Organisational Artifacts and Agents for Open Multi-Agent Organisations: "Giving the Power Back to the Agents"

Rosine Kitio[1], Olivier Boissier[1,⋆], Jomi Fred Hübner[1,2,⋆⋆], and Alessandro Ricci[3]

[1] SMA/G2I/ENSM.SE, 158 Cours Fauriel
42023 Saint-Etienne Cedex, France
{kitio,boissier,hubner}@emse.fr
[2] GIA/DSC/FURB, Braz Wanka, 238
89035-160, Blumenau, Brazil
jomi@inf.furb.br
[3] DEIS, ALMA MATER STUDIORUM Università di Bologna
47023 Cesena (FC), Italy
a.ricci@unibo.it

Abstract. The social and organisational aspects of agency have become nowadays a major focus of interest in the MAS community, and a good amount of theoretical work is available, in terms of formal models and theories. However, the conception and engineering of proper organisational infrastructures embodying such models and theories is still an open issue, in particular when open MAS are considered. Accordingly, in this paper we discuss a model for an organisational infrastructure called ORA4MAS that aims at addressing these issues. By being based on the A&A (Agents and Artifacts) meta-model, the key and novel aspect introduced with ORA4MAS is that organisations and the organisation infrastructure itself are conceived in terms of agents and artifacts, as first-class abstractions giving body to the MAS from design to runtime.

Keywords: Multi-agent Systems, MAS organisations, Open systems.

1 Introduction

Nowadays, current applications of IT show the interweaving of both human and technological *communities* in which software entities act on behalf of users and cooperate with infohabitants, taking into account issues like trust, security, flexibility, adaptation and *openness* [12,24]. As shown in [16], current applications have led to an increase in number of agents, in the duration and repetitiveness of their activities, with a decision and action perimeter still enlarging. Moreover the number of agents' *designers* is also increasing, leading to a huge palette of heterogeneity in these systems. Most designers have doubts about how to put these concepts in practice, i.e., how to program them,

⋆ Partially supported by USP-COFECUB.
⋆⋆ Supported by ANR Project ForTrust.

J.S. Sichman et al. (Eds.): COIN 2007 Workshops, LNAI 4870, pp. 171–186, 2008.

while both addressing the openness and scalability issues and keeping agent's autonomy and decentralisation which are essential features of MAS. The complex system engineering's approach needed to build such applications highlights and stresses requirements on *openness* in terms of ability to take into account several kinds of changes and to adapt the system configuration while it keeps running.

Since it is a huge and complex work to develop systems with this kind of openness, in this paper we propose an organisational infrastructure referred as ORA4MAS which is meant to provide a conceptual and architectural step towards the simplification of this problem. Our proposal is based on the A&A approach [23] where instead of a lot of different components and concepts (e.g., agents, services, proxies, objects, ...), only two types of entities are involved: agents and artifacts. Roughly, while agents model the decisions of the system, the artifacts model its functions. We especially demonstrate this approach showing how the organisational aspect of the MAS can be conceived and designed by only *organisational agents* and *organisational artifacts*. This is in analogy with human organisation and organisation infrastructures, that are populated by humans (as participants and part of the organisation machinery), and by rich sets of artifacts and tools that humans use to support their activities inside the organisation and the organisation itself, encapsulating essential infrastructure services.

In the first part of the paper (Sec. 2), we will have a look at the different approaches that have been developed in the field of multi-agent organisation, stressing what limitations we consider. This is complemented by a look at what has been done in the other dimensions of an MAS, i.e., environment and interaction. Then, we present the basic concepts underlying ORA4MAS infrastructure (Sec. 3), and we briefly describe the shapes of the organisational artifacts devised in ORA4MAS reifying the $\mathcal{M}$OISE$^+$ organisational model (Sec. 4). Finally, we provide concluding remarks and perspectives for the work in (Sec. 5).

2 Background

The recent developments in MAS domain, belonging to what we call *Organisation Oriented Programming (OOP)* [2], have provided many proposals of organisation-oriented middleware. In the different approaches related to OOP, we distinguish two important components: a declarative *Organisation Modelling Language* (OML) and an *Organisation Implementation Architecture* (OIA). The OML *specifies* the organisation(s) of an MAS. It is used to collect and express specific constraints and cooperation patterns imposed on the agents by the designer (or the agents), resulting in an explicit representation that we call *Organisation Specification* (OS). A collective entity, called *Organisation Entity* (OE), instantiates this OS by assigning agents to roles. The OIA will then help these agents to properly "play" their roles as they are specified in the OS.

