An Assistance Infrastructure for open MAS

Pablo Almajano
IIIA-CSIC (Spanish Scientific Research Council)
Applied Mathematics Department UB (University of Barcelona)
Barcelona, Spain
palmajano@iiia.csic.es

ABSTRACT

Organisations are an effective mechanism to define the coordination model that structure agent interactions in Open MAS. Execution infrastructures mediate agents interactions while enforcing the rules imposed by the organisation. Although infrastructures usually provide open specifications to agents, understanding this specification and participating in the organisation could result a difficult task to agents, specially when the system is hybrid (i.e participants can be both human and software agents) and its specification becomes more and more complex. In this paper, we formalize an Assistance Infrastructure for Hybrid open MAS that helps agents to pursue their particular goals and, when they are aligned with global goals, lead to a better system's global performance. With this aim, we propose four advanced services – offered by the assistance infrastructure in a distributed way – besides basic organisational information: (1) refined information, (2) justification, (3) advice and (4) estimation. We define two types of assistant agents. A Personal Assistant provides direct and sole support to one agent while a Group Assistant performs those complex processes which affect a group of participants with common services of interest.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems

General Terms

Performance, Management

Keywords

coordination support, organisations and institutions, self-organisation, agents assistance

1. INTRODUCTION

Usually, multi-agent systems (MAS [16]) design and implementation involves the specification of a coordination model and the development of the infrastructure in charge of enacting it. In open MAS, systems are populated by heterogeneous agents trying to achieve particular and/or collective goals. These agents are developed by third parties so the number and kind of agents that may participate in an open MAS is unknown at development time, and varies at runtime [17]. Organisations have proven to be an effective mechanism to define the coordination model that structures agent interactions in open MAS, and infrastructures give support to their execution imposing the rules established by the organisation. In Organisation Centred MAS (OCMAS [13]) approaches, these infrastructures define frameworks where agents with different cognitive abilities may interact. Although OCMAS infrastructures usually provide open specifications to agents [11] [15], understanding this specification and participating in the organisation could result a difficult task to agents, specially as its specification becomes more and more complex. If we take the humans in the loop and consider hybrid systems, where agents may be humans or software agents, the complexity increases and facilitating agent participation becomes a mandatory issue [2] [21].

This paper focuses on the challenge of improving agents' participation in the organisation by means of an *Assistance Infrastructure*. Certain knowledge about the organisation and its environment require complex computational processes in order to be useful for agents (e.g. planning or estimation). Therefore, agents would improve their participation in the organisation if the infrastructure could provide them with some assistance mechanisms that facilitate such processes. Our aim is to help agents in achieving their goals, and, when they are aligned with global goals, lead to a better system's global performance [20].

In this paper we further formalise the set of Agent's *Assistance Services* defined by Campos et al. [7]. We propose *Assistance Services* to offer, in addition to basic organisational information, more complex services such as (1) elaborated *Information*, as an example, specific statistics; (2) *Justification* of the consequences of a performed action; (3) giving *Advice*, for instance providing a plan to achieve agent's goals; and (4) *Estimation* of the consequences of performing an agent's action prior to its execution. The *Assistance Infrastructure* is the framework extension that enables the proposed *Assistance Services* in an open OCMAS. This extension is added to the system as an additional component we call the *Assistance Layer*. Inside this layer, two assistance agents offer the *Assistance Services*: i) *Personal Assistants* and ii) *Group Assistants*. Each *Personal Assistant* provides direct and sole support to one agent in the organisation. Nevertheless, we propose to group agents based on their properties regarding their services of interest

Cite as: An Assistance Infrastructure for open MAS, Pablo Almajano, *Proc. of 10th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2011)*, Tumer, Yolum, Sonenberg and Stone (eds.), May, 2–6, 2011, Taipei, Taiwan, pp. XXX-XXX.

(*Assisted Groups*) and provide the corresponding services to these groups. These services can be provided by a new level of assistants, that we denote as *Group Assistants*. One *Group Assistant* provides support to all the *Personal Assistants* within its *Assisted Group* and therefore *Personal Assistants* computing time and communication overload may be alleviated.

The rest of this document is structured in the following parts. First, section 2 summarizes the related work. Second, section 3 depicts amWater, our motivation example. Next, section 4 provides basic definitions and notation in order to formalise the proposed assistance services. Section 5 describes and formalises the *Assistance Infrastructure*. Finally section 6 gives some conclusions and future work.

