On the Use of Symbolic Data Analysis to Model Communication Environments

Flavien Balbo and Julien Saunier

LAMSADE, Université Paris-Dauphine
Place du Maréchal de Lattre de Tassigny, Paris Cedex 16
{balbo,saunier}@lamsade.dauphine.fr

Abstract. Recent research on multi-party communications shows how multi-agent communications can take advantage of the complexity of the human communication process. The salient point is the very nature of the communication channels which enable humans to focus their attention on ambient communications, as well as to direct their own communications. For multi-agent systems, the difficulty is the routing of messages according to both the needs of the sender and the needs of the (potential) recipients . This difficulty is compounded by the necessity of taking into account the context of this communication. This article proposes an architecture for the Environment as Active Support for Interaction model (EASI) which is based on a classification data model and supports multi-party communication. Our proposition has been implemented and the functional description of the environment is given.

1 Introduction

Recent research on multi-party communications (MPC) [2,7,14,20,23] shows how multi-agent communications can take advantage of the complexity of the human communication process. In particular, the agents can have opportunistic behavior [12], can monitor the system [9], can have a support for information propagation [4]. The main issue in supporting MPC is to take into account dyadic interaction (one to one), group interaction (one to many) and overhearing (many to one/many) within the same interaction process. In MPC the sender viewpoint is not enough because it does not know all the agents that might be interested in its message. For example, an agent can listen to messages without the agreement/knowledge of the sender through overhearing [20]. Of course, the first MPC issue is the support of MPC itself, the second being the integration of context information, since part of the interaction is contextual and depends on the state of the environment [26]. For a recipient, the usefulness of a message may depend on the context of the sender, *e.g.* its location [11], the context of the message, *e.g.* its theme [4], and the context of the recipient itself, *e.g.* its availability.

These challenges are related to how the recipients are chosen. MPC requires knowledge of the needs of both the sender and the recipients [23]. In [20] and [23], the authors insist that the choice of recipients and the transmission of messages should be done by the environment. The environment is considered to be a first-class abstraction that embodies part of the responsibilities of the multi-agent system [27]. The use of context information requires a mechanism to access it and maintain it. Here, the environment can act as a context server [3]. This paper presents an architecture for the Environment as Active Support for Interaction (EASI) model which is based on a classification data model and supports MPC.

This paper is organized as follows. Section 2 describes the solutions to support interaction in MAS according to the MPC problematic viewpoint. Section 3 presents the adaptation of a classification data model for interaction purposes. Section 4 gives a functional description of the environment. Section 5 compares our proposal with related work; section 6 concludes.

2 Interaction Support

In a message exchange two sub-problems exist, depending on the viewpoint of the agents. From the senders viewpoint, it is a connection problem (CP): which agents are related to my message? The problem is to map the senders needs (information, capabilities[6], resources, ...) to the address of the related agents. From the recipients viewpoint, it is a data extraction problem (DEP): which messages are related to me? The problem is to map the recipients needs to the content of the messages. We classify the solutions to support interaction in three categories: 1) the interaction is dyadic, the solutions are based on direct communication between the agents; 2) the interaction is mediated, the solutions are based on message exchanges through a data space; 3) the interaction is ”*organized*”, the solutions are based on the use of a specific mechanism.

2.1 The Solutions Based on Dyadic Interaction

In an open and heterogeneous MAS, middle-agents are commonly used to look for an agent. The principle is to record in these specialized agents the information needed to look for the recipient with information about agent capabilities for the most usual. When an agent looks for an agent with a specific capability, it sends a message to the middle-agent and receives the list of potential recipients of its request. In the FIPA abstract platform architecture, this principle has been reused for the white or yellow pages directory services. The advantage in using a middle-agent is that it can support other services such as ensuring the anonymity of the agents. Depending on the services, a middle-agent taxonomy has been introduced in [29]. The problem created by centralizing the information is offset by the possibility of having several middle-agents. This solution implies common knowledge about these intermediaries and data updating has to be taken into account. Hence the success of this approach where information about agent capabilities means that not much data updating is required. Another frequently used solution in open and heterogeneous MAS is the Contract Net Protocol (CNP) [6]. The initiator broadcasts its request with the criteria that the agents have to satisfy to be selected for the following interaction process. If they want, recipients respond with their auto evaluation on the criteria. The success of this protocol relies partially on the fact that both sender and recipients are involved in the selection process [22]. Moreover, since the criteria are chosen according to the needs of the sender and independently of the protocol, the CNP has easily been adapted for several applications. The constraint is that it needs a broadcast at the beginning of the protocol. All these solutions are adapted to solve CP but they do not take into account DEP.

2.2 The Solutions Based on a Data Space

These solutions require the sender to put its message in a data space where the recipients read their messages. This data space can be external or can be a component of the MAS environment. In the first category, based on the LINDA model, there are the tuple spaces that are a continuation of the blackboard architecture. The aim of this approach is to avoid the space and time synchronization problems related to point-to-point communication. In this approach, the messages are tuples, and templates are used to look for them. The matching is done by comparing the message tuples and the template tuples according to the position and type of data used to describe the messages. TuCSoN has been developed using this model; it enables the tuple space to be programmed in order to specialize its reaction to the events, the result being what is called a Tuple center [18]. Using the same model, there is also LIME [17], which is a tuple space model for mobile agents.