The OIA normally considers both an agent centered and a system centered point of view.[1] In the former, the focus lies on how to develop different *agent reasoning mechanisms* to interpret and reason on the OS and OE applied to the agents [3,4]. In the latter, the main concern is how to develop an infrastructure, that we call Organisation Infrastructure (OI), that ensures the satisfaction of the organisational constraints

[1] Let's notice that in [28] these points of view are called agent and institutional perspectives.

(e.g., agents playing the right roles, following the specified norms). This second point of view is important in heterogeneous and open systems where the agents that enter into the system may have unknown architectures. Of course, to develop the overall MAS, the former point of view is necessary since the agents probably need to have access to an organisational representation that enable them to reason about it.

The implementation of OI normally follows a common trend in multiagent platforms. These platforms, e.g. JADE [1], have demonstrated the requirement and utility of the notion of "infrastructure" for MAS development [10]. Not only have they supported the implementation of the agents, but are being noticed as a provider of fundamental global generic services going further of only directory facilitator, agent management system or agent communications by also addressing coordination [19]. Therefore, agents related to the application domain operate on top of a middleware layer.

As shown in [2], many implementations of OI follow the general layered architecture depicted in Fig. 1: (i) domain (or application) agents, responsible to achieve organisational goals, use an *organisational proxy* component to interact with the organisation, (ii) the *organisational middleware*, responsible to bind all agents in a coherent OS and OE and provides some services for them, and (iii) communication infrastructure for connecting all components in a distributed and heterogeneous applications. This layered structure results in an engineering approach where the MAS development is considered to be addressed by three kinds of designers: domain or application designers (for the agents and the specification of the OS using the OML), MAS or OI designers (for the organisational infrastructure and OE management), and communication designers.

From the study of the different works considering the OI, we can identify a set of specialised services and proxies (e.g., angels [5], governors [7], managers [14]). In order to stress their ability to manage organisational concepts and to develop dedicated reasoning/processing abilities on the organisation, let's call them *organisational services* (OrgServices). One important point to notice is that all the access to the OI by the agents is mediated by these organisational proxies.

This brief general introduction of OI designs allow us to point out some drawbacks:

1. In some proposals, like $\mathcal{S}$-$\mathcal{M}$OISE$^+$ [14], OrgServices are implemented as agents. The problem is that, conceptually, services are not in the same abstraction level as agents.
2. In the proposals where OrgServices are not agents, whenever an application designer needs to customise some decisions of the system in the organisational dimension (e.g., a sanction system, a reorganisation strategy, the assignment of roles to agents), s/he has to develop/change an OrgService. It can be quite confusing to deal with both OrgServices and agents concepts while developing a system. It will be better to always use the same abstraction level when modelling and implementing the decision aspect of the application.
3. The designer (and the agents) also have to deal with two kinds of environments: a virtual organisational environment (where the agents adopt roles, send messages) and the real environment (where the agents act). An unified view of the environment simplifies the concept of agent interaction.
4. In the general architecture of Fig. 1, the organisation middleware has too much power. Most of the organisational "decisions" are performed at this layer. It is more