2. RELATED WORK

There are two main lines of active research in assistance to MAS provided by *Software Agents*: organisational assistance services [8] [5] [6], and agent assistance services [10] [19] [9]. Regarding organisational assistance, Centeno et al. [8] defined an incentive mechanism (*Incentivators*) which induces individual participants to follow organisational goals by learning their preferences and doing modifications in the environment. On the other hand, Bou et al. [4] defined an Electronic Institution with autonomic capabilities that allows it to adapt its regulations to comply with institutional goals despite varying agent's behaviours (Autonomic Electronic Institutions). In a preceding work, they also applied Case-Based Reasoning (CBR) to reason about the process of adapting the norms of an electronic institution when certain system-wide measures differ from the expected ones [5]. Finally, the Two Level Assisted MAS Architecture (2-LAMA [6]) also provides organisational assistance services. It is composed by two levels. In the domain-level (DL) agents perform domain specific activities. On top of it, a distributed meta-level (ML) is in charge of providing assistance to the DL. This assistance is performed by changing the norms of the organisation and it is provided to groups of not overlapped and fixed clusters of agents. Our approach goes in the line of agent assistance service, so it differs from previous ones in organisational perspectives. Moreover, our concept of groups follows the proposal by Ferber et al. [12] where groups are agent aggregations and one agent may be member of more than one group at time so that groups can freely overlap.

Other works focus on agent assistance services. Electric Elves [10] is a system that applies agent technology in service of the day-to-day activities of the Intelligent Systems Division of the University of Southern California Information Sciences Institute. Chalupsky et al. developed specific Software Personal Assistants (*SPA*) for project activities coordination and external meetings organisation. Since our proposal is general for MAS our *Assistance Layer* can include such kind of services.

These and other proposed *SPA*'s abilities were evaluated in a conceptual framework that simulated human behaviour in different MAS structures [20]. In this research, Okamoto et al. evaluated the impact that *SPA*s have on individual performance and on the collective performance of the organisation as a whole. They built a computational model of human organisations and analysed two types of agent's organisational structures: hierarchical and horizontal. One *SPA* measured ability that is close to our proposal is the decision support (see *Estimation* service in section 5.1.4). They concluded that supporting decision tasks in human organisations increases the success rate (i. e., to meet the deadline with higher probability) and the speed performance average (i. e., to meet the deadline more rapidly), this is particularly the case in organisations with hierarchical structure.

A recent work presented a generic assistant agent framework in which various applications can be built (Military escort planning in peacekeeping and Assistive living applications) [19]. As a proof of concept application, it implemented a coalition planning assistant agent in a peacekeeping problem domain. A more general framework for organising MAS [9] contains *Informative Organisational Mechanisms* as well as *Regulative Organisational Mechanisms* – a generalisation of the *Incentivators* [8] mentioned above –. As mentioned approaches, we also propose a general framework. Moreover, we propose to offer planning in our infrastructure (see section 5.1.3) as in the former. The latter, *Informative Organisational Mechanisms*, is a generalisation of our *Agent's Assistance Services*. Existing OCMAS infrastructures already offer some kind of information about the organisation to a participant. In the $\mathcal{S}-\mathcal{M}oise^{+}$ [15] middleware, the OrgManager provides useful information for the organisational reasoning of agents and its organisational coordination. In this model, an agent is allowed to know another agent information if their roles are linked by an acquaintance relation (defined in the social level). Moreover, the OrgManager also informs actors about the new permissions, obligations, and goals they can pursue when a new state is reached in the organisation. Our framework provides similar information services and also more elaborated ones.

A *Web Service* is a distributed external mechanism that can be used to provide support to agents within a MAS [22]. There are three key features that differentiate *Web Service* (*WS*) and *Software Agent* (*SA*) technologies: i) a *SA* can be integrated within the MAS as regular agents in the system meanwhile a *WS* must be integrated inside a particular agent [3]; ii) a *SA* is able to communicate bidirectionally (ask and be asked) while a *WS* only works under request [18]; and iii) while *WS* only manages external information, a *SA* is able to access the whole system information that the infrastructure makes available to agents. As our proposed services require organisational and environmental knowledge and it is convenient that services are offered by subscription and/or under request, we argue that *SA*s are more suitable than *WS*s to deploy *Assistance Services*.

3. AMWATER: AN EXAMPLE OF ASSISTANCE SCENARIO

Along our formal proposal of the assistance infrastructure we will use as example an electronic market of water rights (*amWater*). *amWater* is a simplification of *mWater* [14] which is an Electronic Institution (EI [1]) focusing on a water market extended with conflict resolution features. EI engine guarantees the correct execution of the system respecting the institutional rules and storing system state. *amWater* implements an electronic market associated to a specific basin.

Many water basins are divided in geographical areas of interconnected water rights, i. e. water transfers between them are possible by using available basin's infrastructures. From now on, we will only refer to this type of water basins.

