Property Based Coordination

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

[1] Inrets - Gretia, National Institute of Transportation Research and their Security.
2, av. du Général Malleret-Joinville,
F-94114 Arcueil cedex.
[2] Lamsade, Paris Dauphine University.
Place Maréchal de Lattre de Tassigny,
75775 Paris Cedex 16, France.
{zargayou,balbo,saunier}@lamsade.dauphine.fr

Abstract. For a multiagent system (MAS), coordination is the assumption that agents are able to adapt their behavior according to those of the other agents. The principle of Property Based Coordination (PBC) is to represent each entity composing the MAS by its observable properties, and to organize their perception by the agents. The main result is to enable the agents to have contextual behaviors. In this paper, we instantiate the PBC principle by a model, called EASI -Environment as Active Support of Interaction-, which is inspired from the Symbolic Data Analysis theory. It enables to build up an interaction as a connection point between the needs of the initiator, those of the receptor(s) and a given context. We demonstrate that thanks to PBC, EASI is expressive enough to instantiate other solutions to the connection problem. Our proposition has been used in the traveler information domain to develop an Agent Information Server dynamically parameterized by its users.

1 Introduction

One of the basic problems for the designer of a multiagent system (MAS) is the connection problem [4]. In [10], the authors define the connection problem as "finding the other agents that have the information or the capabilities that you need". In order to solve this problem for the Internet, they use middle-agents. The authors define a middle-agent as an entity that is neither a requester nor a provider of a service but which participates in the interaction between the requester and the provider: a requester has preferences and a provider has capabilities. This interaction model, with in the one hand the capability of a provider and on the other the preferences of a requester, is called "Capability-Based-Coordination" (CBC). Other solutions are proposed to solve the connection problem when some organizational rules are present and when the connection problem depends on these rules. The AGR (Agent, Group, Role) model [5] proposes agents that are defined as an active communicating entity that plays roles (abstract representation of the agent's functionalities) within a group. However, when the connection problem embeds other *ambient* criteria, which we call a context, a new interaction relation has to be defined. Ambient criteria concern conditions that correspond neither to the initiator nor to the receptor of the interaction. Indeed, when the interaction is not a binary but a multipartite relation, being guarded by contextual conditions, a special model is to be built, considering a collective status of the MAS.

The aggregation of the individual status of the agents inside the MAS into a collective one is one of the motivations for the emergence of the multiagent environment as an explicit entity. Several directions are followed in modeling multiagent environments. Some researches focus essentially on the dynamics of the environment [6, 11], and others focus on the modeling of new interaction patterns thanks to the presence of the environment as an explicit entity. For instance, the stigmergy allows agents to leave traces in the environment, which can be used by others to guide their actions. New kinds of interaction recently appeared, like the overhearing and the mutual awareness [7–9, 12]. They envision new types of interaction that enable agents, that are not necessarily protagonists of the interaction (neither emitter nor receptor of a message), to participate in the interaction. These are propositions that investigate alternatives to the traditional messages' broadcast to every agent present in the system.

Our approach generalizes the modeling of interactions inside the environment. We propose a description of the environment based on the principle of observability, instantiating this way the principle of Property Based Coordination (PBC). We define the PBC as a coordination principle for multiagent systems in which: (i) Every entity composing the system, including the agents, exhibits observable properties, (ii) Agents use the observable properties to manage the interactions, perceptions and actions inside the system. Each entity in the environment is then described uniformly by a set of observable properties. Agents being in the center of the MAS modeling, they manage the interactions by specifying the conditions (the context) of the interaction. The CBC and the AGR model are sub-cases of the PBC: the agents exhibit respectively their capabilities and preferences (CBC) and their groups and role (AGR) as observable properties; and of course the dyadic interaction is also a sub-case (identifiers are the observable properties in this case). In order to describe a detailed proposition, we propose a model, that we called EASI (Environment as an Active Support of Interaction). It supports the PBC principle and its formulation is widely inspired from the Symbolic Data Analysis (SDA) paradigm [1]. This model proposes to share the interactions inside the environment and enables the expression of different patterns of interaction. We believe it is the most straightforward way to instantiate the PBC principle.