Solutions based on the use of the environment extend the principle of a shared data space to take into account an interaction support in which the agents evolve. [19,27] propose a functional architecture of the environment which integrates a layer to support the interaction between agents. Communication between agents mediated by the environment comes from the reactive agent community and is often based on the stigmergy principle. Extended to cognitive agents, this trace mechanism has been used in [21] and [26], where all agent interactions are traces that can be observed by other agents and give extra information in addition to that contained in the message. In [8] the agents communicate through a tuple space that is integrated into the environment. These solutions take the viewpoint of the recipients and cannot be used to solve the CP.

2.3 Solutions Based on a Specific Mechanism

These mechanisms are based on the use of meta-knowledge or of a network protocol. In closed MAS and/or when the agents evolve within a platform, the MAS organization gives information to address messages. For example, in the Madkit platform[1], which is based on the organization model *aalaadin*, the agents communicate according to their role or their group in an organization. The structure of the organization is recorded within what is called the kernel. When an agent sends a message to agents according to a role, it sends the message to the kernel which looks for the recipients and puts the message into each of their letter boxes. In Magique[2], the MAS is organized within a hierarchical network where each agent is located on a node and the intermediaries nodes contain the skills of the agents situated below in the hierarchy. An agent that is looking for a skill sends the message to its superior and the message follows the hierarchy until the skill has been found. In these two examples, the agents do not have to know the address of a middle-agent or a tuple-space, they just use a common structure, their organization. Because the messages are not addressed according to the agent identifier or address, this solution improves the robustness and the efficiency of the MAS but limits communication to organization-based criteria. These solutions take the viewpoint of the sender and are solutions for CP.

[1] www.madkit.org

[2] http://www2.lifl.fr/SMAC/projects/magique/

The mechanism can be based on a network protocol. In [4], the authors use broadcasts restricted to specialized channels. The messages are sent on a channel to which the recipients have subscribed. Each channel has an IP address and is described by a string (following a taxonomy) in an XML file located on a URL. When an agent has chosen its channel it waits for messages. The messages are sent using the multicast function of the UDP protocol. In SIENA [5], the authors proposed a content-based routing protocol. Routing is done by filters that are introduced by recipients according to notifications that have been made by a sender. A notification is an item of information about the information that the sender could send on the network. The problem is to find the subset of recipients for each message and algorithms can improve filter management efficiency [25]. These solutions take the viewpoint of the recipients and are not suitable for the CP.

This discussion about the principal solutions that have been proposed to support interaction between agents shows clearly that each one is suitable for just one of the sub-problems. In this paper, we propose the EASI model, which takes into account, together and separately, the needs of both senders and recipients. The advantage is to have a single model that works with several interaction models and can easily combine them in order to take into account multi-party communication.

3 Communication Environment

In order for messages to be routed successfully, the environment has to stock a large set of data related to the agents, the message and everything necessary to know the context of the communication. In EASI, when a message is sent the environment searches through these data to choose the recipients. It has to gather information about the needs of the sender, the needs of the potential overhearers, and the context. Hence, our model has to address three issues: (1) an efficient search for recipients, (2) an expressive data description model and (3) a straightforward applicability.

Concerning issue (1), the environment contains information on all the components of the MAS and our objective is to let the agents use this common knowledge to define their interaction needs. For example, if information about the relationship between agents is available, then an agent a could send a message to the agents that it has in common with another agent b; in addition, an agent c, a friend of a but not of b and that might be interested could overhear the message. The amount of data increases fast, but the environment has to rapidly find the recipients and deliver the messages. Therefore, the model has to enable the efficient description and organization of large data clusters. Issue (2) is that of data representation, which should be expressive and enable a unified management of the descriptions and of the needs of the agents. The last issue (3) concerns the use of the model: it should be independent of the implementation, but also quickly applicable in real applications thanks to existing technologies.

These issues have led us to base our model on Symbolic Data Analysis (SDA) [1] in order to formalize the environment. SDA relies on the logic concepts of intension and extension to describe and classify data clusters. Its formalization is expressive since it does not depend on the type of data (quantitative, categorical or multi-valued), the comparison operators are given by the designer and variables can be used. SDA is applicable because the cluster definitions can be translated with SQL and/or in first-order logic.

Ω	p_1	p_2	p_3	p_4	...	p_{np}
ω_1	“a1”	“Main Hall”	44	false	...	$p_{np}(\omega_1)$
ω_2	“a2”	“A209”			...	$p_{np}(\omega_2)$
ω_3	“a3”	“A209”	37	true	...	$p_{np}(\omega_3)$
ω_4	“a4”	“A209”	33	true	...	$p_{np}(\omega_4)$
....	...	...	...	...	...	...
ω_n	$p_1(\omega_n)$	$p_2(\omega_n)$	$p_3(\omega_n)$	$p_4(\omega_n)$	...	$p_{np}(\omega_n)$

Fig. 1. Symbolic data table: n individuals and p properties

3.1 Symbolic Data Modeling

Let us begin by introducing the basic SDA definitions. The real world is made up of n individuals $\omega \in \Omega$. Each individual has np properties, and p_j is a mapping from Ω to D_j which associates to each $\omega \in \Omega$ a value in the definition domain D_j of the j^{th} property and $D = (D_1, ..., D_{np})$. For instance, individual ω_1 has four properties p_j: its identifier (string), its location (in a set of building positions), its age (number) and its availability (boolean). The values associated to ω_1 are $p_1(\omega_1) = "a1"$, $p_2(\omega_1) = "MainHall"$, $p_3(\omega_1) = 44$ and $p_4(\omega_1) = false$.