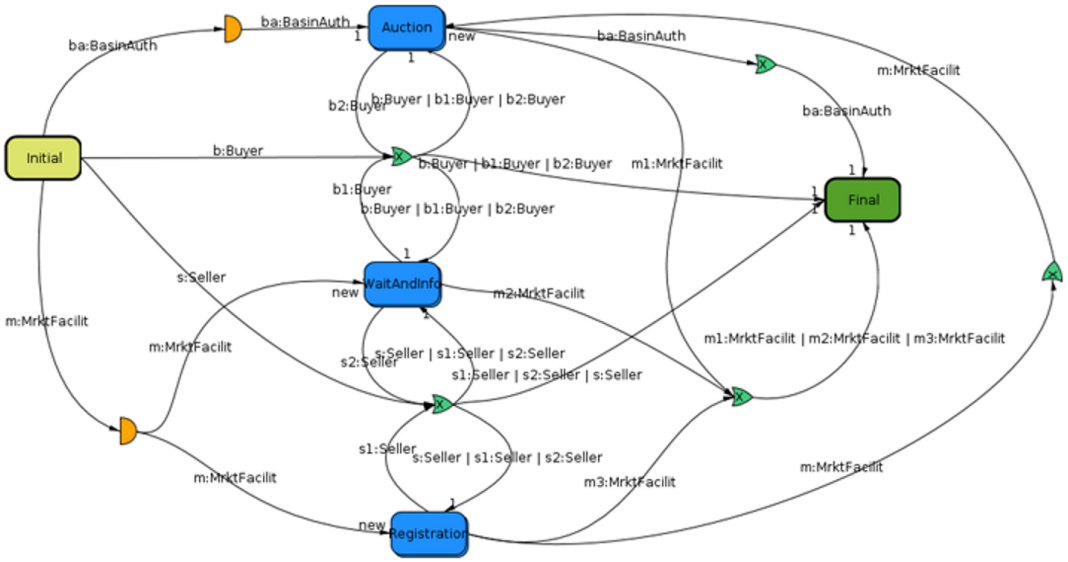

Figure 1: amWater Performative Structure

In our scenario, agents may adopt different roles. Irrigator agents may behave as either buyer or seller subroles while market facilitator and basin authority are staff agents. Figure 1 shows the performative structure [1] of amWater, (i. e. the work-flow along several agent activities). Besides the obligated initial and final scenes, to enter and exit the institution, it has three scenes which enact the market: *Registration* scene, where the market facilitator is in charge of registering sellers' rights; *Waiting and Information* scene, where irrigators can ask for information about auctions to the market facilitator; and *Auction* scene.

The *Auction* scene is where the auction of water rights takes place. We have selected the Japanese Auction protocol because its characteristics seem appropriate for the evaluation of the *Assistance Layer* because its relevant information (starting price, bids and agreements) is public so it can be used by assistance services without any restrictions. There are three roles involved in this scene: buyers are the bidders, the market facilitator is the auctioneer and the basin authority announces the valid agreements. The organisation creates one Auction scene for each area of the basin. Water rights are auctioned in consecutive rounds. Only buyers with owned water rights belonging to the area of the auction are allowed to join it.

The market opens once at the beginning of the irrigation campaign. The agreements reached in the negotiations are valid from the announce of the agreement until the end of the period. The auction protocol is multi-unit Japanese. Basic norms of this protocol are: (1) bidding starts at a low price which goes up in regular increments; (2) to win, a buyer must bid at each price increment; (3) the process ends when i) just one buyer bids at current price (single winner) or ii) no bids are performed, so the winners are the buyers that bid at previous price; and (4) winners request the amount of water they want; Once a round is finished, the basin authority validates the result(s) and announces the agreement(s).

4. BASIC DEFINITIONS

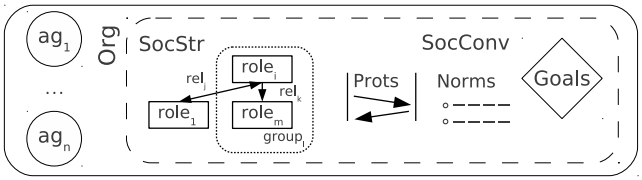

Figure 2: MAS Organisational Model

In this section we provide some basic definitions needed to formalise agent assistance services.

4.1 Organisation

To refer to the *organisation*, we consider the following definition of an organisation model from Campos et al. [6]:

$$Org = \langle SocStr, SocConv, Goals \rangle$$

Figure 2 illustrates its main concepts: a social structure, its social conventions and some organisational goals. We consider open OCMAS and thus we assume agents (Ag) to not belong to the *organisation* itself.

- $SocStr = \langle Rol, Rel \rangle$ is a *social structure* consisting of a set of roles (Rol) and the relationships (Rel) among agents playing certain roles. In amWater, the different roles are irrigator - with buyer and seller subroles -, market facilitator and basin authority.

- $SocConv = \langle Prot, Norms \rangle$ stands for the *social conventions* agents should conform to and expect others to conform to. Social conventions are expressed as a set of interaction *protocols* ($Prot$) and a set of *norms* ($Norms$). In our example scenario, we have three protocols, one for registering rights to transfer, another to ask for next transfers to negotiate and the last one to negotiate transfers, in this case, a Japanese auction protocol. As an example of a norm, when a buyer

wins an auction, the norm is *"the winner is obliged to request for a minimum quantity of water (m^3)"*.

