Informational middleware based on mutual awareness

Flavien Balbo[1,2], Mahdi Zargayouna[1,2], Julien Saunier[1]

[1]*Lamsade*
Université Paris-Dauphine
Place du Maréchal de Lattre de Tassigny,
Paris Cedex 16
{balbo, zargayou, saunier}@lamsade.dauphine.fr

[2] *Inrets / Gretia*
2, Avenue du Général Malleret-Joinville, F-94114
Arcueil
{balbo, zargayou}@inrets.fr

Abstract

This paper proposes a middleware based on the multi-agent paradigm. Our proposition enables agents to locate and to interact easily with heterogeneous services and information providers. Interaction within the middleware is based on the mutual awareness concept, which makes it possible to define the middleware as an information sharing place making easy a capability-based coordination. A real application from the transportation domain illustrates the use of this middleware.

1. Introduction

Designing distributed applications requires effective information processing and service management. In a Multi-Agent System (MAS), a process is considered efficient if the agents can locate and interact easily with the service or information providers. Our proposal based on mutual awareness interaction, takes advantage of two paradigms, middle-agent (preference/ability matching) and mobile agents (reducing communication cost) in order to solve the problems dealing with a dynamic informational context

The remainder of the paper is structured as follows: section 2 presents the mutual awareness paradigm; section 3 shows in detail the components of our architecture; section 4 draws general conclusions.

2. Why mutual awareness?

Dugdale [5] has proved that, in a dynamic informational context like regulation, a large part of the interactions derive from the concept of mutual awareness. In the context of regulation, information is accessible, and agents pay attention to that which is relevant to their current activity, depending on the agent's criteria. Therefore, assimilating preference and capability like with middle-agent [4] is not sufficient when the problem is not the location of the information but the content itself. The receiver can gain efficiency by choosing itself its sources and criteria. In addition, with a middle-agent, dynamic information rapidly increases the number of message exchanges in order to maintain a valid representation of the world for the agents [2].

Mobility could be a solution, as its main objectives are to limit message cost, to facilitate access to local information and agents, and to distribute computation cost. But against these advantages there are three downsides: (1) server/agent compatibility, (2) server security, (3) multiple sources processing. (1) and (2) are the reasons why this paradigm is limited to dedicated applications [11]. Finally, mobility does not solve the problem of multiple sources processing (3), as the cost of migration is high and the identification of relevant sources still needs some form of broker. Therefore, even if we keep the idea of reducing the costs by permitting the agent to use locally the server information, the problems induced by mobility in open systems have led us to look for another feature that could provide a solution to the initial question of global information sharing.

LIME [15] permits tuple-space sharing among distributed platforms, and Javaspaces[1] makes possible to share objects put and retrieval spaces between distributed agents. Even if these two technologies are close to our interaction needs, LIME doesn't ensure the consistency of the tuple-space. It is a sort of distributed blackboard, which means that the agents have to read the blackboard explicitly instead of receiving their messages, because the templates cannot be composed, it is not possible for agents to have a combined interest for several sources simultaneously. We also think dynamic messaging is better in a high interactional context, with another useful principle, which is mutual awareness.

Some work is related to the concept of mutual awareness, particularly “overhearing”. The purpose of overhearing is to allow agents to intercept messages they're not intended to receive.

Overhearing has been used in real dynamic environments to simulate indirect communication [12] as well as to improve the knowledge consistency of teams [7] or to monitor a MAS [6]. These three systems validate the usefulness of overhearing, but their implementation through broadcast functionally penalizes the system.

In order to limit the communication cost, channelled multicast [3] proposes a focused broadcast, by means of dedicated channels of communication where agents

[1] http://java.sun.com/developer/products/jini/

subscribe and/or emit. Nevertheless, two limits can be underlined: (1) the complexity of the system increases proportionally to the number of channels; (2) the sender still has to assume the emission of the messages to every agent. However, we observe that proposing a solution for overhearing has also led to an improvement for the sender: it can choose to emit a message through a channel, which is the visible expression of the interests of the agents, instead of using addresses or capability (via middle-agents).