Considering the values of all the properties $p_j,\ j = 1, ..., np$ for the i^{th} individual, $d_{\omega_i} \in \mathcal{D}$ is the description of ω_i in the model. In the previous example, the description d_{ω_i} of the individual ω_1 is $"a1"$, $"MainHall"$, 44, $false$.

In practice, the symbolic data related to a given set of individuals are represented in an $n \times np$ matrix. The columns are the properties and the rows are the individuals. Figure 1 contains a symbolic data table, where the first rows and columns have been filled out. Not all the properties make sense for all the individuals, for instance the *age* of a message.

In the real world, the individuals can be grouped together thanks to concepts. The concepts $C_k,\ k \in \mathbb{N}$ are intents, they describe descriptions which satisfy certain individuals. An example of a concept C_1 would be all the individuals below 40 situated in room A209.

The extent of a concept is the set of individuals which satisfy this intent. In the table, the extent of the concept “younger than 40, in room A209” is the set $\{\omega_3, \omega_4\}$. An assertion is a symbolic object that is a mapping $\Omega \rightarrow \{true, false\}$. Let $v = v_1, ..., v_{np}$ be the description of an individual or a concept i, with v_j data that may be quantitative, categorical or multi-valued. An assertion is defined as:

$as = [p_{j1} R_{j1} v_{j1}] \wedge ... \wedge [p_{jq} R_{jq} v_{jq}]$ for $1 <= j1, ..., jq <= np$, where R_j is a comparison operator between the property p_j and the value v_j. The set of symbolic objects is $\mathcal{S}$. For example, the assertion $as_1 = [p_2(\omega) = A209] \wedge [p_3(\omega) < 40]$ is the description of the concept C_1. A symbolic object is an intent description. Its extent that is $E(a) = \{\omega \in \Omega | a(\omega) = true\}$ contains all the individuals which satisfy the comparisons with the description values of the assertion. For example, the extent of a_1 is a class of individuals which contains ω_3 and ω_4.

An assertion is therefore a comparison between the description of an entity and given values. It defines a class of entity which contains all the entities that satisfy these comparisons. A concept is also "*what the user needs*" with an exact but unknown extent

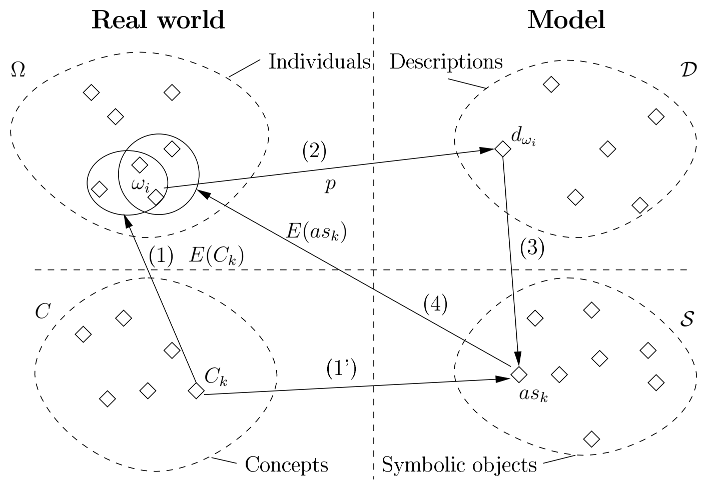

Fig. 2. Real world and model in Symbolic Data Analysis

in Ω, and a symbolic object is its formalization in the model with an imperfect but computable extent in Ω.

Figure 2 sums up the SDA definitions. The real world is made up of individuals and concepts. The concepts are intent descriptions of one or more individuals, for example "the agents available and situated in room A209". The extent of a concept (relation (1)) contains individuals $\omega_i \in \Omega$ which, taken as a whole, are a class. To each individual of the real world corresponds a description $d_{\omega_i} \in \mathcal{D}$ in the modeled world, thanks to the mapping p (relation (2)). A symbolic object is the formalization of a concept C (relation $(1')$). Symbolic objects $\mathcal{S}$ are intent descriptions using the descriptions of the individuals d_{ω_i} (relation (3)). The extent of a symbolic object contains the individuals of the real world whose description satisfies its description (relation (4)).

3.2 Communication Routing

This section shows how the Symbolic Data Analysis model is adapted to design a communication environment; more details about the EASI syntax can be found in [23]. In EASI, the environment contains the data for communication (Ω, $\mathcal{D}$) and information about how to manage it ($\mathcal{S}$). The concept is found at the MAS design level and not in the environment model. Some adaptation is direct: an individual is called an entity and Ω is the set of entities. An entity is the description of a component of the multi-agent system: agent, message, or object, the last one being a generic term to take into account anything which is neither an agent nor a message and that can be used to give information about the context of an interaction. For example, rooms are objects which can be described (size, capacity, facilities, etc.) and a message could be addressed according to this information: "message to the agents in the closest room". There is a distinction between the real component and its description in the EASI model. For example, an agent has its own process and knowledge, and a description of it is recorded in the environment. The set of properties $\mathcal{P}$ is the ontology of the communication environment, that is to say the information that can be used for communication purposes. $\mathcal{D}$ is the set of descriptions of the entities and completes the ontology with the values of the entities properties that are currently recorded in the environment.