- *Goal* is a set of *goals* that describe the organisation design purpose in terms of desired values for certain *observable properties*. The *observable properties* of the system are environmental and agent properties, denoted respectively by $EnvP$ and AgP. Since we are considering an open MAS, particular agents may pursue individual goals that can differ from organisational ones. In order to achieve their goals, agents may execute a set of possible actions, that we denote by A. In amWater, the main goal is to foster an efficient use of water. As an example, in amWater one $EnvP$ could be the water needs of a cultivation. AgP could be the rights traded by one particular agent. In amWater, some actions may be to enter to one scene and bidding in an auction.

4.2 Organisational Trace

An organisation evolves at run time with the interactions of the agents. As the result of these interactions, both organisation and agent's properties could experiment changes, modifying the organisation's state. In this way, the organisation goes through different execution states at run time. The circumstances that made the organisation evolve may be processed by our proposed *Assistance Layer* in order to offer information about past events (for instance, past auction results or statistics). Similarly, the *Assistance Layer* may analyse agents behaviour in order to help them to interact in the organisation (as example, it could provide plans on what to do in a particular situation to achieve a particular goal).

Therefore, the *Assistance Layer* may provide support taking into account current and past execution states of the organisation. For this purpose, the *Assistance Layer* keeps the different execution states of the organisation (S). The information saved on S has to be complete enough to characterize the state of the organisation and its agents. Moreover, for each step s of the simulation, S only contains non-static information of the organisation. We first define S_s as a tuple:

$$S_s = \langle SocStr_s, SocConv_s, Goals_s, AgP_s, EnvP_s \rangle \quad (1)$$

where:

- $SocStr_s$ are the identifiers of participating agents and their roles at step s.

- $SocConv_s$ are the social conventions. It is composed by $Prot_s$ and $Norms_s$:

$$SocConv_s = \langle Prot_s, Norms_s \rangle \quad (2)$$

  $Prot_s$ contains for each step s, the state of the different interaction protocols and agents involved in step s. For instance, in amWater one possible initial state (S_0) of an execution of the *Auction protocol* in a particular auction scene could be: the auction scene is opened but the auction has not yet started; there are two agents within the scene: one enacting the role of *market facilitator* and one acting as *basin authority*.

  Similarly, $Norms_s$ stands for the execution state of norms at step s. This includes: (1) the set of active norms (when one auction is running, only bidders in the last iteration are allowed to bid), (2) the set of pending obligations as a result of the applied norms (when the auction is paused, the *winner*(s) has to request a water quantity), (3) the list of norm violations (on the auction's pause time-out, the *winner*(s) has not requested any quantity), or (4) the list of applied sanctions due to norm violations (minimum quantity is assigned to the *winner*(s) - because s/he has not requested any quantity -).

- $Goals_s$ represents the degree of fulfilment of each organisational goal and the degree of satisfaction of agents' goals. While the former can be computed by the organisation, the latter can only be gathered by asking the agents.

- AgP_s are the values of the properties of agents, where the property j of agent i ($agP_{i,j}$) is the attribute-value pair: $\langle att_{i,j}, value_{i,j} \rangle$. For example in amWater, the property *quantity* of the k-th water right of a seller ($seller.wrs(k).quantity$) and of the l-th buyer's water right ($buyer.wrs(l)$) change when a *basin authority* validates an agreement on transfer quantity qt of water from $seller.wrs(k)$ to $buyer.wrs(l)$. These two operations are performed atomically:

$$seller.wrs(k).quantity = seller.wrs(k).quantity - qt$$

$$buyer.wrs(l).quantity = buyer.wrs(l).quantity + qt$$

- $EnvP_s$ are relevant values of organisational and/or environmental properties. The list of opened negotiations and the items they are going to negotiate constitute some examples in amWater.

The transitions between execution states of the organisation are mostly driven by actions. We define agents' possible actions (A) at step s (A_s) as :

$$A_s = \{a_1, \ldots, a_n\} \quad (3)$$

where n is the number of agents of the institution and a_i is the action performed by the agent with unique identifier i. We assign the *skip* action to a_i ($a_i = skip$) when agent i did not perform any action.

S and A denote the sequence of different consecutive execution states of the organisation, namely, its trace. In this way, we finally define the *Organisational Trace* ($Trac$) as:

$$Trac = \{(S_0, A_0), \ldots, (S_k, A_k), \ldots, (S_c)\} \quad (4)$$

where S_0 is the initial state, S_c is the current state and in general A_k is the set of actions that take place at state S_k and step k. Scalability and distribution should be taken into account when managing the presented $Trac$, but this is out of scope in this paper which only pretends to formalise it.

As an example in amWater, let imagine that one particular auction scene is opened at current S_c, but the auction has not yet started ($auction.state = stopped$). It just starts when the market facilitator says a message with the content *"start round"*. Hence, current state changes ($S_c = \{auction.state = running\}$) as a consequence of the market facilitator's action $A_c = \{send\_message\ (\ "start\ round"\ )\}$.