A model that integrates the other interaction patterns is important because it provides a general framework, that has to be expressive enough, to enable designers to express different configurations, different application scenarios, within the same model. *Tuplespaces* systems, with Linda [3, 2] as a first implementation, resembles to our proposition. They provide a shared collection of data (tuples), and a set of operations (read, write, take) on the collection to manipulate the data. However, the agents need to know beforehand, the location of a tuplespace to interact in. In addition, the expressivity of the querying process does not permit to match different facts e.g. to condition the perception of a tuple to the presence of an other tuple, whereas in EASI, agents are able to express such a condition.

The remainder of this paper is organized as follows: section 2 presents the EASI model. Section 3 details interactions features that our model enables to express. Section 4 shows the use of our model for a traveler Information System. We finally draw general conclusions and perspectives in section 5.

2 Environment as Active Support of Interaction

2.1 Introduction

EASI is a model that instantiates the PBC principle, it proposes an environment model that enables to share the interactions. The problem, when all the interactions are in common, is to enable the agents to find those they are interested in. The formulation of the model is inspired from the Symbolic Data Analysis (SDA) theory [1]. SDA is aimed at discovering *hidden* knowledge within large data sets, modeling both qualitative and quantitative data, which are clustered via the so-called symbolic objects. A symbolic object is a symbolic representation of a subset of entities materialized by a set of conditions that these entities have to satisfy. In our work, the entities composing the environment are agents (active entities) and objects. Both are described via observable descriptions. The reification of the active behavior of agents is their exclusive possibility to *select* their interlocutors by defining autonomously the symbolic objects (*filters* in the model). Depending on the considered MAS, other classifications for the entities are possible; for instance messages, traces, events, services etc. But our classification is the most general that remains consistent with the multiagent paradigm. Indeed, the only finalistic entities (i.e. pursuing an objective) in the MAS are the agents, whereas every entity that is deterministic is considered as an object. Besides, the possibility to manage the properties' privacy i.e. to hide or to exhibit them, enables to model messages (e.g. exhibiting the header and hiding the body of the message), basic services (exhibiting the 'invokable' operations of the service) etc.

2.2 EASI: basic definitions

Definition 1 (Environment) $E = \langle \Omega, D, P, F \rangle$ *where :*

- Ω *is the set of entities, it contains the set of agents (*A*) and the set of objects (*O*).* $\Omega = A \cup O$.
- $P = \{P_i | i \in I\}$, P *is the set of observable properties.* $P_i : \Omega \rightarrow d_i \cup \{null, unknown\}$. P_i *is an application which, for an entity, gives the value for the corresponding property.*
- $D = \prod_{i \in I} d_i$ *with* d_i *the description domain of* P_i.
- F *is the set of filters. A filter is a set of conditions defining the interaction context inside the environment.*

The basic component of our model is an entity. Each entity is described by symbolic variables (observable properties); I contains the indices ranging over P. If an entity does not have a value for a given P_i (this property does not exist for this entity, e.g. the property $radius$ for a triangle), then its value is $null$; if the property exists but does not have a value (hidden by the corresponding entity), its value is $unknown$. Since agents are autonomous, each agent is responsible for the update of its own properties, the latter could be an information about the agent's position in the environment, an activity indicator etc. Each d_i (description domain of P_i) can be quantitative, qualitative as well as a finite data set. A property P_i of an agent a can thus have a value d_i, a can hide it

($P_i(a) = unknown$) or it can be undefined for the corresponding agent ($P_i(a) = null$). The case where an agent hides a given property is temporary and could dynamically be changed during the execution, but the information about the non-definition of P_i is structural and clearly clusters the set A of agents in several categories.

Definition 2 (Agent Category) *$A_C \in A$ is an Agent Category* $\iff \forall a_i \in A_C, \forall P_j \in P$, *if* $P_j(a_i) \neq null \implies \nexists a_k \in A_C | P_j(a_k) = null$

Reduced to this definition, agent categories define a Galois-lattice (cf. Fig.1), with g and h are the Galois correspondences.
$g : \mathcal{P}(A) \rightarrow \mathcal{P}(P), g(A_C) \mapsto \{P_j \in P | \forall a_i \in A_C, P_j(a_i) \neq null\}$
$h : \mathcal{P}(P) \rightarrow \mathcal{P}(A), h(P_C) \mapsto \{a_i \in A | \forall P_j \in P_C, P_j(a_i) \neq null\}$
The top of the lattice represents the agents (may be virtual) which exhibit only null values and the bottom represents the (virtual) agents which exhibit no null values. The designer could define a new top by defining a minimal property (typically an identifier).

