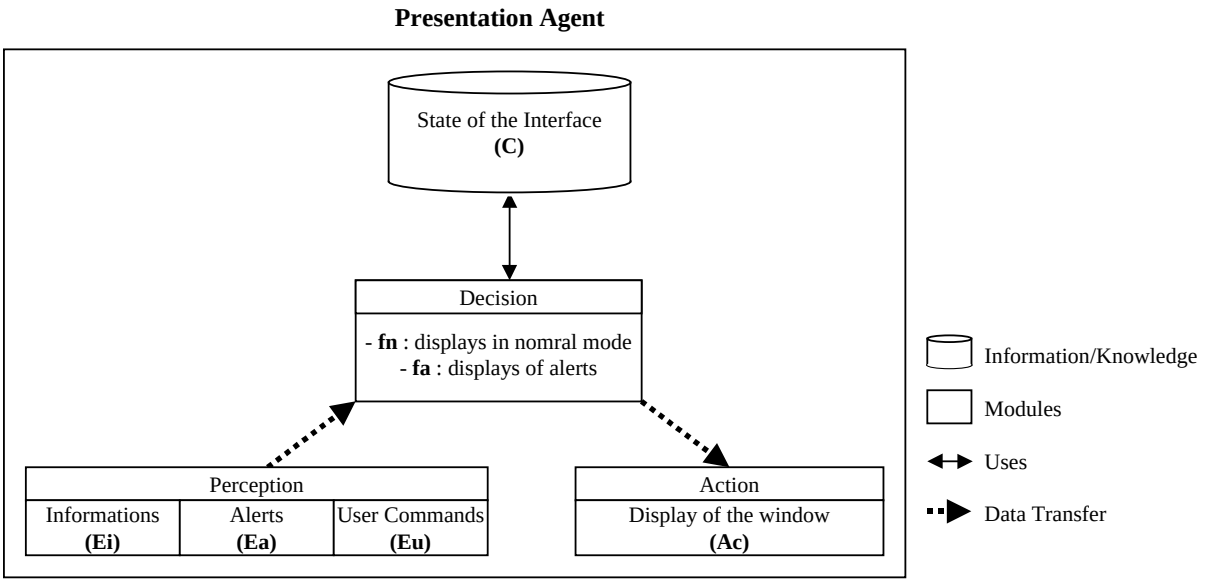

Figure 7. Definition of a *Presentation* agent.

A *command* agent aco belonging to Aco is formally defined by a quadruplet (E, Ac, C, fa), figure 8:

- E: set of the states of the environment which are perceived by the agent:
  - Ei: States of information coming from agents in the *Dialogue Controller* component,
  - Eu: States of information coming from the user commands,
- Ac: set of agent's possible agents to propagate the user's commands, sub-set of Op,
- C: knowledge of the agent:
  - Cacq: knowledge concerning the agent's acquaintances: the set of agents in the *Dialogue Controller* component,
  - Ccont: knowledge concerning the control of instruction consistency,
  - Cint: knowledge concerning the state of the interface,
- fp: instruction propagation function: Ei x Eu x Cint x Cacq x Ccont → Ac.

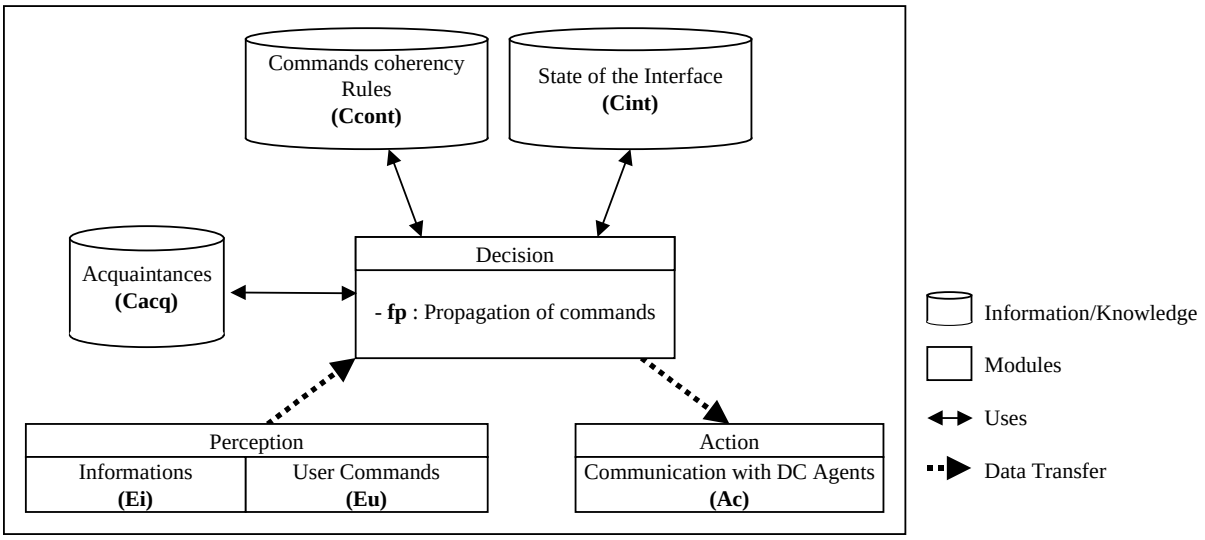

Figure 8. Definition of a *Command* agent.