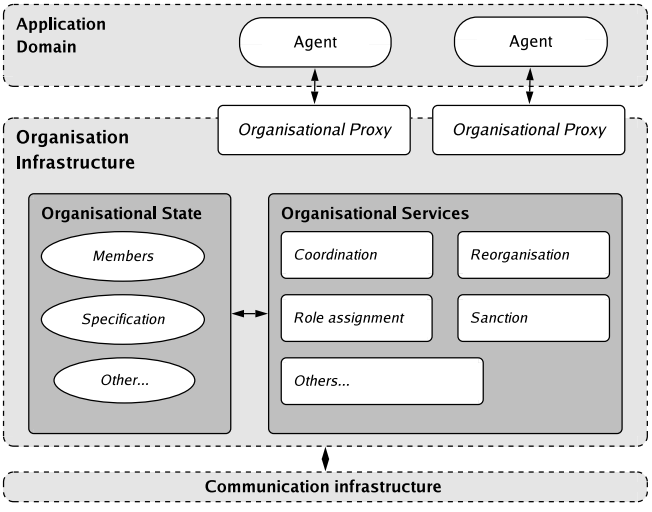

Fig. 1. Common Organisation Infrastructure for open MAS

suitable if the agents make decisions and not the OrgServices. For example, if some agent wants to perform some action or send a message that its organisation does not allow, it can not do it since the middleware (and its organisational proxy) will detect this violation tentative and decide on the sanction to apply. The middleware is thus performing two functions: detection and decision/judgement. In some cases agents operating on the application layer should get their control power back in the sense that they should play some of the roles of the OrgServices. As another example, reorganisation requires that *agents* should be able to manage and access the creation of new organisations.

The problems of existing approaches of organisations are consequence of some properties of the OrgServices design: (i) the enforcement of organisational functions and constraints and (ii) the inclusion of reasoning and decision aspects that can be managed by agents and thus should be in the agent layer.

It's worth noting that the issues stated here do not concern solely the implementation level, but also the conceptual and theoretical level: what is the nature of OrgServices in MAS where only agents are considered as first-class entities?

3 An Organisational Infrastructures Based on Agents and Artifacts

The proposal presented in this paper draws its inspiration from human organisation infrastructures. Human organisation and organisation infrastructures, that are populated by humans (as participants and part of the organisation machinery), and by rich sets of artifacts and tools that humans use to support their activities inside the organisation and the organisation itself, encapsulating essential infrastructure services. According

to psycho-sociological theories and studies such as Activity Theory and Distributed Cognition [17]—recently adopted in computer science fields such as CSCW, HCI and MAS [27,25,22]—the notion of artifact (and tool, taken here as a synonym) plays a key role for the overall sustainability of an organisation and the effectiveness and efficiency of activities taking place inside the organisation.

In particular, some of these artifacts—that we call here *organisational artifacts*—appear to be vital for supporting the coordination of organisation processes and management: for instance by making more effective the communication among the members of an organisation (e.g. the telephone, instant-messaging services, chat-rooms), by providing information useful for orienting the activities of organisation participants (e.g., signs inside a building), by coordinating participants (e.g., queue systems at the post-office), by controlling access to resources and enforcing norms (e.g., the badge used by members in a computer science department to access certains rooms or use some other artifacts, such as copiers). Human societies and organisations continuously improve their experience in designing artifacts more and more effective to support both organisation participation—helping members to cope with the complexity of social activities and work—and organisation management—helping managers to monitor and control the organisation behaviour as a whole.

In the remainder of the section, first we recall the basic ideas provided by the A&A meta-model [23], and then describe how such concepts are exploited to shape the ORA4MAS infrastructure.

3.1 The Notion of Artifacts in MAS

The notion of *MAS environment*, as remarked by recent literatures, has gained a key role in the recent past, becoming a *mediating* entity, functioning as enabler but possibly also as a manager and constrainer of agent actions, perceptions, and interactions (see [29] for comprehensive surveys). According to such a perspective, the environment is not a merely passive source of agent perceptions and target of agent actions—which is, actually, the dominant perspective in agency and in MAS—, but a first-class abstraction that can be suitably designed to *encapsulate* some fundamental functionalities and services, supporting MAS dimensions such as coordination and organisation, besides agent mobility, communications, security, etc.

Among the various approaches, the A&A in particular introduces a notion of *working environment*, representing such a part of the MAS explicitly designed on the one hand by MAS engineers to provide various kinds of functionality—including MAS coordination, organisation—and perceived as first-class entity on the other hand by agents of the MAS [23,20]. A&A working environment are made of *artifacts*, representing function-oriented dynamic entities and tools that agents can create and use to perform their individual and social activities. Among the several sort of artifacts, *coordination artifacts* have been introduced as an important class of artifacts organisations [21], as artifacts mediating agent interactions and encapsulating some kind of coordinating functionality— whiteboards, event services, shared task schedulers are examples. Artifacts can be considered as a complimentary abstraction to agent populating an MAS: while agents are goal-oriented pro-active entities, artifacts are a general abstraction to model function-oriented passive entities, designed by MAS designers to encapsulate