Definition 3 (Minimal property of an agent) *$P_j \in P$ is a minimal property* $\iff \forall a_i \in A, P_j(a_i) \neq null$

The same categorization can be applied to the objects O, and provides then a typology of objects present in the system. This information (together with D, the description domains) can be exhibited as a meta-information in the environment, which could be used by the newly coming agents in order to know the interaction possibilities inside the MAS.

In SDA, a symbolic object puts together (in the same 'class') entities according to their description. In order to model an interaction, more than one kind of entities (agents and objects) have to be gathered. A filter f is then defined as a conjunction of symbolic objects (which is a symbolic object too). A filter connects the description of an agent to the *context* that it perceives; a context is a state of the environment at a given moment

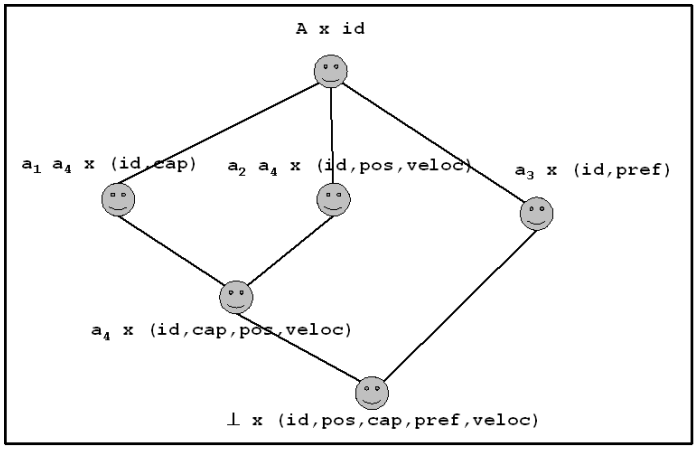

Fig. 1. Agent Categories Galois-lattice

e.g. the presence of a certain agent in the vicinity of a given node. Thus, the definition of a filter is:

Definition 4 (Filter) $f : A \times O \times \mathcal{P}(\Omega) \rightarrow \{true, false\}$
$f(a, o, C) \quad = \quad \wedge_{i \in I_a}[P_i(a) R_i V_i] \ \wedge_{i \in I_o} \ [P_i(o) R_i V_i] \ \wedge_{i \in I_C} \ (\wedge_{c \in C}[P_i(c) R_i V_i])$
$f(a, o, C) \implies perceive(a, o)$

$I_a \subset I$ (respectively I_o and I_C) contains the indices ranging over P that are used in f as selection criteria for the agent (respectively the object and the context). R and V are respectively the binary operators and the values of the descriptions that define the conditions to be held by a, o and C. A filter is the intention definition of the relation between a particular entity a and other entities c in C according to its own description and those of each c, it conditions the perception of a certain o by a. For instance, consider the following filter: $f(a, o, \{a_2\}) = [P_{identifier}(a) = P_{identifier}(a_1)] \wedge [P_{owner-id}(o) = P_{identifier}(a_2)] \wedge [P_{state}(a_2) = \text{“}busy\text{”}]$. The first condition indicates that this filter concerns the perception a given agent a_1; a_1 perceives the objects for which the property $owner - id$ is defined and is equal to the $identifier$ of a given agent a_2 iff the latter is busy. The observable properties of a special agent, the exchange of a message between two defined agents, the presence of a particular object or the combination of these instances can be used to define a particular context. A filter is the conjunction of at least two conditions, the first being related to the receptor and the second being related to o (in this case, there is no special context). This definition implies that the same o can be perceived according to different contexts, or on the contrary that, with the same context, several o can be perceived.

The whole system can have laws that cannot be broken e.g. a light signal cannot pass through an opaque object or an opaque fluid; the control by the environment is possible inside our model in a straightforward way [9]. In fact, the environment can put new filters itself and it manages the whole filters' scheduling; since its filters express the possibilities of interactions, they are executed before those of the agents. In order to express the environment's filters, we introduce the notion of *negative filters*. They are defined in the same way as filters, except that $f(a, o, C) \implies \neg perceive(a, o)$. Instead of enabling perception, the negative filters prevent it. As the environment may manage both filters and negative filters, it is possible to control standard behaviors, by enforcing and/or forbidding certain message transmissions or certain objects perception e.g. radius of perception in situated agents. The negative filters block any further filter that would concern the same a and o. Detailing these features is the purpose of the next section: we show how EASI can be used to supervise protocol executions and how it embeds the classical interaction patterns.