3.2.3. Specification of the *Dialogue Controller* component

The *Dialogue Controller* component is made up of agents which represent the same process elements as for the *Interface with the Application* component (vehicle and station agents). Their role is no longer to simulate the real process, but to:

1. propagate the modifications observed at the application level on the interface agents,
2. detect and propagate the regulator's instruction actions towards the *Interface with the Application* agents.

This multi-agent system provides an abstraction when compared with the *Interface with the Application* agents. It must also manage specific constraints linked to interaction such as the suggestion of typical information solutions according to the situation. This point is not taken into account for the present time.

The agent model selected is a reactive type according to the sense given by (Ferber, 1995). The *Dialogue Controller* component is a multi-agent system formally defined by a triplet (E, A, Op):

E: the environment made up of information coming from the *Interface with the Application* and *Presentation* components,

A: the set of agents formed by the union of two sets:

- Av: the set of agents representing the vehicles,
- As: the set of agents representing the stations,

- Op: the set of agents' possible operations.

A *vehicle* agent av being to Av (and respectively a *station* agent as belonging to As) is formally defined by an n-uplet (E, Ac, C, fi, fa, fc), figure 9:

- E: set of the states of the environment which are perceived by the agent:
  - Ea: States of information coming from the *vehicle* agent (and respectively the *station* agent) in the *Interface with the Application* component to which the agent is linked,
  - Ec: States of information coming from the command agents in the *Presentation* component,
- Ac: set of agent's possible actions, sub-set of Op:
  - Aci: set of operations for the propagation of the state of the *Interface with the Application* agent towards the presentation agents in the *Presentation* component,
  - Aca: set of operations for the propagation of states of alert towards the presentation agents in the *Presentation* component,
  - Acc: set of operations for the propagation of commands from the command agents towards the *Interface with the Application* agent,
- C: knowledge of the agent:
  - Cacq: knowledge concerning the agent's acquaintances: vehicle agent (or station agent respectively) of the *Interface with the Application* component to which the agent is linked, presentation agents in the *Presentation* component giving a view of the vehicle agent (or station agent respectively), command agents from the *Presentation* component,
  - Cprop: knowledge concerning the rules of information propagation,
  - Cal: knowledge concerning the rules for the detection and propagation of alerts intended for the presentation agents in the *Presentation* component,
- fi: function for information propagation towards presentation agents in the *Presentation* component: Ea x Cacq x Cprop → Aci,
- fa: function for the propagation of alerts towards presentation agents in the *Presentation* component: Ea x Cacq x Cal → Aca,
- fc: function for the propagation of user's commands from the command agents towards the *Interface with the Application* agent: Ec x Cacq, x Cprop → Acc.

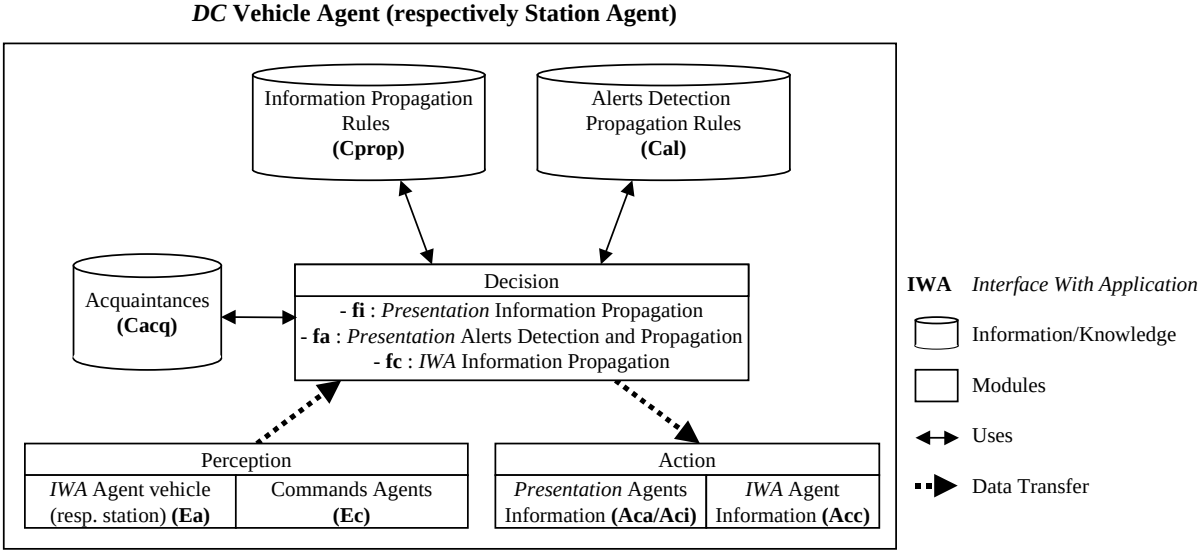

Figure 9. Definition of the vehicle and station agents in the *Dialogue Controller* component.

3.3. Development of the Information Assistance System (IAS)

3.3.1. *Modeling of the Information Assistance System (IAS)*

The prototype of the complete system is made up of three main parts:

1. an *Interface with the Application*: a simulator of the urban transport network which communicates information to the system like the EAS,
2. a *Dialogue Controller*: a multi-agent system which is made up of agents intended to provide resources and to monitor the graphic interface; these agents are called intermediate agents,
3. a *Presentation*: a graphic interface intended to present the information and to capture the regulator's requests.