5. ASSISTANCE INFRASTRUCTURE

With the aim of assisting –further than enabling– agent coordination, we propose a new set of assistance services that could be incorporated in MAS infrastructures. The services may concern: personal information, only known by the agent; group information, that corresponds to information about agents' groups; and/or global information, information about the whole system.

Next subsections introduce the elements of the *Assistance Infrastructure*.

5.1 Assistance Services

There are three main issues an agent should face to effectively participate in a given organisation: first, the agent only has access to partial information about the actual organisation execution state and its environment; second, the information gathered by the agent should be processed in order to obtain useful knowledge; third, agents have to reason about the coordination model defined by the organisation. Therefore, agents should elaborate plans to achieve their individual goals. On one hand, when programming agents, a developer is required to implement such a planning tasks. On the other hand, fulfilling these tasks by human agents becomes even more complex because of limited speed of human information processing system.

We state that *Assistance services* improve both human and software agents participation in the organisation because: first, they have access to the *Organisational Trace*, $Trac$; second, services are designed to offer complex processes; third, they reason about the coordination model in line with agents' goals.

In addition to services offering basic organisational information, we propose the following services: (1) providing agents with refined *information* to participate in the MAS, (2) *justifying* action consequences or effective constraints, (3) giving *advice* to agents and/or (4) *estimating* action consequences. These services may be provided in a distributed way under request and/or by subscription. Different frequencies of subscriptions could be: each time the information changes, e.g. when a new water right has been registered to transfer; regular frequency, e.g. at the beginning of each period of the irrigation campaign; only when it is interesting for the agent, e.g. the first time entering the organisation the agent receives a "welcome pack"; and never (e.g. subscription disabled).

Next, we describe and formalise these agent assistance services.

5.1.1 Information

This service provides, besides basic organisational information offered by most infrastructures, complex processed information. We can assume that the infrastructure informs agents proactively (equation 5) or upon a request (equation 6). In both cases, we assume in this definition that the infrastructure sends information about the organisation specification (Org) and its execution state ($Trac$):

$$Information :\rightarrow Org \cup Trac \tag{5}$$

$$Information : Req \rightarrow Org \cup Trac \tag{6}$$

where Req is any possible request for information an agent can send. The information requested may be for example: current norms, statistics about specific protocols or environmental property values. The provided or accessible information for an agent depends on its role, properties and its context. For instance, different roles may have access to different information or an agent may only receive information about those interaction protocols he is taking part on.

The system designer can establish different categories of information that may be provided under different circumstances. For example, when an agent joins the organisation it may receive a "welcome pack" containing all the necessary information to participate in the system. The infrastructure could also send messages when an organisation component is modified or when relevant events occur (e.g. an agent leaving the system).

Let us consider the water market scenario implemented in amWater, in order to illustrate the mentioned functionalities. To the irrigator's question *"How many water rights did other irrigators buy?"*, some possible answers could be: *"The market is very active, each irrigator did buy 100 m^3 in average (Average)"*; or *"Here you have the list of (public) validated market transactions (Detail)"*

5.1.2 Justification

This service justifies *the consequences* of actions that agents perform. These consequences depend on the action itself, the current social conventions and the current context. For instance, depending on the enforcement policy, an action can be filtered out or performed with extra consequences –e.g. new obligations–. In both cases we suggest to provide a justification to the agent that performed it. Equation 7 defines this service.

$$Justification : A \times Org \times Trac \rightarrow J \tag{7}$$

where J stands for the set of possible justifications. Hence, we formalise this service as a function that given an agent action, the organisation and its trace, it returns the corresponding justification. The justification provided by the infrastructure may consist on:

- the involved norm(s) (for example, if an auction is running then only buyers can bid);
- the involved norm(s) and current state values (auction is not running);
- or the active involved norm(s) (the norm "buyers can bid" is not active because the auction is not running).

This service may be offered under request and/or by subscription. As an example under request could be: after trying to execute non valid actions then a justification is requested. Another one by subscription may be: each time an agent performs one action that triggers prohibitions.

Finally, the justification may be provided together with an advice: what actions should the agent perform without violating the norms in order to be able to execute the action.

5.1.3 Advice

This service provides agents with a set of alternative plans (P) —i.e. action sequences: $P = \{a_1, \ldots, a_m\}$, where m is the number of actions conforming the plan. The way to create an *advice* can be as simple as indicating what actions other agents have performed on the same situation. Or it

can be as complex as planning possible actions given current state, regulations –e.g. social conventions– and goals. We define two specialisations of advice: Imitation and Planning. It is important to notice, though, that plans can fail due to the actions performed by other agents or to changes happened in the environment. Therefore, more sophisticated plans may be produced by having into account other possible agent's actions, advices given to them and/or environmental changes. But these three options could make really complex the problem to solve. In first attempt, we expect to just use current state and norms as input parameters of the service.