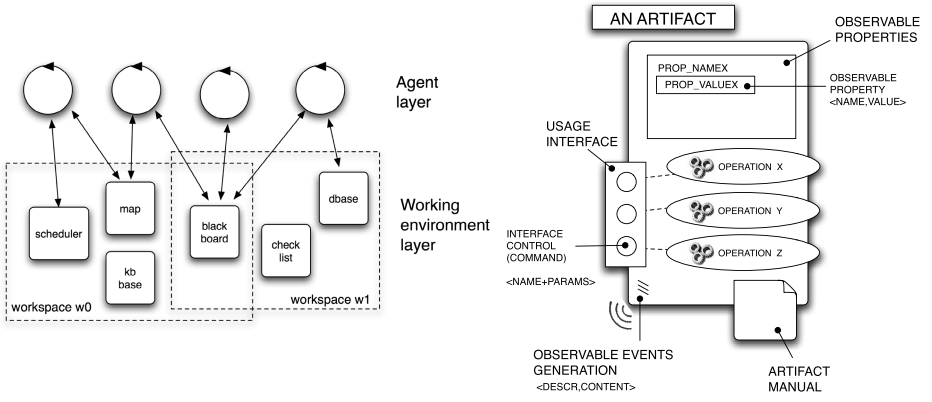

Fig. 2. *(Left)* abstract representation of workspaces, populated by agents—represented by circles—and artifacts—represented by squares. *(Right)* A representation of the main parts and properties of an artifact, with the usage interface, the observable properties and the manual.

some kind of functionality, by representing (or wrapping existing) resources or instruments mediating agent activities. Passive here means that—differently from the agent case—they do not encapsulate any thread of control.

Fig. 2 shows an abstract representation of an artifact as defined in the A&A metamodel, exhibiting analogous parts and properties of artifacts as found in human society. The artifact *function*—and related artifact behaviour—is partitioned in a set of *operations*, which agents can trigger by acting on artifact *usage interface*. The usage interface provides all the controls that make it possible for an agent to interact with an artifact, that is to *use* and *observe* it. Agents can use an artifact by triggering the execution of operations through the usage interface and by perceiving *observable events* generated by the artifact itself, as a result of operation execution and evolution of its state. Besides the controls for triggering the execution of operation, an artifact can have some *observable properties*, i.e., properties whose value is made observable to agents, without necessarily executing operations on it. The interaction between agents and artifacts strictly mimics the way in which humans use their artifacts: let's consider a coffee machine, for a simple but effective analogy. The set of buttons of the coffee machines represents the usage interface, while the displays that are typically used to show the state of the machine represent artifact observable properties. The signals emitted by the coffee machine during its usage represent observable events generated by the artifact.

Analogously to the human case, in A&A each artifact type can be equipped by the artifact programmer with a *manual* composed essentially by the *function description*—as the formal description of the purpose intended by the designer—, the *usage interface description*—as the formal description of artifact usage interface and observable states—, and finally the *operating instructions*—as the formal description of how to properly use the artifact so as to exploit its functionalities. Such a manual is meant to be essential for creating open systems with intelligent agents that dynamically discover and select which kind of artifacts could be useful for their work, and then can use them effectively even if they have not been pre-programmed by MAS programmers for the purpose.

3.2 ORA4MAS Infrastructure

The basic idea in ORA4MAS is to engineer the organisational infrastructure—and the organisations living upon it—in terms of agents and artifacts, following the basic A&A metamodel. Here we use the terms *organisational agents* and *organisational artifacts* to identify those agents and artifacts of the MAS which are part of the organisational infrastructure, and that are responsible of activities and encapsulate functionalities concerning the management and enactment of the organisation. In particular, organisational agents —analogously to managers and administrators in human organisation— are responsible of management activities inside the organisation, concerning observing, monitoring, and reasoning about organisation dynamics, etc. Such activities take place by creating and managing organisational artifacts that are then used by member agents of the organisation. Organisational artifacts are those artifacts that agents of an organisation may want or have to use in order to participate in organisation activities and access to organisation resources, encapsulating organisation rules and functionalities, such as enabling and mediating (ruling) agent interaction, tracing and ruling resource access, and so on.

Even from this abstract characterisation, it is possible to identify some general properties that are of some importance to face the drawbacks listed at the end of Section 2.

Abstraction & encapsulation. By using agents and artifacts to reify both the organisation and the organisation infrastructure—from design to runtime—, we raise the level of abstraction with respect to approaches in which organisation mechanisms are hidden at the implementation level. Such mechanisms become parts of the agent world, suitably encapsulated in proper entities that agents then can inspect, reason and manipulate, by adopting a uniform approach.