The aim of the Information Assistance System (IAS) is to allow the regulator to visualize, edit, create and transmit information intended for the passengers situated in the stations and/or in the vehicles. Data is exchanged with the Exploitation Assistance System (EAS) which has real time knowledge of the position and the state of all the vehicles, and with the Decision Assistance System (DAS) which helps the network regulators with the making of certain decisions (not considered in this research project). The EAS is totally independent from the IAS, which must be capable of understanding the information and generating other information in the same format. For this, the multi-agent system (*Dialogue Controller*) acts mainly as an information filter and buffer, as well as a translator and it also checks the consistency of requests.

Our prototype, used for the development of the interface, takes the form of a usable interface, associated to a simulator of the process. The entire application is developed in Java, object oriented language, while the graphic interfaces are defined using dialogue components from the Swing library of the Java J2SE platform (Java 2 Standard Edition). The layout of the static elements of the views was developed using the graphic interface editor from the development environment Forte for Java from Sun Microsystems.

Modeling of the *Interface with the Application* in the form of a simulator

The simulator groups together the information necessary for the generation of messages emitted by the EAS. It is also capable of modifying and adapting the information according to the requests or updates required by the regulator (translation and verification functions). This data represents the stations, the lines, the vehicles, the operating charts based on the schedules (specifying the itinerary of each vehicle: departure time, arrival time of each vehicle, passing points, simple stations and connecting stations), as well as the alert messages for regulation or information messages for the passengers. A module generates the events, according to the timescale in order to modify the simulator data. The modifications can be, for example, the addition or modification of a delay for a vehicle, an itinerary modification, an alert issued at a station, etc..

Modeling of the *Dialogue Controller*

The *Dialogue Controller* multi-agent system supplies all the resources necessary to the interface, i.e. data and services. With regard to the specification presented earlier, some simplifications were made (regrouping of agents), without modifying the fundamental principles of the architecture suggested. This shows the flexibility of the specification developed. This architecture enables the IAS to be totally independent from the EAS. It keeps a local representation of the state of the EAS using messages sent by the simulator. In this way, the system only retains information which is useful for the regulator. The intermediate agent (*Dialogue Controller*) determines whether it is necessary to communicate the modification of the state of the transport system it has received (state of the infrastructure and state of the vehicles) to the graphic interface (*Presentation*). If need be, it communicates an order to the graphic interface to update using this data.

The simulator we developed enables us to display graphic entities on the interface which represent the stations and/or the vehicles. The changes of state of these entities is highly static because it depends upon the operating charts which are modified by the EAS alone.

Modeling of the Graphic interface (*Presentation*)

The graphic interface depends upon the *Dialogue Controller* in which it finds its resources. Any modification of the interface reflects a modification of the corresponding intermediate agent (*Dialogue Controller*). The parameters or actions registered by the interface are transmitted to the intermediate agents (*Dialogue Controller*) which will indicate to the interface that it can update using the data in question.

In the following paragraphs, we will present the various graphic views created according to a specification developed by the designer and validated by the users. The graphic views represent the various agents already specified concerning: the line synopses (itineraries), the vehicles, the stations and the messages (message box).

3.3.2. *View of the line*

To help the regulator to perform his/her supervision task, it is both useful and necessary to present a global view of the transport process on his/her workstation , along with an overall view of its state and dynamics. In our case, the process is a transport system represented by a view made up of graphic elements such as stations, sections, vehicles, etc. (figure 10). The representation of the elements is variable, according to the parameters of the entity they represent: each class of problem is represented by a color, each type of vehicle by a shape, etc.

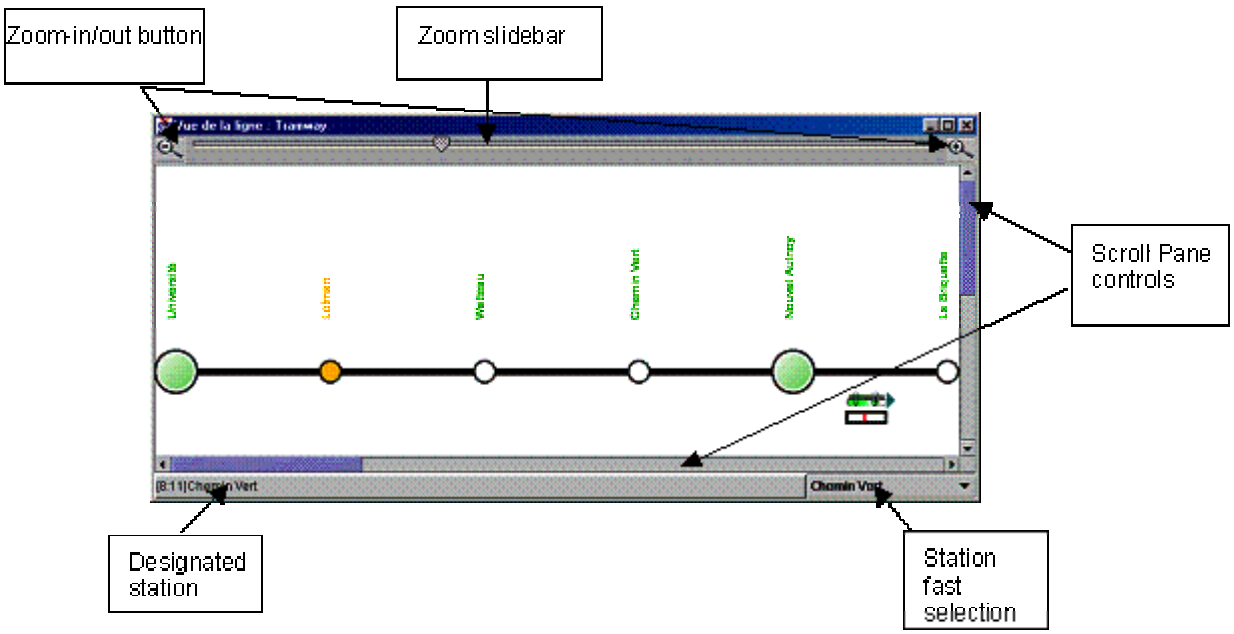

Figure 10. View of a line

The various graphic elements (vehicle, station, etc.) represent interactive agents which either react to changes of context in the transport process or reply to requests from the regulator (click on the graphic element).