Tummolini [14] defines the concept of Behavioral Implicit Communication (BIC), within the framework of cooperative systems for task realization, as the set of every interaction that can be observed in an implicit way. However, the properties that are required to fulfill BICs, make this framework hardly useable. It needs very cooperative agents, that is why it is hard to model and implement in an heterogeneous and open system. Platon's model of overhearing [9] is the most generic to our knowledge, as it considers overhearing independently of the domain of the application. Nevertheless, their proposition has not already been implemented.

3. The middleware modeling

Mutual awareness is based on the sharing of interactions. To be efficient, this principle implies that agents share a common communication media. In the reactive agent community, the environment is already used as a common media of interaction. The EASI model [1] enables cognitive agents to use the environment to exchange messages and, more precisely, it enables an agent to send messages to an other agent that is located by the environment and it enables agents to perceive every exchanged message. In our work, we consider that environment contains descriptions of messages and agents. The interactional problem is to make possible for agents to use these descriptions to locate messages according to the environment state, that implies the matching between those properties and the needs of the agents.

We therefore propose to represent every component of the environment (e.g. the external properties of the environment itself as well as the agents and messages) as entities. Every entity has its visible properties, accessible via the environment, and the ability to put filters in the environment. These filters are logical expressions on properties, and determine, when a message is added to the environment, whether the agent is interested in it, in which case it will receive it, or not. In our EASI model, we have added this notation to formalize the knowledge about the description of interaction components (messages and agents). Because it enables to represent the agents, it makes possible for agents to create their own interactional context as a set of filters. Each agent description is updated by the agent itself, modifying dynamically the value of its visible properties. A message put in the environment will be perceived by every agent that has a filter that is matched in the current informational context.

3.1. The application domain

A Traveler Information System (TIS) should provide two types of information: (1) information before the trip starts, that is to say the global offer over all transportation modes for a given request; (2) information during the user's trip, notifying him about events that could occur on his route. The transportation operators are proposing convenient responses to these needs including passive information - such as variable messages boards - and/or interactive information such as web servers or Personal Assistants. However, the information provided by the operators usually only concerns their own transportation mode(s). However, operator information makes the mutual management of different sources difficult, and requires the user to be adaptable.

The organization proposed by FIPA [8] is efficient for obtaining pre-trip information. But for a daily travel the problem is not to identify the information sources but to manage its update. In a real-time configuration, the request/response pattern becomes expensive in a very dynamic context like daily information in an urban area.

In order to test our proposal, we used existing information sources as a web service[2] (called *Planning*) that creates a trip as an answer to a request and an information system (called *Traffic*) [14] that gives information about the traffic. We also have defined a specific application (called *Alternative*) that gives the nearest alternative station rather than the one which has been received as a request parameter. To each traveler an agent (called *MPTA*) is associated.

3.2 The middleware architecture

The mutual awareness model proposed by EASI makes it possible to put together all the information. Each agent perceives only that information which, according to its filters, concerns its interests. This Agent Information Server (AIS) architecture does not duplicate information from the provider but organizes its use in a defined context. Our server is a common place where requesters and providers exchange information through a common environment.

There are two types of provider behavior according to the dynamicity of their information. For static information, a provider waits for users' requests (like *Planning*). In this case, the server is used by requesters in order to identify (with the use of the public properties) the provider and to interact with it in a normalized way. For dynamic information, a provider (like *Traffic*) puts updated information into the environment. In this case,

[2]chotto.free.fr/tatami/Metro

the server is used by requesters to identify which information has some interest for them from among all data available in the environment. New information is put in the environment once and is received by all interested users.

The multi-agent system upon which our architecture is based is made up of three types of agent. The first, *Interface* agent, is the link between an existing information server and our own. This agent is used by the others to interact with the external server in a normalized way (static information) and/or to gather relevant information for the MAS and to put it in the environment (dynamic information). Using an *Interface* agent within our multi-agent system makes it possible to keep a homogeneous system with heterogeneous components. Agents do not have to know the external server to interact with it; they only need to know which kind of service it provides. This implies that the server may be changed and that the technical means used to interact (http, ftp, SOAP, etc.) is hidden from the users' agents. In TIS *Planning* and *Traffic* are connected to the AIS via *Interface* agents (respectively *PlanningAgent* and *TrafficAgent*). The first interact via http and the second via ftp.