Remember that a concept in ADS is an intent description of individuals. It represents what the agents need to identify in Ω, that is to say in all their common knowledge. A concept is reified by a symbolic object with an extent that is computed in Ω in order to directly identify the entities, or in $\mathcal{S}$ in order to find the symbolic objects that are related to the same need. There are three types of concept. The first type is related to the category of entities. A category of entities is a set of entities that is described by the same properties. This property set is called the *Pdescription* of the category, and the entities are clustered using the existence condition of required properties. Let C be a category and P_C its *Pdescription*; the assertion $cat_C = [P_C \subset P_\omega]$ is *true* for an entity $\omega \in \Omega$ if P_C is included in its own *Pdescription* (a $null$ value for a property expresses the absence of this property [23]. For example, let *FIPA* be the category of the FIPA messages that are recorded in the environment, let P_{FIPA} be the *Pdescription* of this category, $E(P_{FIPA}) = \{m \in \Omega | cat_{FIPA}(m) = true\}$ contains all the entities that have the description of a FIPA message (the FIPA language component is the properties in this case). This description level clusters the data in Ω according to their link with the communication needs. The advantage is that this link is independent of the property values and thus is independent of the update process.

The second type of concept is related to the communication needs of the agents, for example transmitting a message to "the agents available and situated in room A209". A filter is the reification in $\mathcal{S}$ of this need and gives the intent description of the constraints on the entities that are related to a connection. A filter $f \in \mathcal{F} \subset \mathcal{S}$ is a tuple $\langle f_a, f_m, [f_{co}], n_f, [priority_f], initiator_f \rangle$. The first three elements are assertions: f_a is the intent description of the constraints on the recipients, f_m is the intent description of the constraints on the messages and f_{co} is the intent description of the constraints on the context. The other elements are name, priority and initiator (agent that adds this filter to the environment) of the filter. Each assertion has a *Pdescription* too, for example P_{f_a} is the *Pdescription* of f_a and contains the properties that a recipient must have to be taken into account as a recipient.). The extent of a filter according to the property existence constraint in Ω contains the potential components of a connection that are gathered in the tuple $\langle E(P_{f_a}), E(P_{f_m}), E(P_{f_{co}}) \rangle$, with for example $E(P_{f_a}) = \{a \in \mathcal{A} | \forall p_i \in P_{f_{ag}}, p_i(a) \neq null\}$ with $\mathcal{A} \subset \Omega$ the agents set.

The last type of concept is related to the need to manage the relation between the entities and the symbolic objects. Basically, the idea is to match the $Pdescriptions$ of the entities to those of the assertions of the filters. A symbolic assertion is a mapping $\mathcal{S} \rightarrow \{true, false\}$. For a symbolic object, the symbolic assertion sa takes the value $true$ if the symbolic assertion is valid and false if it is not. For example, the relation between a message and its filters is designed as a concept: which filters f enable the message m to be received. Its reification is given by the symbolic assertion $sa_m(f) = [P_{f_m} \subset P_m]$ and its extent in $\mathcal{F}$ is $Channel_m = \{f \in \mathcal{F} | as_m(f) = true\}$, with P_{f_m} the *Pdescription* of the message related to the filter f. The result of this extent is a new symbolic object that we call SO and that has an extent in Ω. This extent is the union of the extent of the symbolic objects belonging to it and gives a solution to identify

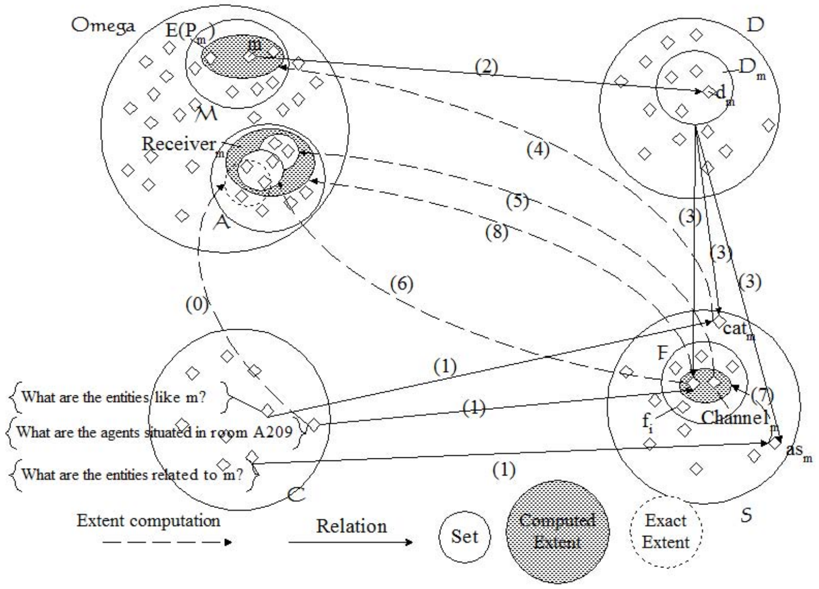

Fig. 3. Interaction component modeling

the relation between entities. For example, the extent of $Channel_m$ in Ω contains the entities that have in common the messages m; more precisely, this extent contains all its recipients ($recipient_m = \{a \in \mathcal{A} | \exists f \in Channel_m, a \in E(P_{f_a})\}$) and the whole context ($Context_m = \{C \subset \Omega | \exists f \in Channel_m, C \in E(p_{f_m})\}$).