3.3.3. Detailed view of a station (station agent)

The detailed view of a station can be accessed via its associated representation on the view of the line. The view is made up of several independent panels (messages and alerts, synthesis of the state and information from the operating chart). Each of the panels is linked to the data concerning it by the corresponding intermediate agent. It shows the user the information contained in the route chart, in the form of a set of thumbnails which vary according to a direction which is selected in a retractable scrolling menu (figure 11a). The content of each window representing a station is updated by the intermediate agent whenever there is a change in the context of the transport system (vehicle delayed or ahead of schedule on a line, maintenance works, new message from the controller or the EAS, etc.). In order for the regulator to fulfill his/her task of monitoring and regulating the transport flow correctly, he/she sometimes needs to communicate with a station (and/or a bus). If he/she wishes to send a message to a bus or a station, he/she must select the line in question and click on the graphic element representing the station (or bus) agent which is to receive the message. A window corresponding to the agent concerned opens and the regulator can write the message and send it.

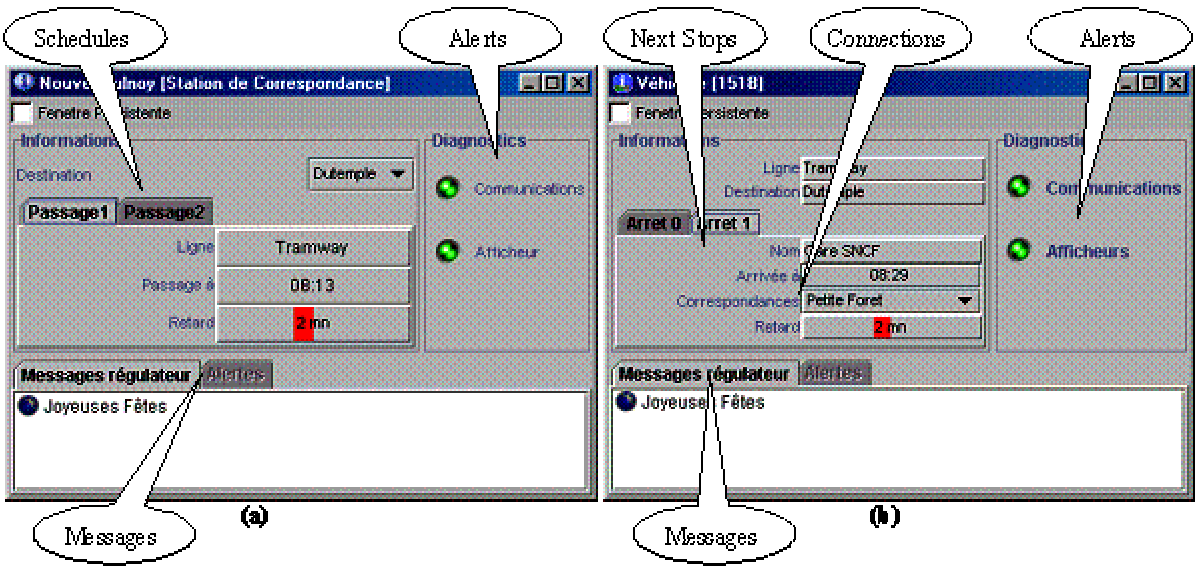

Figure 11. (a) Station information window, (b) Bus information window

3.3.4. Detailed view of a vehicle

The detailed view of a vehicle (figure 11b) includes the same elements as the detailed view of a station, except the representation of the information associated to its operating chart. A retractable scrolling menu contains the connections available upon arrival at the station at the actual time. These connections are supplied by the agent via a service which works with the station operating charts.

3.3.5. The message box

The message box is the component which allows the user to obtain a synthetic view of all the messages currently displayed and intended for passengers. The message box is made up of two types of panels:

1. A panel which manages a list of all the messages (figure 12a). This list is made up using the agent's data on the set of messages.
2. A second panel, activated by the previous one, is intended for the edition or creation of messages (figure 12b). Once it has been validated, it is responsible for communicating the new message to be diffused to the intermediate agent. The intermediate agent is then responsible for circulating the

message (updating the local state of each station/vehicle) and for communicating these changes to the interfaces. There are various types of message, according to the addressee :

- Mono-addressee, if it is sent to one single bus or station,
- Multi-addressee, if the message is sent to a group of buses or stations,
- Global diffusion, if it is intended for all of the buses or all of the stations, or both at the same time.

This characteristic gives rise to a specific representation in the interface (with an icon). The "message" agent answers a request from the regulator by displaying the graphic element shown above. The regulator selects either a bus or a station (or a group of buses or stations), writes his/her message and sends it. The names of the stations, the numbers of the buses, along with several typical (repetitive) messages are already recorded in order to save time for the regulator in performing his/her task of writing and sending messages.