3 Protocol and control

As we argue that the PBC principle embeds the other patterns of interaction, we demonstrate how to express them in a unified way via the EASI model. In this section, we introduce an example of the use of EASI, and we give the different possibilities a designer has thanks to the environment model. We consider a multiagent application in the domain of electronic auctions. There are *seller* agents and *customer* agents. Every

agent has two minimal properties: its "identifier" ($P_{identifier}$) and its "group" (P_{group}). In addition, the *customer* agents have a third property, "interest" ($P_{interest}$), which allows them to express what kind of item interests them, and the *seller* agents have a property "speciality" ($P_{speciality}$), which shows the kinds of goods they sell. The negotiation over the goods is fulfilled via a Contract-Net Protocol (CNP [4]) (cf. Fig. 2). The *customer* agent calls for proposals (cfp) for a good, and the *seller* may reject the

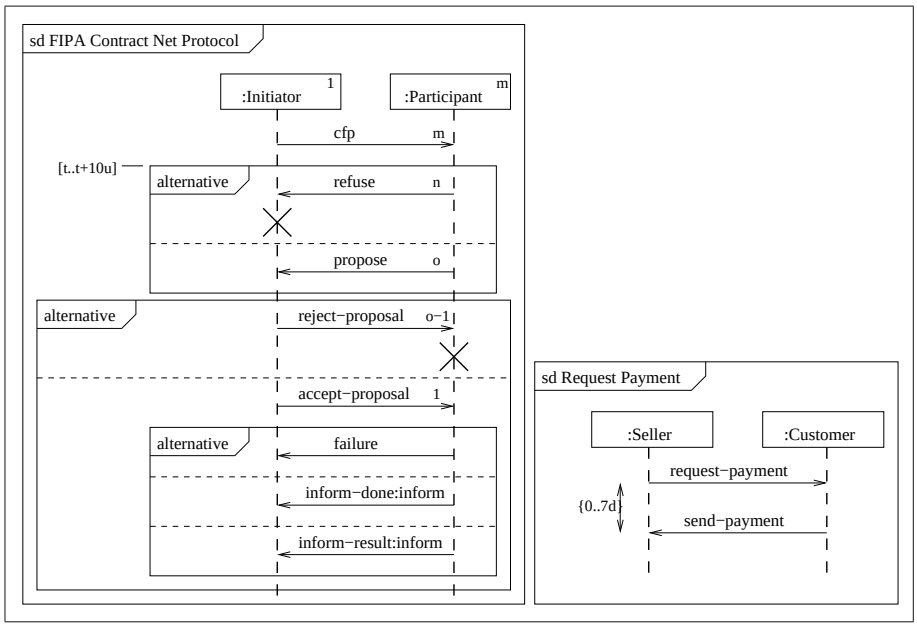

Fig. 2. FIPA Contract-Net Protocol (left) and Payment protocol (right)

cfp call, or make a proposition with a particular price. The buyer chooses one (or none) of the sellers and sends the confirmation or rejection messages to the corresponding agents. The messages are FIPA-ACL compliant, i.e. they exhibit as observable properties the parameters defined in FIPA-ACL [3]: $P_{performative}$, $P_{conversation-id}$, P_{sender} etc. Finally, the *sender* sends the goods and requests the payment via a second protocol. In this MAS, we enable dyadic interaction via a first filter: every addressee of a message should receive it via $f_{Dyadic}(a, m) = [P_{identifier}(a) = P_{receiver}(m)]$. Depending on the policy of the MAS, the first message may be sent to every agent by broadcast, or to selected agents by multicast. The broadcast can be realized for every cfp message with the following filter: $f_{Brodacast\ cfp}(a, m, C) = [P_{identifier}(a) \neq null] \wedge [P_{performative}(m) = \text{“}call - for - proposal\text{”}]$. A multicast is achieved thanks to $f_{Multicast}(a, m) = [P_{identifier}(a) \in P_{receiver}(m)]$, considering that the property $P_{receiver}$ of the message is composed of several addresses.