In EASI, the filters dedicated to the management of the MAS belong to the environment. These filters are introduced either by a group of system agents, or by a mechanism which is internal to the environment. Thus, the filters are partitioned in two categories, depending on their initiator: $\mathcal{F} = \mathcal{F}_E \cup \mathcal{F}_A$, where $\mathcal{F}_E$ is the set of filters introduced by or on behalf of the environment and $\mathcal{F}_A$ is the set of filters introduced by the agents. In this way, the environment can add messages to the agents, i.e. they will receive messages that would not have been received otherwise.

Figure 3 shows the components of the EASI model and their relations. In Ω the entities are described with a value for each of their properties. This set is modified according to the update process of the MAS, such as D that gives a global overview of the entity values and therefore of the state of the MAS. $\mathcal{S}$ contains the tools to manage the recorded information. For example, a message is an entity m in Ω and d_m in $\mathcal{D}$ records its values (2). $\mathcal{D}_m \subset D$ contains all the values for each property that are related to the same category as m and enables the *Pdescription* P_m to be computed. According to this description and following the initial concepts (1),the assertion cat_{p_m}, the filters and the sa_m are computed (3). The extent (4) of the assertion (cat_m) related to the *Pdescription* P_m gives the entities that share the same properties. This cycle is used to find a subset of entities from one of their component or from a *Pdescription* coming from a user need (a concept). A filter is a subset of assertions and its extent is computed in Ω on the tuple that is composed of the extents (5-6) (only the extent for the agents

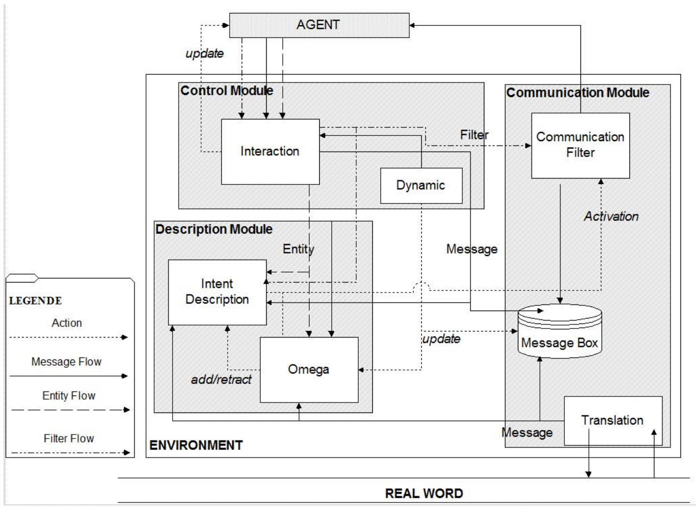

Fig. 4. An Environment Functional Description

are given in Fig. 3). The objective is to limit the processing to just the entities that are related to the interaction while remaining independent of the update process of the entities. The symbolic assertion sa_m and its extent $Channel_m \in \mathcal{S}$ are computed (7). The extent in Ω is the union of the extent of the filters (8). These data clusters can be computed *a priori* according to the description of the entities that are related to the interaction process in the MAS, which means that only the link between a new entities and these clusters has to be computed.

4 Functional Description of the Environment

This section provides a functional description of the environment (figure 4) that has been developed using the interaction model EASI. The proposal is based on the abstract functional description of the environment in [28]. The description is divided into two parts, firstly a description of the internal modules, then the common cases of interaction between agents and the environment.

4.1 Description of the Environment Modules

Our environment is composed of the modules related to the processing of the description of the MAS components (*Description Module (DM)*), to the processing of the communication (*Communication Module (CoM)*) and to the control of the communication (*Control Module (CM)*).

DM is composed of the sub-modules that manage the extent (*Omega* module) and the intent (*Intent Description (ID) module*) descriptions of the MAS components. *Omega* contains all the entities and enables modification operations (add/retract/modify).

ID contains the organization of the data as described in section 3.2. It contains the *Pdescription* of the entities, and for each filter the *Pdescription* of the assertions composing it and the symbolic objects that are related to the link between the entities and the symbolic objects. For example, for a filter f, *ID* contains P_{f_a}, P_{f_m},$P_{f_{co}}$ and for each of these sets their links with the other symbolic objects like $Channel_m$. *ID* maintains the link between the intent description and their extent in *Omega*, and the module containing the filters. When an entity *Pdescription* is added, if it is already recorded then the reference to the entity is added to all the structures the *Pdescription* is related to; in the other cases all the structures have to be tested. At this point the entity *Pdescription* are never deleted and this operation is under study. When the $Pdescriptions$ related to a filter are added, *ID* looks for SO where the filter has to be recorded; in the case of failure, new SO are created according to the *Pdescription* of the assertions.

CoM is composed of the *Message Box* sub-module where messages are stored, the *Communication Filters* sub-module where the filters are stored and triggered, and the *Translation* sub-module, which is an interface with the non EASI world. *Translation* stores received messages in the *Message Box* and "translates" them using the EASI model. *Communication Filters* contains the filters of the agents ($\mathcal{F}_A$) and of the environment ($\mathcal{F}_E$). Filters are activated according to the modification in *ID*, which is then matched against the entity in *Omega*. When the matching is successful, the related descriptions are sent to the agent with the real message that is in *Message Box*.