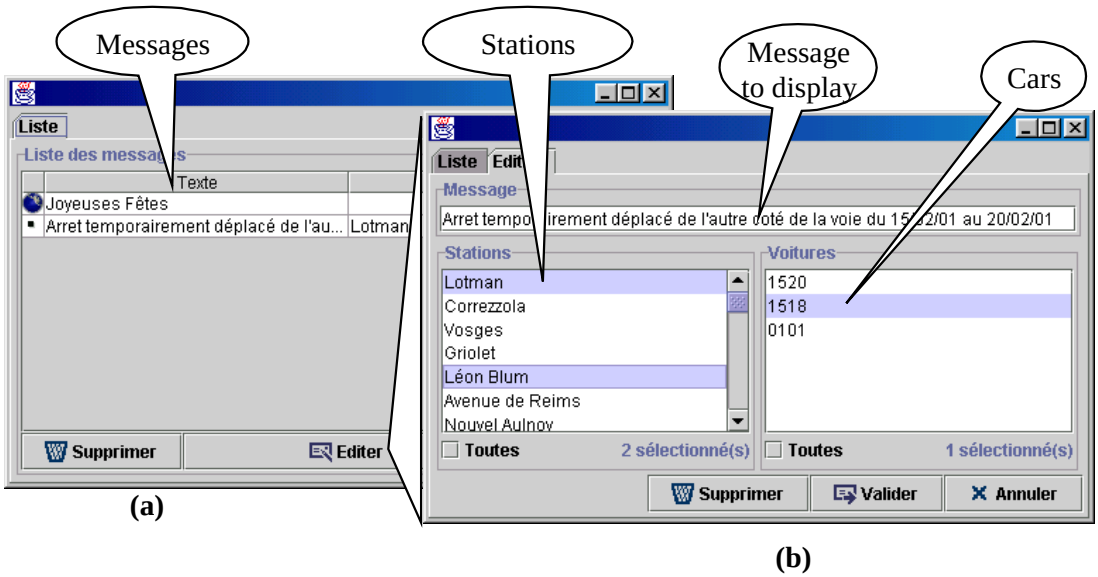

Figure 12. Message box: (a) list of messages, (b) message writing panel

4. RESULTS AND DISCUSSION

4.1. Results

In the context of the research project, this work was assessed by the company which manages the transport network (Ezzedine *et al.*, 01b). The views on the user interface were studied with the users (monitoring and instruction display). In particular, the layouts, the colors and the pictograms used in the views were taken from the work environment currently used by the regulators and were validated. As the prototype can be used, an assessment procedure is currently underway in order to validate firstly the *usability* which will give results concerning the quality of the Human-Machine interaction in terms of ease of learning and use, as well as the quality of the documentation, and secondly the degree of *usefulness* which will determine whether the HMI enables the user to attain the relevant work goals. It corresponds to the functional capacities, to the performance and to the quality of the technical assistance provided by the system for the user (Shneiderman, 1992; Nielsen, 1993; Ezzedine and Abed, 1997).

The application is faced with a dual-level problem: it must register the constant evolution of the system at the interface level (with a sampling period of 30 seconds), and it must take into account the user commands and the elements concerned in the process (modification of the displays in the stations and the vehicles). This point makes it necessary to have an exact model of the process, making it possible to present its evolution and reflect the commands. Thus, going beyond the multi-agent structure of the system, the approach suggested makes it possible to develop the *Interface with the Application* component as a multi-agent simulation which directly and faithfully reifies the actual process.

As regards the prototype developed, the use of intermediate agents (*Dialogue Controller*) enabled an implementation of the interface which was totally independent from the EAS (and the simulator). The use of these agents also made it possible to lighten the EAS load by filtering and keeping a copy of its

data. The modular breakdown of the interface brought about possibilities of interface distribution. This distribution made it possible to create the final interface in the form of screens without having to worry about interactions between the windows. In the same way, it enables us to envisage a spatial distribution of the elements over several administration posts or redundancy posts.

4.2. Discussion on agent oriented interface modeling

The first part of the article gave a brief state of the art of the architectures possible for human-machine interfaces. In particular, our suggestion of architecture is based on both functional and structural components. For the functional side, for the moment we have used the Seeheim architecture (Pfaff, 1985). For the structural side, we took a multi-agent approach which has already been used in other research work.

Compared with multi-agent approaches for interfaces such as PAC (Coutaz, 1987; Nigay and Coutaz, 1991), our approach encourages a true multi-agent structuring of the human-machine interfaces, that is to say a structure built around a multi-agent organization defining task distribution modes as well as established and specified communication protocols, thus going beyond mere composition or communication type links between the agents. The information perceived by the agents coming from the other agents (environment) and from the acquaintances (knowledge) introduced in the specification of the agents explains the communication links (orientation) and the communication content. The final objective is to develop systems capable of adapting themselves in a dynamic way to the complexity of the data to be handled, as well as to the complexity of the user [1].

As the applications targeted are of process supervision type, i.e. process control/command, this approach makes it possible to take the two aspects of the problem into consideration separately. During the supervision of a process, the operator has to perform four types of tasks requiring particular functions (Millot, 1988). We have studied two of them in particular[2]: (1) tasks of regulation/monitoring at a permanent operating rate, (2) tasks of detecting defects, diagnosis and compensation. The functions associated to these tasks are relatively independent from a functional point of view. The regulation/monitoring at a permanent operating rate functions depend on information proposed by the system to the operator and concern the state of the process. The functions for the defect detection, diagnosis and compensation depend on an instruction/command from the operator submitted to the system in order to regulate the process; the return of the process to a stable state being observed via the aforementioned functions.