[3] Foundation for Intelligent Physical Agents, Version J http://www.fipa.org/specs/fipa00061/

We may also integrate other models for particular messages. For example, the AGR (Agent, Group, Role) model proposes agents that are situated in groups and play roles [5] inside each group. If these two pieces of information appear as observable properties, it is possible to manage the interaction in the same way as the Madkit [4], which is the platform dedicated to this model. For instance, the sending of a message to the group of sellers is achieved thanks to $f_{Multicast\ Group}(a, m) = [P_{identifier}(a) \neq null] \wedge [P_{group} = \text{“}seller\text{”}]$. In the same way, it is possible to restrict broadcast to a role. If the initiator sends a message to the agents having a specific role in a group (a restricted broadcast) then the filter is: $f_{RG}(a, m, a1) = [P_{identifier}(a1) = P_{sender}(m)] \wedge [P_{group}(a1) \in P_{group}(a)] \wedge [P_{role}(a1) \in P_{role}(a)]$. This filter restricts the reception to messages which emitter belongs to one of the groups to which the receptor a belongs and who plays one of the roles that the receptor a plays.

Finally, we may explore *ad hoc* interactions, notably by taking advantage of the observability of the property “speciality”. The buyer may send its message to the only sellers that sell a particular kind of goods. This may be achieved either at each message by putting the corresponding filter: $f_{Ad\ Hoc}(a, m) = [P_{speciality}(a) = spec] \wedge [P_{conversation-id} = id]$ with $spec$ the speciality and id the particular id of the call for proposal it sends. Another way to achieve this is to enable a property “speciality” in the cfp messages; which is then managed by a generic filter each time this field is filled: $f_{Ad\ Hoc\ Gen}(a, m) = [P_{speciality}(a) = P_{speciality}(m)] \wedge [P_{performative}(m) = \text{“}call - for - proposal\text{”}]$. The remainder of the messages (proposals etc.) are sent via the dyadic filter, i.e. a classical dyadic interaction. However, the agents may overhear some messages that are not addressed to them, in order to monitor the MAS for example, or to improve their perception of their environment. Let the monitoring agent(s) exhibit $P_{group}(a) = \text{“}monitor\text{”}$. A non-discriminating monitoring filter would be: $f_{Monitor}(a, m) = [P_{sender}(m) \neq null] \wedge [P_{group}(a) = \text{“}monitor\text{”}]$. If the agent wants to overhear only one agent a_1, it modifies the first clause with $[P_{sender}(m) = P_{identifier}(a_1)]$; if it wants to overhear only acceptation messages it adds a clause $[P_{performative}(m) = \text{“}accept\text{”}]$. Other examples of focused overhearing are for a seller to receive the offers of the other sellers in order to be more competitive; or a buyer may want to receive offers it has not issued a cfp for. For the first situation, the filter is: $f_{awareness\ 1}(a, m, \{a_2\}) = [P_{identifier}(a) = P_{identifier}(a_1)] \wedge [P_{performative}(m) = \text{“}propose\text{”}] \wedge [P_{sender}(m) = P_{identifier}(a_2)] \wedge [P_{group}(a_2) = \text{“}seller\text{”}] \wedge [P_{speciality}(a_2) = \text{“}discs\text{”}]$: a_1 overhears all the “propose” messages issued by “discs” sellers. The filter that corresponds to the second situation is: $f_{awareness\ 2}(a, m) = [P_{identifier}(a) = P_{identifier}(a_1)] \wedge [P_{performative}(m) = \text{“}propose\text{”}] \wedge [P_{speciality} = \text{“}books\text{”}]$: a_1 overhears every book proposal message. These examples describe how EASI enables the agents to achieve interactional awareness thanks to their filters and to specialize their perception.

When the negotiation is over, the seller asks the payment of the good(s) to the client (cf. Fig. 2). This protocol must not be overheard, as it may contain secured information. This is achieved by the two following negative filters, which secure the protocol by forbidding the reception of the messages with the performatives request-payment and send-payment by other agents than those specified as “receiver” parameter:

[4] http://www.madkit.org

$f_{OH\ Ban\ 1}(a, m, C) \implies \neg perceive(a, m)$ with $f_{OH\ Ban\ 1}(a, m, C) = (P_{identifier}(a) \neq P_{receiver}(m)) \wedge (P_{performative}(m) = \text{“}request - payment\text{”})$ and $f_{OH\ Ban\ 2}(a, m, C) \implies \neg perceive(a, m)$ with $f_{OH\ Ban\ 2}(a, m, C) = (P_{identifier}(a) \neq P_{receiver}(m)) \wedge (P_{performative}(m) = \text{“}send - payment\text{”})$. As the filter is managed by the environment, even the monitoring filters cannot overrule the perception ban. Note that the filter f_{Dyadic} automatically manages the messages for the intended receiver.