Imitation.

We can provide some plans defined as the most common action, A, carried out by other agents facing the same situation. Note that this service does not require that the agent reveal its goals. An advice in this case could be defined as:

$$Advice : Org \times Trac \rightarrow A \tag{8}$$

In amWater, one buyer may ask *"How much should I pay for the water rights I am interested in?"* and the infrastructure could answer with the advice: *other agents paid (in average) 10 monetary units per* m^3.

Planning.

In order to offer a planning advice, we assume that the agent provides a goal ($IndGoal$) it wants to achieve. This service is defined as:

$$Advice : IndGoal \times Org \times Trac \rightarrow P^N \tag{9}$$

where P^N is a set of alternative plans aimed to achieve goal $IndGoal$, and N is the number of plans conforming the advice. The advice provides plans with the utility of the agent maximised.

For instance, in amWater, the first time an agent enters the institution as a buyer, in the Waiting scene s/he may request for an advice on *"How can I purchase rights?"*. One possible advice may be the following sequence of actions: 1) leave the waiting scene; 2) go to the auction scene number 14; 3) wait until auction starts; 4) bid

This service may be complemented with a *Justification Service*:

$$Advice : IndGoal \times Org \times Trac \rightarrow P^N \cup J^N \tag{10}$$

where J^N are the justifications of providing plans P^N As an example of justification of previous example advised plan could be: Justification: Auction scene number 14 has the lowest starting price.

5.1.4 Estimation

An estimation service predicts the organisational state reached if an agent's indicated action is executed.

$$Estimation : IndA \times Org \times S_c \rightarrow E \tag{11}$$

where E stands for the set of possible estimations. The agent provides an action s/he wants to execute, and the infrastructure uses the organisation definition and the current state (S_c) to estimate its consequences. We provide three different possibilities for the E:

1. *Boolean*. True meaning that the action is valid, while false that it is invalid.

2. $Boolean \times J \times Consequences$. The boolean has the previous meaning, J stands for a justification and *Consequences* stands for the consequences of executing the action. Regarding *Consequences*, one possibility is to assume that it could be a list of obligations the agent will acquire if the action is executed (when the action is valid) or the list of sanctions (when it is invalid).

3. An estimation of the system state after executing the action. For instance, expected values of properties, expected degree of goal achievement, etc.

Estimation may work as a decision support for agents, which can directly focus on evaluating action consequences instead of directly evaluating social conventions. In fact, an agent could use a learning mechanism to acquire empirical knowledge about social conventions instead of analysing their specifications.

For instance, in amWater, some examples of the three different estimations could be:
Q 1: *"Can I enter to the auction scene number 14?"*
E 1: *yes / no.*
Q 2: *"Can I request 40* m^3 *of water?"*
E 2: *Yes, because you won, then you have to pay 400 monetary units / No, because auction is still running.*
Q 3: *"What If I bid at current price?"*
E 3: *You may win / the price will be increased.*

5.2 Assisted Groups

Groups are usually defined in MAS as sets of agent aggregation, where each agent is part of at least one group and it may be member of several groups at the same time so that groups can freely overlap [12]. This definition of MAS' groups is based on organisational activities. Our proposition is in the same line, but based on agent properties related to their services of interest.

Because we are considering open MAS, agents are heterogeneous. Their goals, preferences and interests may vary and are unknown for the system. Agents will be specially interested in assistance services regarding her/his goals. Therefore, agents can be grouped depending on their services of interest. As a consequence, agents in the same group could not participate in the very same activity. We define an *Assisted Group* ($AssGrp$) as:

$$AssGrp = \mathcal{F}(Ag, AgP) \tag{12}$$

$\mathcal{F}(Ag, AgP)$ is a boolean function, that is, it will return true if the agent belongs to the group and false otherwise. Ag is the set of agents and AgP is the set of their properties. The properties and their necessary conditions to be member of a group should be identified at design time with a basic expression language. The operations allowed in this language are: boolean ($\wedge$, $\vee$, $\neg$), comparison ($=$, $\neq$, $<$, $\leq$, $>$, $\geq$), arithmetic ($+$, $-$, $\div$, $\times$), set ($\in$, $\subset$, $\subseteq$) and access variable values ($x.attribute$). Each different definition of group will offer a different partitioning of the agents interacting in the domain.