This relative independence of functions makes it possible to apply a principle of functional separation to each sub-system supporting these two types of functions in each architecture component. The separate analysis of the two sub-systems of each component can be achieved by two modeling directions: each component is divided into two sub multi-agent system, or else each component is structured as a single multi-agent system, but each agent has competences in the two fields (control/command). In our study, given the constraints met with and the functions developed, we have chosen the first solution for the *Presentation* component and the second for the two other components (*Interface with the Application* and *Dialogue Controller*).

The multi-agent model chosen for the *Interface with the Application* makes it possible, in the context of supervision, to represent the process at the level of the elements relevant to the user. The dialogue can then be built easily around the basis of these relevant elements. In the application presented, for example, the *Interface with the Application* component does not take account of all the exploitation details found in the EAS such as the information necessary for vehicle maintenance. The automatic traveler information calculation was done in the *Interface with the Application*. However, in a HMI structuring perspective, the *Interface with the Application* component corresponds more to the *Interface with the Application* part of the ARCH model (Bass *et al*., 1991). Thus, we should separate the functions of the agents in the component in question into two separate multi-agent systems: passenger information calculation functions (*Application* component), and state of passenger information (*Interface with the Application* component). The breakdown into vehicle agents and station agents would remain valid in the two multi-agent systems thus created.

---

[1] For example (Péninou et al., 1999) have studied the possible applications of multi-agent systems for adaptative interfaces (in the sense given by (Edmonds, 1981; Schneider-Hufschmidt et al., 1993)).

[2] The two other types of function not studied here are: transition tasks during changes in process operating speed and specific procedure tasks.

In the application developed, the emphasis was placed in particular on the process control. The model proposed is therefore particularly suitable for this context. Because of this, the dialogue control operated by the system remains relatively easy to implement. The study of the command part of the process was more limited (message box). In order to take into account more complex cases, it would be possible to envisage to use existing dialogues modeling techniques in order to complete the model obtained up to now. A PAC type multi-agent approach could be suitable for this. Nevertheless, it would appear wise to keep the model selected for the control part and to complete it for the command part. We could, for example, envisage complex dialogues, modeled in the form of agents similar to PAC agents but linked to the dialogue controller agents as we have defined them. Finally, it would appear to be possible to envisage the integration of our approach into an ARCH type architecture model in order to meet the criteria of software quality: modifiability, reusability, portability.
The modeling process of the three components in three multi-agent systems was done in a natural, structural and modular way, but was partly dependent upon the problem, by the design team grouping together engineers and experts in the fields of HMI and multi-agent systems. A long term aim is to be able to formulate rules making it possible to propose a more standardized model, or else multi-agent design patterns, meeting the constraints and specificities of supervision. The links between agents in our system consist of communication links to exchange information, the decisions being taken locally as a reaction to this information which forms the environment. They are explained through the specification of the environment of an agent and its acquaintances. The detailed protocols for communication and data exchange in the HMI framework still have to be studied.

5. CONCLUSION

The design of interactive system architectures has been the subject of active research over the past twenty years. However, few research projects consider the problem of the architecture of human-machine interface used in supervision-type industrial contexts. These interfaces are characterized by their high application dynamics and their complexity.
In this article, we have put forward an agent-oriented design approach for human-machine interfaces to be used in process supervision applications. The architecture suggested for the interfaces is both functional and structural. The interface is split into three functional components (*Interface with the Application, Dialogue Controller, Presentation*). Each component is broken down into a multi-agent organization defining the task distribution modes and the protocols for cooperation and communication. Our agent-oriented design approach for the development of human-machine interfaces intended for process control/command was applied to the context of land traffic (bus/tram) and provided considerable help in the field of passenger information. A prototype was designed and developed according to the multi-agent specification put forward.
The perspectives of this research work on human-machine interfaces for supervision involve several points. Firstly, a long-term objective is to be able to formulate rules making it possible to suggest a more standardized model, or else multi-agent design patterns, whilst meeting the constraints and specificities of the supervision. Secondly, one of the perspectives of this work is to put forward a design method for more interactive applications in the process control/command field. Finally, our current research concerns the study of the assessment methods for such agent-based human-machine interfaces.

6. ACKNOWLEDGEMENTS

The work reported in this article has obtained the support of the Nord-Pas de Calais regional authority (Projects GRRT SART, NIPO) and the FEDER (Fonds Européen de Développement Régional, European Fund for Regional Development).

7. REFERENCES

Anderson, G., Graham, N., Wright, T., 2000. Dragonfly: Linking Conceptual and Implementation Architectures of Multiuser Interactive Systems. *Proceedings ICSE 2000*, ACM Press, pp. 252-26.

Bass, L., Little, R., Pellegrino, R., Peed, S., Seacord, S., Sheappard, S., Szesur, M., 1991. The Arch model: Seeheim revisited. *Proceedings of User Interface Developers Workshop*, Seeheim, April.

Bass, L., Clements, P., Kazman, R., 1998. *Software Architecture in Practice*. Addison Wesley Publications.

Beka Be Nguema, M., Kolski, C., Malvache, N., Waroux, D., 2000. Design of human error tolerant interface using fuzzy logic. *Engineering Applications of Artificial Intelligence*, 3, 279-292.

Bond, A., Gasser, L., 1988. *Readings in distributed artificial intelligence*. Morgan Kaufman, San Mateo, CA.