EASI enables the environment to support the MAS protocol management. In addition, thanks to the PBC principle, it enables the use, in the same MAS, of different interaction patterns such as AGR, multicast and indirect interactions. The designer employing EASI may choose for every situation the best solution, which can be dynamic during runtime, and equally use different means to adjust the protocol endorsement to the needs of the system.

4 Application: Agent Information Server

We apply the EASI model to an information server embedding agents representing both human users and transportation web services. The server informs users about transportation networks' status online. Every user has a specific goal during his connection to the server. The transportation web services exhibit their domain of expertise as observable properties. Transportation services providers can provide informations as a response to a request, or asynchronously by sending periodic notifications about disturbances, accidents, special events etc.

4.1 Technical details

We implemented a web server, the environment is a rule-based engine wrapped inside the server, it handles rules (*filters*) and facts (both agents and objects) [12]. The only entities' attributes taken into account by the engine are observable properties. Every web service has a representative inside the server responsible of the convey of messages from the server to the physical port of the web service and inversely. Messages' exchange between the server and the web services are SOAP messages [5] and asynchronous communication is fulfilled through the JAXM api [6] for web services supporting SOAP, and a FTP server otherwise, used this way as a mailbox. This communication heterogeneity is of course transparent for the agents inside the environment i.e. they interact exactly the same way within the MAS environment whatever the transport protocol used is. Every user is physically mobile and connects via a MPTA (Mobile Personal Transport Assistant) to the server, and has during his session a representative agent inside the server, it is his interlocutor during his connection.

4.2 Execution

The problem in this kind of application domains concerns the information flows that are dynamic and asynchronous. Every information source is a hypothetic source of relevant

[5] Simple Object Access Protocol http://www.w3.org/TR/SOAP

[6] Java Api for XML Messaging, http://java.sun.com/webservices/jaxm/

information, thus an agent cannot know *a priori* which information it is interested in, it depends on its runtime changing context. Its context depends on a crossing of different information. Also, all the human users are not the same, their preferences are not homogeneous and their profiles have to be taken into account in their interaction with the system. Let us describe an example of use of the server. There is an abstraction of the transportation network inside the server, composed of stops. They are objects as described in the model. Every stop is described by a line number P_{line} to which it belongs and a position number P_{number} reflecting its position in the line. A user (its representative, say u) is described by its actual position in the network (a couple $(myline, mynumber)$), and by properties expressing its transportation modes' preferences ($mode$), the desired extra-transportation services (coffee-shops, parkings etc.). Basically, u has a path to follow during its trip i.e. a list of triples $(line, number_{source}, number_{destination})^+$, each triple represents a continuous trip, without change. u creates a filter f restricting its interaction to messages dealing with events occurring on its road. For instance, let $path(u) = [(2, 4, 10), (4, 5, 14)]$ reflecting that u has to change at the stop number 10, walk to stop number 5 of the line 4 and continue on the line 4 until the stop number 14. The corresponding filters are: $f_1(u, m) = [P_{id}(u) = myid] \wedge [P_{subject}(m) = \text{“}alert\text{”}] \wedge [P_{line}(m) = 2] \wedge [mynumber \leq P_{number}(m)] \wedge [P_{number}(m) \leq 10]$ and $f_2(u, m) = [P_{id}(u) = myid] \wedge [P_{subject}(m) = \text{“}alert\text{”}] \wedge [P_{line}(m) = 4] \wedge [5 \leq P_{number}(m)] \wedge [P_{number}(m) \leq 14]$. u is interested by the only alerts concerning its own path expressed in f_1 and f_2. It is then notified about every alert event occurring in these segments of network. Since $mynumber$ is updated in every move, the concerned segment is reduced gradually until $mynumber = 10$, then u retracts f_1 and f_2 becomes: $f_2(u, m) = [P_{id}(u) = myid] \wedge [P_{subject}(m) = \text{“}alert\text{”}] \wedge [P_{line}(m) = 4] \wedge [mynumber \leq P_{number}(m)] \wedge [P_{number}(m) \leq 14]$ until the trip ends.