As previously mentioned in amWater interconnected rights are located within the same area. A general goal of any ir-

rigator entering the institution will be to transfer water between their owned rights and other interconnected rights, by buying or selling them. Interconnected rights are identified by the area they are located in. Therefore, most irrigators will be specially interested in services concerning the area of their rights. As a consequence, they will generally request for such services to the *Assistance Layer*. Then, we group irrigators by the property *areas* that is the set of areas where they own lands. Therefore, function *AssGrp* in this particular domain is defined as:

$$AssGrp\,(area) = \{x \in Ag, x : irrigator | area \in x.areas\}$$

This group subsumes the roles *buyer* and *seller* because they are subroles of *irrigator*. Note that in this definition of groups, one particular *irrigator* belongs to at least one area and s/he may belong to more than one area because s/he could have lands in different areas. Therefore, one agent is part of one or more groups and groups may overlap, as defined in Ferber et al. [12]. Moreover, there may be one *irrigator* in the institution with lands in several areas (so potentially belonging to different groups) that only wants to be assisted on one particular area, so normally s/he only will request support to such area services. As a difference with [12], agents within the same group can be located in a different scene when they are using the same service. Another form of grouping in amWater could be the type of plant cultivated. Irrigators could be interested in historical water transfer's agreements, that is other agents' historical water necessities. They will be specially interested in agreements concerning the plants they currently cultivate in their lands.

5.3 Assistants

We propose two classes of assistants: *Personal Assistants* and *Group Assistants*. We define them in next subsubsections.

5.3.1 Personal Assistants

One Personal Assistant (PAs) may provide direct and sole support to one agent of the open MAS. PAs could observe the organisational trace ($Trac$) so as to deploy the *Assistance Services*.

As depicted in section 5.1, *advice* and *estimation* services could be provided in line with agents' goals, i.e., maximising the agent's utility. As open MAS, agent's goals are information only known by the agent. Therefore, an agent should reveal her/his personal information to its PAs in order to be adequately assisted. We stress that it is the assisted agent's decision which personal information s/he communicates to the PAs. We also stress that the more relevant information revealed, the better *advice* and *estimation* services will be provided.

We propose to establish a private communication channel between the PAs and its assisted agent in order to preserve the privacy of the information in the communications. To ensure the use of private information in the defined terms and conditions, a service contract may be signed between assistants and agents. On one hand, this contract commit the personal assistants to keep personal information private, to not exploit it, and to offer services pursuing participant's revealed goals following social conventions. On the other hand, each participant is responsible for the use s/he makes of the services rendered based on the personal information revealed by her/him and when the personal information should be erased by the assistants. As a PAs is designed to use organisational trace and personal information in a private way, we conclude that agents should feel confidence using them.

5.3.2 Group Assistants

We propose one Group Assistant (GAs) to provide services specialised in one *Assisted Group*. The GAs could implements complex processes using information concerning its *Assisted Group*. This information could be provided by accessing to the organisational trace ($Trac$). Moreover, as discussed above, some services need agent's personal information in order to be performed. We propose that the agent has the decision to reveal its personal information to the GAs, and the GAs should not reveal nor store such information.

We extract the next advantages from introducing a GAs:

- It offers information interesting to the agent, because, as depicted in section 5.2, agents may be grouped by their services of interest.
- GAs could assist to all PAs within its *Assisted Group* simultaneously.
- The information provided by the GAs is anonymous, so agent's privacy policy is preserved.
- GAs may classify $Trac$ information relevant to the group to avoid centralisations, and maintain such information to avoid large size databases.
- PAs computing time and communication overload may be alleviated.

6. CONCLUSIONS AND FUTURE WORK

This paper provides a preliminary formalisation of an Assistance Infrastructure for open OCMAS that helps agents to pursue their individual goals. This is done with the aim of increasing overall system performance, which can only be accomplished if individual agent goals are aligned with global goals. We argue that this requirement is not too demanding if we take into account that agents are free to enter and leave these open systems so the organisation has to somehow guarantee that participant agents will be able to pursue their individual goals (since, otherwise, they would autonomously choose to leave it).

Assistance is provided by means of four different categories of services: information, justification, advice and estimation. Each service category provides specialized assistance devoted to cover individual agent necessities as well as to compensate for agent shortages, both in information processing and/or in computational or reasoning capabilities. The assistance infrastructure is conceived as an extension of current OCMAS infrastructures, and is designed in terms of an Assistance Layer populated by assistant agents. These agents can play two different roles: personal and group assistants. On the one hand, each agent enacting a personal assistant role provides direct and sole support to a single participant agent (i.e., one performing domain activities). On the other hand, an agent enacting the group assistant role performs those complex processes which affect a group

of participants with common services of interest. As intended work, we will complete the formalisation of the infrastructure by further specifying the elements proposed in this paper and providing an architecture that enacts them in an open OCMAS.

In order to illustrate our approach we use an open OCMAS example scenario that implements an electronic market of water rights. Nevertheless the proposed services still require being included in this example. We leave this as future work. For example, following this research objective, we plan to implement the advice by imitation service as a Case Based Planning System that uses *Trac* as training information.

7. ACKNOWLEDGMENTS

This work is partially funded by EVE (TIN2009-14702-C02-01 / TIN2009-14702-C02-02) and AT (CONSOLIDER CSD2007-0022) Spanish research projects, EU-FEDER funds and the Catalan government (Grant 2005-SGR-00093).