CM contains the *Interaction* sub-module, which controls interaction between the agents and the environment, and *Dynamic*, which controls the update process. *Interaction* distributes the input of the agents in the environment (entity, messages, filter) and sends the agents the output of the environment (messages, update action). *Dynamic* maintains up-to-date *Omega*, it deletes messages that are too old (if a property related to time exist) or initiates an update of the agents properties. The objective is to give the MAS a global strategy in order to limit the cost of the update process. For instance, if agents have a property that gives their position, at least two strategies are possible: agents update their property each time they move or only when they are requested. A compromise has to be found between the number of errors and the cost of the update process.

4.2 Description of Common Cases

This section gives an example of MPC and concerns the dynamic management of resources in MAS. Each group of agents has a set of resources and, to simplify, each resource is unique and indivisible. An agent that needs a resource does as follows: it anticipates its needs and tries to know which agent in its group has its next resource or, if it cannot anticipate this, looks for the agent that has it. The owner of a resource accepts or refuses to give it to the requester. The objective is to avoid contacting all the agents of the same group each time one of them needs a resource.

Firstly, the agents record themselves in the environment. Each agent adds an entity to the environment that describes it (there is only the $group$ property in our example). *Interaction* adds the properties id (the value is unique and identifies the agent in the environment) and $address$ (the value is the address where the message will be sent). The entity is added to *Omega* and the *Pdescription* in *ID*. The latter module looks for symbolic objects that are related to this new *Pdescription*. For each of them that is modified the related filters are tested.

When an agent knows which resource it will have to use, it adds to the environment filters that match it against the description of the messages that are exchanged when an agent asks for a resource. There are two filters: the first (succeed) is related to when the resource owner agrees to give it and the second (fail) to when it refuses. *Interaction* adds each filter to *Communication Filters* and computes the *Pdescription*s (for example P_{fa}) that are added to *ID*. This module computes the extents in *Omega* and adds the result to *Communication Filters* to be triggered. If successful, the result is sent to the agents. The filter related to the success of the request informs the overhearer of the new owner of the resource and its behavior is not modified. The filter related to the failure of the request informs the overhearer of the present owner of the resource. In this case, the overhearer withdraws its filter related to the failure of the request. This filter will be added again if the resource owner responds favorably to a new request. The objective is to limit the number of useless messages. In the environment, the filter is deleted from *Communication Filter* and the related $Pdescriptions$ from *ID*.

When an agent needs to request a resource, it uses a filter that is already in the environment. This filter belongs to $\mathcal{F}_E$ and the content of this set is known to all the agents. Like in [28] the environment may or may not be distributed. In the first case, communication between environments is done through *Translation*; in the second case it is done directly with *Interaction*. In the two cases, the environment process is the same. The entity related to the message is added to *Omega* and its *Pdescription* is added to *Intent Description*. The choice of properties related to a message can be parameterized and additional properties like the date of the reception and the address of the recorded value in *Message Box* can be added. *ID* looks for the symbolic objects that are related to this new description. The set $Channel_m$ is added to *Communication Filters* to be triggered.

5 Discussion

This section compares our proposal with those presented in section 2, using three criteria. The first is the *expressiveness* of the proposals. A proposal is more expressive than another if it can take into account more complex constraints in the search for agents related to the interaction. The second criterion is the *completeness* of the proposal. A proposal is more complete than another if it can be used in more interaction models. The third criterion is the possibility of having *context aware* interaction. A proposal is more *context aware* than another if it can take into account more complex constraints on the context in the search for agents related to the interaction.

5.1 Expressiveness of the Delivery Mechanism

EASI is a model that aims to organize the data required to realize an interaction. For solutions based on dyadic interaction, it is necessary to compare our proposal with the matching process performed by the matchmaker or the sender of the messages. The matchmakers that come closest to our proposal are those that enable content-based routing [15,24]. In the case of solutions based on the CNP, the sender receives the evaluation of the agents on the requested properties and can also apply at least our selection process. For solutions based on a tuple space, it is necessary to compare matching on tuples and our solution. In tuple-spaces, the matching is limited to the type, position and value of the components of the tuples, while EASI enables the use of comparison operators, and the matching between entities. This is because in these works, the authors focused on the interaction model and not on the data model. In [8] the data model is not described, only its organization is: an input and an output matrix. The aim, for each item of data, is to find its link with agents. The senders are the lines of the matrix *outbox* and the recipients are the lines of the matrix *inbox*. As in EASI, set modeling gives a sufficient level of abstraction to manage the data and to use projections to identify the searched data. The operators that are used to do these projections are not included in the model and depend on the implementation phase.

In SIENA [5], a content-based routing model, filters are matched against single notifications (a tuple $<< type, name, value >^{+}>$) and patterns are matched against one or more notifications. The matching is done by comparing the value in the notification and the constants in the patterns. There are no variables that can be used to generalize the patterns. Moreover, in EASI part of the matching is done on the recipient and the context although only the data are evaluated in SIENA. Based on this criterion, this work comes closest to our proposal.

To sum up, in the majority of this research, the expressivity of the model is not taken into account and emphasis is put on other problems such as the interaction model in the tuple model or scalability in the delivery mechanism. This means that each of these solutions is complementary to the EASI model.