A private transportation operator, say p, has a representative in the server. It has the following filter: $f^p_{awareness}(a, m, a_1) = [P_{id}(a) = myid] \wedge [P_{subject}(m) = \text{“}alert\text{”}] \wedge [P_{line}(m) = P_{myline}(a_1)] \wedge [P_{mynumber}(a_1) \leq P_{number}(m)]$. The message that is caught by u is also caught by p, it knows that u has a disturbance on its road and can propose him an alternative transportation mode. p identifies this way a context in which its target market is involved, it identifies the subset of alert messages that are actually received by travelers and it uses this information to send addressed propositions to its possible customers. The described process is an application of the PBC model and it avoids the periodic requests for new relevant information, handles automatically *ad hoc* events occurring in the network and enables the agents to react following their runtime changing contexts.

5 Conclusion and Perspectives

We propose a generic coordination principle: the Property Based Coordination. Based on the notion of observability, it generalizes existing cooperation schemas as the Capability Based Cooperation, Agent Group Role, Overhearing, Mutual Awareness etc. We also presented the EASI model, a straightforward way to instantiate PBC. It enables the use of the environment as a sharing place where the agents manage their interaction according to their context. Entities composing the environment are agents and objects.

The latter embeds the non-agent entities. With the observable properties, the entities composing the environment can be clustered into agent categories, messages typology etc. which can be available as meta-information in the system. The environment is responsible of the aggregation of the filters to determine who perceives what (e.g. by a rule-based engine) and can restrict the interaction possibilities thanks to filters. In future works, we propose to investigate a formal model of actions in a system based on PBC and its dynamics. Applications from the transportation domain are implemented, in which we experiment new heuristics to solve the scheduling and routing problems, these heuristics are discovered thanks to the retained model of the environment.

References

1. H.-H. Bock and E. Diday. *Analysis of Symbolic Data. Exploratory Methods for Extracting Statistical Information from Complex Data*. Springer-Verlag, Heidelberg, second edition, 2000. 425 pages.
2. N. Carriero and D. Gelernter. *How to Write Parallel Programs: A First Course*. MIT press, Massachusetts, first edition, 1990. 250 pages.
3. N. Carriero, D. Gelernter, and J. Leichter. Distributed data structures in linda. In *Proceedings of the 13th ACM symposium on Principles of programming languages*, pages 236–242, Florida, USA, 1986.
4. R. Davis and R. G. Smith. Negotiation as a metaphor for distributed problem solving. *Artificial Intelligence*, 20:63–109, 1983.
5. J. Ferber, O. Gutknecht, C. Jonker, J. Muller, and J. Treur. Organization models and behavioural requirements specification for multi-agent systems. In *Proceedings of the Fourth International Conference on Multi-Agent Systems (ICMAS'00)*, pages 387–388, Boston, USA, 2000. IEEE.
6. J. Ferber and J. P. Müller. Influences and reaction: a model of situated multiagent systems. In *Proceedings of the second International Conference on Multi-Agent Systems (ICMAS'96)*, pages 72–79, Kyoto, Japan, 1996. AAAI Press.
7. E. Platon, N. Sabouret, and S. Honiden. Overhearing and direct interactions: Point of view of an active environment, a preliminary study. In D. Weyns, H. Parunak, and F. Michel, editors, *Environments For Multiagents Systems I*, volume 3374 of *Lecture Notes in Artificial Intelligence*, pages 121–138. Springer Verlag, 2005.
8. A. Ricci, M. Viroli, and A. Omicini. Programming mas with artifacts. In R. H. Bordini, M. Dastani, J. Dix, and A. E. Seghrouchni, editors, *Programming Multi-Agent Systems III*, volume 3862 of *Lecture Notes in Computer Science*, pages 206–221. Springer Verlag, 2006.
9. J. Saunier, F. Balbo, and F. Badeig. Environment as active support of interaction. In *Proceeding of the third workshop on Environment for Multiagent Systems (E4MAS'06)*, Hakodate, Japan, 2006. to appear.
10. K. Sycara, K. Decker, and M. Williamson. Middle-agents for the internet. In *Proceedings of the 15th Joint Conference on Artificial Intelligence (IJCAI'97)*, pages 578–583, 1997.
11. D. Weyns and T. Holvoet. A colored petri net for regional synchronization in situated multiagent systems. In *Proceedings of First International Workshop on Petri Nets and Coordination*, Bologna, Italy, 2004. PNC.
12. M. Zargayouna, F. Balbo, and J. Saunier. Agent information server: a middleware for traveler information. In O. Dikenelli, M.-P. Gleizes, and A. Ricci, editors, *Engineering Societies in the Agents World VI*, volume 3963 of *Lecture Notes in Artificial Intelligence*. Springer Verlag, 2006.