The second type of agent is the *Domain* agent. Contrary to *Interface* agents which are not interested in using information coming from the environment (they are only information providers), *Domain* agents may be requesters and/or providers of information. In TIS, *Alternative* and MPTA are *Domain* agent. The first interact via a SOAP connection and the second via http. These two agent categories are not located on the server. Using them in our proposal solves the problem of provider identification and standardizes interaction with heterogeneous information providers. Nevertheless the communication cost (via a SOAP connection) remains high because each interaction is carried out with a message exchange between distant agents.

To solve this problem our multi-agent system has a third category of agent, *Local* agent (LA), which is located on the server. Thus, a part of the processing may be done on the server, reducing the communication cost (as with mobile agents). Each distant agent (*Interface* and *Domain* agent) has a representative (*Local* agent) on the server. The role of this entity is to manage interaction for the distant agent, by creating/deleting filters according to the needs of the distant agent. For each perceived message, it decides what to do with it. Alternative is to deal with it or to forward it to the distant agent. In that way, the exchanged messages are limited to those that are essential for distant agents. The role of the *Local* agent implies a hybrid architecture, since this agent is the link between the informational environment (it can put and perceive messages) and the application environment (it can send and receive messages).

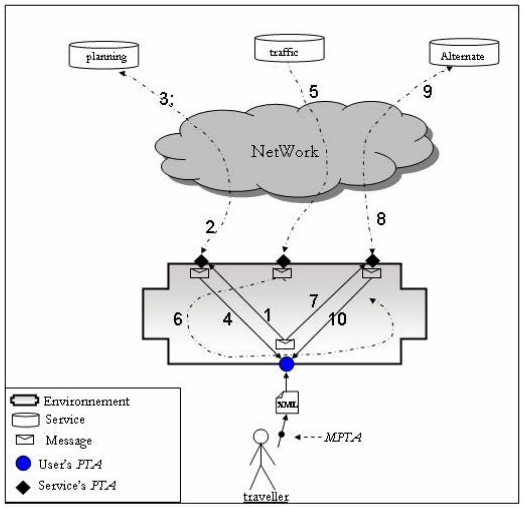

Figure 1: Example of messages routing

Chronologically (Figure 1), the traveler connects to the server (via his MPTA) and a LA (called PTA) representing him is created. After specifying his departure and destination points, he is asked to wait until his request is processed. His PTA creates a message with this information and deposits it in the environment. In this case, the message is "addressed" to the LA which has a planning capability (1). When the planning LA receives the message, it forwards it to the *PlanningAgent*. Then, the *PlanningAgent* requests the *Planning* for a plan, which it sends back in a message containing an xml trip plan (3); This is transformed – by the LA – into a message obeying the environment syntax, addressed to the user's PTA (4). Note here that the presence of the *Interface* agent between the LA agent and the planning system has multiple advantages. First, the fact that this agent is physically distant and interacts with the server by asynchronous SOAP messages means that the server doesn't have to worry about the synchronization of the http protocol. Second, the same *Interface* agent can have more than one LA agent representing it in different middleware servers, covering different transportation networks. This way, the user connects in exactly the same way to different networks, and the presence of different services is transparent to him.

When the user's PTA receives the xml plan it first sends it to its MPTA to inform the user, then it parses it and generates a filter for every plan segment (a plan segment is a part of a trip, provided by only one transport mode). This way, the PTA of a user restricts its reception conditions just to the information concerning its own trip.

If *TrafficAgent* sends a warning concerning a part of the user's trip (5), the message is intercepted by its PTA (6). The PTA puts in the environment a message "addressed" to the agent which has the alternative capabilities (7-8-9-10), Note that the alternative service has a filter enabling it to receive all the information relative to disturbances, so it doesn't send another station concerned by a traffic problem. Once the alternative station is received, the PTA sends an addressed message to the planning LA asking for a plan with the alternative station as a departure point (1-2). When receiving the new plan (3-4), if the gain with the alternative trip is higher than the current delay, the PTA proposes it to the human user and asks him if he wants to avoid the disrupted station. In this case, and if the new plan is validated by the user, the old filters are replaced by the new ones concerning the new plan; only the events concerning this new plan will henceforth be received.