5.2 Completeness of the Delivery Mechanism

Using this criterion, the evaluation depends on the ability of a solution to enable: 1) a sender to find a recipient; 2) a recipient to choose its messages; 3) mutual-awareness. The first two points have been discussed in section 2 and we have shown that each of the solutions has been designed to take mainly into account the sender or the recipient. Mutual awareness is the possibility to receive messages that are sent to other agents. From the delivery mechanism viewpoint this means that the sender can choose its recipient and that another agent than the recipient can choose to receive the message. The delivery mechanism must also support both the search for recipients by sender and the search for messages by recipients. This is why the other solutions that take mutual awareness into account are mainly based on broadcast [13,16].

If mutual awareness is extended to the case where an agent can choose to receive messages coming from a subset of agents, then mutual awareness is a subcase of the delivery mechanism used to choose a message. In this case, solutions based on a shared data space and content-based routing solutions can be used if the data exchanged contains information about the sender. However, the advantage of anonymity found in these solutions is lost. Middle-agents mediate the search for receivers. Instead of delivering the messages according to both the senders and the receivers needs, the choice is restricted to a subpart of it. Furthermore, middle-agents are autonomous, while the environment is a supporting infrastructure [27].

The channeled multicast is a restricted broadcast and the recipients can be described as listeners of a subset of subjects, where a subject could be an agent. With the use of filters, EASI takes the needs of the sender and the recipients into account independently and simultaneously, according to the agent that has introduced the filter. If only the needs of the sender are taken into account (the filter is introduced by the sender of the message) then EASI is used as a delivery mechanism to choose the recipients. If only the needs of the recipients are taken into account (the filter is introduced by the recipients of the message), then if the message is not addressed EASI is used as a delivery mechanism to choose messages. If the message is addressed (filter introduced by sender) and heard (filter introduced by recipient) then EASI is used as a delivery mechanism to choose both recipients and messages and thus enables mutual awareness.

5.3 Expressiveness of Context Awareness in the Delivery Mechanism

The evaluation of each solution depends on if and how it can take the context into account in the search of the recipient or the message. This criterion is related to the expressiveness of the solution and to the availability of context-related information. Expressiveness is related to the expressiveness of the delivery mechanism and has already been discussed in 5.1. This criterion implies that the delivery mechanism has access to the state of the components of the MAS. Not all solutions have been designed to support context-awareness interaction: middle-agent, CNP, organization and channeled multicast. These solutions cannot take into account any information other than that related to the message itself. The solutions based on a shared data space or content-based routing have been designed to support context awareness. The only restriction is about the information available. In these solutions, information about agents is not taken into account and so the interaction cannot be conditioned by the observable state of the agents involved.

6 Conclusion

Designing a middleware to support MPC not only requires taking the needs of the agents related to communication into account but also evaluating these needs according to the context. The most common proposal is the use of a dedicated blackboard [7,10], which is a static medium for sharing messages. Thanks to the design and use of filters, EASI is an answer to these issues. The advantages of our proposal, based on the ADS clustering model, are connected to the different ways in which the information is managed. It can be done at different levels of abstraction, from an entity to a concept. The advantage is to limit the cost of the entity-updating process by the use of direct links (concepts) between the entities and the data clusters (the concept *extents*). These links are computable *a priori* and are dynamically changed if the MAS is open and new entity categories are added. Another advantage is to be able to obtain a description of the interaction facilities of the environment directly. This description is related to the ontology of the description of the entities with their existing value and to the filter description too. A filter (specially the environment filter) can be described as an interaction service: dyadic, broadcast, multicast, related to position, *etc*. Finally, the homogeneous description of the information related to entities and filters (their $Pdescriptions$) simplifies a processing that can be based on the same structure (under study) that is used to record them. Our proposal can be distributed with agents that are recorded in several environments or with environments that are federated. The environment nevertheless remains centralized and will be "really" distributed when the data model is distributed. Future work is related to the distribution of the data model, including the distribution of the description of the entities and filters.