Bradshaw, J.M., 1997. *Software agents*. Menlo Park, AAAI Press & MIT Press.

Calvary, G., Coutaz, J., Nigay, L., 1997. From Single User Architectural Design to PAC*: a Generic Software Architecture Model for CSCW. *Proceedings of CHI'97* (Atlanta, March 1997), ACM Press, p. 242-249.

Calvary, G., Coutaz, J., Thévenin, D., 2001. Supporting context changes for plastic user interfaces: a process and a mechanism. In Blanford A., Vanderdonckt J., Gray P. (Eds), *People and Computer XV – Interaction without Frontiers*, Springer, pp. 349-363.

Calvary, G. Coutaz, J. Thevenin, D. Limbourg, Q. Bouillon, L. Vanderdonckt, J., 2003. A unifying reference framework for multi-target user interfaces, *Interacting with Computers*, 15(3), pp. 289-308.

Cartegnie, F., Ezzedine, H., Kolski, C., 2002. Agent Oriented Specification of Interactive Systems: Basic Principles and Industrial Case Study. In Kolski C., Vanderdonckt J. (Eds.) (2002). *Computer-Aided Design of User Interfaces III*. Kluwer Academics Publishers, Dordrecht, pp. 381-389.

Chihaib, F., Hammadi, S., Borne, P., 2000. The Contribution of Linear Programming by Constraints Propagation In the Regulation of Traffic of Urban Transport Network. *Proceedings ITS 2000 7th World Congress On Intelligent Transport Systems*, Torino November 6-9.

Coutaz, J., 1987. PAC, an implementation model for dialog design. *Proceedings Interact'87 Conference*, H.J. Bullinger, B. Shackel (Eds.), North Holland, September, pp. 431-436.

Coutaz, J., 1990. *Interfaces homme-ordinateur*. Dunod, Paris.

Coutaz, J., 1993. Software architecture modeling for user interfaces. In J. Marciniak (Ed.), *Architectural design for user interfaces; the encyclopedia of software engineering*, Wiley & Sons Publications, pp. 38-49.

Coutaz, J., Nigay, L., 2001. Architecture logicielle conceptuelle des systèmes interactifs. In Kolski C. (Ed.), *Analyse et conception de l'IHM*, Hermes , Paris, pp. 207-246.

Dewan, P., 1999. Architectures for Collaborative Applications. In *Computer Supported Co-operative Work*, M. Beaudouin-Lafon (Ed.), Wiley & Sons Pub.

Dix, A., Finlay, J., Abowd, G., Beale, R., 1993. *Human-Computer Interaction.* Prentice Hall, New York.

Edmonds, E.A., 1981. Adaptative Man-Computer Interfaces. In: M.J. Coombs and J.L. Alty (eds.), *Computing skills and the user interface*, London, Academic press.

Ezzedine, H., Abed, M., Une méthode d'évaluation d'Interface Homme Machine de supervision d'un procédé industriel, *Revue JESA (RAIRO-APII-JESA)*, Vol. 31, n°7, pp. 1078-1110, 1997.

Ezzedine, H., Péninou, A., Maoudji, H., 2001a. Towards agent oriented specification of human-machine interface: application to transport systems. *Proceedings IFAC Human-Machine Systems HMS'2001*, Kassel, Germany, September.

Ezzedine, H., Cartegnie, F., Péninou, A., Maoudji, H., Kolski, C., 2001b. *Amélioration de la qualité des correspondances dans les réseaux de transports urbains*, Final report, projet coopératif GRRT, Valenciennes: LAMIH.

Ferber, J., 1995. *Multi-Agent Systems: Introduction to Distributed Artificial Intelligence*. Addison-Wesley.

Foley, J.D., Van Dam, A., 1982. *Fundamentals of interactive computer graphics*. Addison-Wesley, Reading, Mass.

Franklin, S., Graesser, A., 1986. It is an agent, or just a program ? A Taxonomy for autonomous agents. *Proceedings of the 3nd International Workshop on agent theories, Architectures, and languages*, Springer Verlag.

Gasser, L., Huhns, M.N., 1989. *Distributed artificial intelligence*. Vol. 2, Pitman, London.

Gilmore, W.E., Gertman, D.I., Blackman, H.S., 1989. *User-computer interface in process control, a Human Factors Engineering Handbook*. Academic Press.

Goldberg A., 1984. *Smalltalk 80 : The Interactive Programming Environment*, Addison-Wesley Publications.

Gram, C., Cockton, G. (Eds.), 1993. *Design Principles for Interactive Software*. Chapman & Hall.

Grislin-Le Strugeon, E., Péninou, A., 2001. Interaction with agents systems: problematics and classification. In M.J. Smith, G. Salvendy, D. Harris, R. Koubek (Ed.), *Usability evaluation and Interface design: Cognitive Engineering, Intelligent Agents and Virtual Reality, volume 1*. pp. 479-483. London: Lawrence Erlbaum Associate Publishers.

Grislin-Le Strugeon, E., Adam, E., Kolski, C., 2001. Agents intelligents en interaction homme-machine dans les systèmes d'information. In Kolski C. (Ed.), *Environnements évolués et évaluation de l'IHM. Interaction Homme Machine pour les SI 2*, pp. 207-248. Paris: Éditions Hermes.

Helander, M, Landauer, T.K., Prabhu, P. (Eds.), 1997. *Handbook of Human-Computer Interaction*. Elsevier Science B.V.