Thus, using the EASI model in our application it was possible, for the interaction of an agent (representing a user), to be dynamically parameterized by its context, through the updating of its filters. The use of existing classical web services is also totally transparent to him; interaction with any kind of service is homogenized by the environment interaction protocol. Using AIS also enabled us to build a complex service based on different sources that had not been pre-defined to offer such a service.

5. Conclusion and perspectives.

The basic principles behind our informational middleware (AIS) described in this paper (capability-based coordination with middle-agents and reduction of communication costs with mobile agents) are principles generally acknowledged to be of interests in the multi-agent community. The operationalisation of these principles for a dynamic informational context imposes to take into account the update of information and/or of the agents' interest. Our proposition to use mutual awareness to create a communication space where representative of distant agents interact limits the communication cost.

A real application of a Transportation Information System illustrates our proposition. This specialized middleware integrates several information servers and enables normalized interaction with them.

We have several directions for future works. We plan to investigate the consequences of the admission or the exit of agents on services management and to propose a management process for taxonomy of available services.

6. References

[1] F. Balbo, "A model of environment, active support of the communication", in American Association of Artificial Intelligence Conference, AAAI-99, Workshop on Reasoning in Context for AI Applications, AAAI Press, 1999.

[2] F. Balbo, "A new interaction model for agent based simulation". In European Simulation Multiconference, 2004.

[3] P. Busetta, A. Donà and M. Nori, "Channeled Multicast for group communications", in "Proceedings of AAMAS", pp. 1280-1287, 2002.

[4] K. Decker, K. Sycara, Williamson M., "Middle-Agent for the Internet", in Fifteenth IJCAI, Morgane Kaufmann, 1997, pp 578-583.

[5] J. Dugdale, J. Pavard and B. Soubie, "A Pragmatic Development of a Computer Simulation of an emergency Call Center", in Frontiers in Artificial Intelligence and Applications, IOS Press' 2000.

[6] G. Kaminka, C. Pynadath and M. Tambe, "Monitoring teams by overhearing: A multi-agent plan-recognition approach", in *Journal of Artificial Intelligence Research*, vol.17, p.83-135, 2002.

[7] F. Legras, C. Tessier, "Lotto: Group formation by overhearing in large teams", in proceedings of AAMAS, Melbourne Australia, Springer Verlag, pp 425-432, 2003.

[8] D. OSullivan, J. Nùnez-Suàrez, H. brochoud, P.Cros, C. Moore and C. Byrne,"Experiences in the use of the FIPA agent technologies for the development of a Personal Travel Application", in proceeding of Agents, Barcelone, 2000.

[9] E. Platon, N. Sabouret and S. Honiden, "T-compound: An Agent-Specific Design Pattern and its environment", in Proceeding of the 3rd international workshop on Agent Oriented Methodologies at OOPSLA, pp. 63-74,2004.

[10] K. Rothermel, F. Hohl and N. Radouniklis, "Mobile Agent Systems: What is missing?", in H.König, K. Geihs and T. Preuß (eds.) Distributed Applications and Interoperable Systems (DAIS'97), Chapman & Hall, pp. 111-124, 1997.

[11] S. Fünfrocken, F. Mattern: "Mobile Agents as an Architectural Concept for Internet-based Distributed Applications - The WASP Project Approach". In: Steimetz (Hg.) Proc. KiVS'99, Springer-Verlag, pp. 32-43, 1999

[12] D. Traum, J. Rickel, "Embodied agents for multi-party dialogue in immersive virtual worlds", Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 2, pp. 766 – 773, 2002.

[13] L. Tummolini, C. Castelfranchi, A. Ricci, M. Viroli and A. Omicini, ""Exhibitionists" and "voyeurs" do it better: A shared environment approach for flexible coordination with tacit messages" in Proceedings of Workshop E4MAS 2004, Springer Verlag, 2004. pp 215- 231, 2004.

[14] G. Scemama, O. Carles, "CLAIRE-SITI, Public and Road Transport Network Management Control: A Unified Approach», IEE Road Transport Information and Control Conference, Londres, 2004.

[15] G.P. Picco, M.L. Buschini, "Exploiting Transiently Shared Tuple Spaces for Location Transparent Code Mobility", in Proceedings of the 5th International Conference on Coordination Models and Languages, York (UK), F. Arbab and C. Talcott, eds., Springer, LNCS vol. 2315, pp. 258-273., 2002