8. REFERENCES

[1] J. Arcos, M. Esteva, P. Noriega, J. Rodríguez-Aguilar, and C. Sierra. An integrated development environment for electronic institutions. In *Software Agent-Based Applications, Platforms and Development Kits*, pages 121–142. 2005.

[2] A. Bogdanovych, M. Esteva, S. Simoff, C. Sierra, and H. Berger. A methodology for developing multiagent systems as 3d electronic institutions. volume 4951 of *Lecture Notes in Computer Science*, pages 103–117. Springer Berlin / Heidelberg, 2008.

[3] D. Booth, H. Haas, F. McCabe, E. Newcomer, M. Champion, C. Ferris, and D. Orchard. Web service architecture: W3c working group note, Feb. 2004.

[4] E. Bou, M. López-Sánchez, and J. Rodríguez-Aguilar. Towards self-configuration in autonomic electronic institutions. volume 4386 of *Lecture Notes in Computer Science*, pages 229–244. 2007.

[5] E. Bou, M. López-Sánchez, and J. Rodríguez-Aguilar. *Autonomic Electronic Institutions' Self-Adaptation in Heterogeneous Agent Societies*, volume 5368, pages 18–35. Springer, 2009.

[6] J. Campos, M. Esteva, M. López-Sánchez, J. Morales, and M. Salamó. Organisational adaptation of multi-agent systems in a peer-to-peer scenario. *Computing*, 91:169–215, 2011.

[7] J. Campos, M. López-Sánchez, and M. Esteva. Coordination support in multi-agent systems. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, pages 1301–1302, 2009.

[8] R. Centeno and H. Billhardt. Adaptive regulation of open mas: an incentive mechanism based on online modifications of the environment (extended abstract). In *Proceedings of The 10th International Conference on Autonomous Agents and Multiagent Systems*, pages 1243–1244, 2011.

[9] R. Centeno, H. Billhardt, R. Hermoso, and S. Ossowski. Organising mas: a formal model based on organisational mechanisms. In *Proceedings of the 2009 ACM symposium on Applied Computing*, pages 740–746, 2009.

[10] H. Chalupsky, Y. Gil, C. A. Knoblock, K. Lerman, J. Oh, D. V. Pynadath, T. A. Russ, and M. Tambe. Electric elves: Applying agent technology to support human organizations. In *Proceedings of the Thirteenth Conference on Innovative Applications of Artificial Intelligence Conference*, pages 51–58. AAAI Press, 2001.

[11] M. Esteva. *Electronic institutions. from specification to development*. PhD thesis, Universitat Politecnica de Catalunya, 2003.

[12] J. Ferber and O. Gutknecht. A meta-model for the analysis and design of organizations in multi-agent systems. *Multi-Agent Systems, International Conference on*, pages 128–135, 1998.

[13] J. Ferber, O. Gutknecht, and F. Michel. From agents to organizations: An organizational view of multi-agent systems. In *Agent-Oriented Software Engineering IV*, volume 2935, pages 443–459. 2004.

[14] A. Garrido, A. Giret, and P. Noriega. mwater: a sandbox for agreement technologies. In *Proceeding of the 2009 conference on Artificial Intelligence Research and Development: Proceedings of the 12th International Conference of the Catalan Association for Artificial Intelligence*, pages 252–261, 2009.

[15] J. Hübner, J. Sichman, and O. Boissier. $\mathcal{S} - \mathcal{M}oise^{+}$: A middleware for developing organised multi-agent systems. In *Coordination, Organizations, Institutions, and Norms in Multi-Agent Systems*, volume 3913, pages 64–77. 2006.

[16] N. R. Jennings, K. Sycara, and M. Wooldridge. A roadmap of agent research and development. *Autonomous Agents and Multi-Agent Systems*, 1:7–38, 1998.

[17] I. J. Jureta, M. J. Kollingbaum, S. Faulkner, J. Mylopoulos, and K. Sycara. Requirements-driven contracting for open and norm-regulated multi-agent systems. Technical report, 2007.

[18] K. Mockford. Web services architecture. *BT Technology Journal*, 22:19–26, 2004.

[19] J. Oh, F. Meneguzzi, K. Sycara, and T. J. Norman. Prognostic agent assistance for norm-compliant coalition planning. In *The Second International Workshop on INFRASTRUCTURES AND TOOLS FOR MULTIAGENT SYSTEMS*, pages 126–140, 2011.

[20] S. Okamoto, K. Sycara, and P. Scerri. Personal assistants for human organizations. In V. Dignum, editor, *Organizations in Multi-Agent Systems*. 2009.

[21] T. Trescak, M. Esteva, and I. Rodriguez. Vixee an innovative communication infrastructure for virtual institutions (extended abstract). In *Proceedings of The 10th International Conference on Autonomous Agents and Multiagent Systems*, pages 1131–1132, 2011.

[22] D. Weyns, A. Omicini, and J. Odell. Environment as a first class abstraction in multiagent systems. *Autonomous Agents and Multi-Agent Systems*, 14:5–30, 2007.