Hill, R., Brinck, T., Rohall, S., Patterson, J., Wilner, W., 1994. The Rendez-vous language and architecture for constructing multi-user applications. *ACM Transactions on Computer-Human Interaction*, 1(2), p. 81-125.

Höök, K., 2000. Steps to take before intelligent user interfaces become real. *Interacting with computers*, pp. 409-426, 12, 2000.

Keeble, R.D., Macredie, R.D., 2000. Assistant agents for the world wide web intelligent interface design challenges. *Interacting With Computers*, 12, pp. 357-381.

Klusch, M., 2001. Information agent technology for the internet: a survey. *Data and Knowledge Engineering*, 36 (3), pp. 337-372.

Kolski, C., 1997. *Interfaces homme-machine, application aux systèmes industriels complexes*. Editions Hermes, Paris.

Kolski, C., Grislin-Le Strugeon, E., Tendjaoui, M., 1993. Implementation of AI techniques for intelligent interface development. *Engineering Application of Artificial Intelligence*, 6, 295-305.

Kolski, C., Le Strugeon, E., 1998. A review of intelligent human-machine interfaces in the light of the ARCH model. *International Journal of Human-Computer Interaction*, 10 (3), pp. 193-231.

Lévine, P., Pomerol, J. C., 1995. The role of the decision maker in DSSs and representation levels. *Proceedings of 24th IEEE Annual Int. Conf. on System sciences*, Hawai.

Logan, B., 1998. Classifying Agent Systems. *Software Tools for Developing Agents: papers from the 1998 Workshop*, J. Baxter and B. Logan (Eds.), Technical Report WS-98-10, AAAI Press, pp 11-21.

Mandiau, R., Le Strugeon, E., Agimont,. G., 1999. Study of the influence of organizational structure of the efficiency of a multi-agent system. *Networking and information Systems Journal*, vol.2, n°2, pp 153-179.

Mesghouni, K., Hayat, S., Hammadi, S., Borne, P., 1998. Modeling of the traffic regulation for a subway line with double loops. *Proceedings IEEE Systems, Man and Cybernetics*, San Diego, California, USA, October 11-14.

Millot, P., 1988. *Supervision des procédés automatisés et ergonomie*. Hermes. Paris.

Moray, N., 1997. Human factors in process control. In G. Salvendy (ed.), *Handbook of Human Factors and Ergonomics*, John Wiley & Sons, pp. 1944-1971.

Nielsen, J., 1993. *Usability Engineering*, Academic Press, Boston.

Nigay, L., Coutaz, J., 1991. Building user interfaces: organizing software agents. In Commission of the European Communities, Directorate-General, Telecommunications, Information Industries and Innovation (Ed.), *Proceedings of the Annual ESPRIT Conference, ESPRIT'91* (pp. 707-719). Luxembourg: Commission of the European Communities.

Nigay, L., 1994. *Conception et modélisation logicielles des systèmes interactifs: application aux interfaces multimodales*. Thèse de l'université Grenoble I, Université Joseph Fourier.

Palanque, P., Bastide, R., 1995. Spécifications formelles pour l'ingénierie des interfaces homme-machine, *Technique et Science Informatiques*, 14(4), pp. 473-500, 1995.

Palanque, P., Bastide, R., 1997. Synergistic modelling of tasks, users and systems using formal specification techniques, *Interacting With Computers*, 9 (2).

Paternò, F. Santoro, C., 2003. A Unified Method for Designing Interactive Systems Adaptable to Mobile and Stationary Platforms, *Interacting with Computers*, 15(3), pp. 347-364, 2003.

Peninou, A., Grislin-Le Strugeon, E., Kolski, C., 1999. Multi-Agent Systems for Adaptative Multi-User Interactive System Design: somes Issues of Research. In H.J. Bullinger, J. Ziegler (Eds.). *Human-Computer Interaction: Ergonomics and user interfaces (HCI'99)*, Lawrence Erlbaum Associates, pp. 326-330.

Pfaff, G.E., 1985. *User Interface Management System*. Springer-Verlag.

Rasmussen, J., 1986. *Information processing and human-machine interaction, an approach to cognitive engineering*. Elsevier Science Publishing.

Schneider-Hufschmidt, M., Kühme, T., Malinkowski, U. (Eds.), 1993. *Adaptive User Interfaces*. North Holland.

Sheridan, T.B., 1988. Task allocation and supervisory control. In M. Helander (ed.), *Handbook of Human-Computer Interaction*, Elsevier Science Publishers B.V., North-Holland.

Shneiderman, B., 1992. *Designing the user interface: strategies for effective human-computer interaction*, Reading, MA: Addison-Wesley.

Tarpin-Bernard F., David B.T., 1997. AMF a new design pattern for complex interactive software? *Proceedings International HCI'97*, Elsevier, San Francisco, Vol. 21B, p. 351-354, August.

Thévenin, D., Coutaz, J., 1999. Plasticity of user interfaces: framework and research agenda. *Proceedings of Interact'99 seventh IFIP Conference on Human-Computer Interaction*, Edinburgh, Scotland.

Wooldridge, M., Jennings N.R., 1995. Intelligent Agents: Theory and Practice. *The Knowledge Engineering Review*, 10 (2), pp. 115-152.

Wooldridge, V., Jennings, N.R., Kinny, D., 2000. The Gaia Methodology for Agent-Oriented Analysis and Design. *Journal of Autonomous Agents and Multi-Agent Systems*, 3 (3) 285-312.