References

1. Billard, L., Diday, E.: Symbolic Data Analysis: Conceptual Statistics and Data Mining (Wiley Series in Computational Statistics). John Wiley, Chichester (2007)
2. Branigan, H.: Perspectives on multi-party dialogue. Research on Language & Computation 4 (2-3), 153–177 (2006)
3. Bucur, O., Beaune, P., Boissier, O.: Steps towards making contextualized decisions: How to do what you can, with what you have, where you are. In: Roth-Berghofer, T., Schulz, S., Leake, D.B. (eds.) MRC 2005. LNCS (LNAI), vol. 3946, pp. 62–85. Springer, Heidelberg (2006)
4. Busetta, P., Donà, A., Nori, M.: Channeled multicast for group communications. In: AAMAS 2002: Proceedings of the First International Joint Conference on Autonomous Agents and Multi-agent Systems, pp. 1280–1287. ACM Press, New York (2002)
5. Carzaniga, A., Rosenblum, D.S., Wolf, A.L.: Design and evaluation of a wide-area event notification service. ACM Transactions on Computer Systems 19(3), 332–383 (2001)
6. Davis, R., Smith, R.G.: Negotiation as a metaphor for distributed problem solving, 333–356 (1988)
7. Dignum, F., Vreeswijk, G.: Towards a testbed for multi-party dialogues. In: Workshop on Agent Communication Languages, pp. 212–230 (2003)
8. Gouach, A., Michel, F., Guiraud, Y.: Mic*: a deployment environment for autonomous agents. In: Weyns, D., Van Dyke Parunak, H., Michel, F. (eds.) E4MAS 2004. LNCS (LNAI), vol. 3374, pp. 109–126. Springer, Heidelberg (2005)
9. Gutnik, G.: Monitoring large-scale multi-agent systems using overhearing. In: AAMAS 2005: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems, p. 1377. ACM Press, New York (2005)
10. Huget, M.-P., Demazeau, Y.: First steps towards multi-party communication. In: van Eijk, R.M., Huget, M.-P., Dignum, F.P.M. (eds.) AC 2004. LNCS (LNAI), vol. 3396, pp. 65–75. Springer, Heidelberg (2005)
11. Julien, C., Roman, G.-C.: Supporting context-aware interaction in dynamic multi-agent systems. In: Weyns, D., Van Dyke Parunak, H., Michel, F. (eds.) E4MAS 2004. LNCS (LNAI), vol. 3374, pp. 168–189. Springer, Heidelberg (2005)
12. Kamali, K., Fan, X., Yen, J.: Towards a theory for multiparty proactive communication in agent teams. International Journal of Cooperative Information Systems 16(2), 271–298 (2007)
13. Kaminka, G., Pynadath, C., Tambe, M.: Monitoring teams by overhearing: A multi-agent plan-recognition approach. Journal of Artificial Intelligence Research 17, 83–135 (2002)
14. Kumar, S., Huber, M.J., McGee, D., Cohen, P.R., Levesque, H.J.: Semantics of agent communication languages for group interaction. In: Proceedings of the Seventeenth National Conference on Artificial Intelligence, pp. 42–47. AAAI Press / The MIT Press (2000)
15. Kuokka, D., Harada, L.: On using kqml for matchmaking. In: ICMAS, pp. 239–245 (1995)
16. Legras, F., Tessier, C.: Lotto: Group formation by overhearing in large teams. In: Dignum, F.P.M. (ed.) ACL 2003. LNCS (LNAI), vol. 2922, pp. 254–270. Springer, Heidelberg (2004)

17. Murphy, A.L., Picco, G.P., Roman, G.-C.: Lime: A coordination model and middleware supporting mobility of hosts and agents. ACM Trans. Softw. Eng. Methodol. 15(3), 279–328 (2006)
18. Omicini, A., Zambonelli, F.: Tuple centres for the coordination of internet agents. In: SAC 1999: Proceedings of the 1999 ACM symposium on Applied computing, pp. 183–190. ACM Press, New York (1999)
19. Platon, E., Mamei, M., Sabouret, N., Honiden, S., Parunak, H.V.D.: Mechanisms for environments in multi-agent systems: Survey and opportunities. Autonomous Agents and Multi-Agent Systems 14(1), 31–47 (2007)
20. Platon, E., Sabouret, N., Honiden, S.: Tag interactions in multiagent systems: Environment support. In: Weyns, D., Van Dyke Parunak, H., Michel, F. (eds.) E4MAS 2006. LNCS (LNAI), vol. 4389, pp. 106–123. Springer, Heidelberg (2007)
21. Ricci, A., Omicini, A., Viroli, M., Gardelli, L., Oliva, E.: Cognitive stigmergy: Towards a framework based on agents and artifacts. In: Weyns, D., Van Dyke Parunak, H., Michel, F. (eds.) E4MAS 2006. LNCS (LNAI), vol. 4389, pp. 124–140. Springer, Heidelberg (2007)
22. Sandholm, T.W.: An implementation of the contract net protocol based on marginal cost calculations. In: Proceedings of the 12th International Workshop on Distributed Artificial Intelligence, Hidden Valley, Pennsylvania, pp. 295–308 (1993)
23. Saunier, J., Balbo, F.: An environment to support multi-party communications in multi-agent systems. In: Burkhard, H.-D., Lindemann, G., Verbrugge, R., Varga, L.Z. (eds.) CEEMAS 2007. LNCS (LNAI), vol. 4696, pp. 52–61. Springer, Heidelberg (2007)
24. Skarmeas, N., Clark, K.L.: Content-based routing as the basis for intra-agent communication. In: Rao, A.S., Singh, M.P., Müller, J.P. (eds.) ATAL 1998. LNCS (LNAI), vol. 1555, pp. 345–362. Springer, Heidelberg (1999)
25. Tarkoma, S., Kangasharju, J.: Optimizing content-based routers: posets and forests. Distributed Computing 19(1), 62–77 (2006)
26. Tummolini, L., Castelfranchi, C., Ricci, A., Viroli, M., Omicini, A.: “exhibitionists” and ”voyeurs” do it better: A shared environment approach for flexible coordination with tacit messages. In: Weyns, D., Van Dyke Parunak, H., Michel, F. (eds.) E4MAS 2004. LNCS (LNAI), vol. 3374, pp. 215–231. Springer, Heidelberg (2005)
27. Weyns, D., Omicini, A., Odell, J.: Environment as a first class abstraction in multiagent systems. Autonomous Agents and Multi-Agent Systems 14(1), 5–30 (2007)
28. Weyns, D., Parunak, H.V.D., Michel, F., Holvoet, T., Ferber, J.: Environments for multiagent systems, state-of-the-art and research challenges. In: Weyns, D., Van Dyke Parunak, H., Michel, F. (eds.) E4MAS 2004. LNCS (LNAI), vol. 3374, pp. 2–52. Springer, Heidelberg (2005)
29. Wong, H.C., Sycara, K.P.: A taxonomy of middle-agents for the internet. In: ICMAS, pp. 465–466. IEEE Computer Society, Los Alamitos (2000)