Agent autonomy. Agents are still autonomous with respect to decision of using or not a specific artifact—including the organisational artifacts—and keeps its autonomy—in terms of control of its actions—while using organisational artifacts. Agents however can depend on the functionalities provided (encapsulated) by artifacts, which can concern, for instance, some kind of mediation with respect to the other agents co-using the same organisational artifact. Then, by enforcing some kind of mediation policy an artifact can be both an enabler and a *constrainer* of agent interactions. However, such a constraining function can take place without compromising the autonomy of the agents regarding their decisions.

Distributed management. Distributing the management of the organisation into different organisational artifacts installs a distributed coordination (meaning here more particularly synchronisation) of the different functions related to the management of the organisation. Completing this distribution of the coordination, the reasoning and decision processes which are encapsulated in the organisational agents may be also distributed among the different agents. Thanks to their respective autonomy, all the reasoning related to the management of the organisation (monitoring, reorganisation, control) may be decentralized into different loci of decision with a loosely coupled set of agents.

Openness. Organisational artifacts can be created and added dynamically according to the need. They have a proper semantics description of both the functionalities and

operating instructions, so conceptually agents can discover at runtime how to use them in the best way. Related to openness, the approach promotes heterogeneity of agent (societies): artifacts can be used by heterogeneous kinds of agents, with different kinds of reasoning capabilities. Extending the idea to multiple organisations, we can have the same agents playing different roles in different organisations, and then interacting with organisational artifacts belonging to different organisations.

"Power back to agents". The decisions that were embedded in the OrgServices in the OI go back to the agents' layer in organisational agents. In ORA4MAS artifacts encapsulate the coordination and synchronisation which were implemented in OrgServices. Control and judgement procedures are separated from these aspects and are embedded in organisational agents. Organisational agents can then use organisational artifacts to help them in deciding and eventually applying sanctions to other agents.

After sketching the basic concepts underlying the ORA4MAS approach, in next section we finally describe how a full-fledged organisational model can be abstractly implemented on top of agents and artifacts.

4 Shaping ORA4MAS Artifacts Upon $\mathcal{M}$OISE$^+$

$\mathcal{M}$OISE$^+$ (Model of Organisation for multI-agent SystEms) [13] is an OML that explicitly decomposes the organisation into structural, functional, and deontic dimensions. The structural dimension defines the *roles*, *groups*, and *links* of the organisation. The definition of roles states that when an agent decides to play some role in a group, it is accepting some behavioural constraints related to this role. The functional dimension describes how the *global collective goals* should be achieved, i.e., how these goals are decomposed (in global *plans*), grouped in coherent sets (by *missions*) to be distributed to the agents. The decomposition of global goals results in a goal-tree, called *scheme*, where the leafs-goals can by achieved individually by the agents. The deontic dimension is added in order to binds the structural dimension with the functional one by the specification of the roles' *permissions* and *obligations* for missions. Instead of being related to the agents' behaviour space (what they can do), the deontic dimension is related to the agents' autonomy (what they should do).

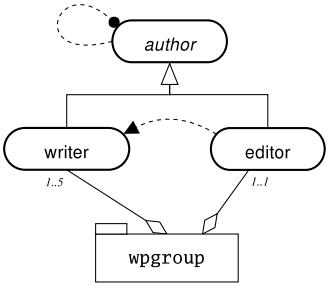

Fig. 3. Structure

As an illustrative and simple example of an organisation specified using $\mathcal{M}$OISE$^+$, we consider a set of agents that wants to write a paper and therefore has an organisational specification to help them to collaborate. The structure of this organisation has only one group (`wpgroup`) with two roles (editor and writer) that are sub-role of the role author. The cardinalities and links of this group are specified, using the $\mathcal{M}$OISE$^+$ notation, in Fig. 3: the group can have from one to five writers and exactly one editor; the editor has authority on writers and every author (and by inheritance every writer

and editor) has a communication link to all other authors. In this example, the editor and the author roles are not compatible, to be compatible a compatibility relation must be explicitly added in the specification.

To coordinate the achievement of the goal of writing a paper, a scheme is defined in the functional specification of the organisation (Fig. 4). In this scheme, an agent initially defines a draft version of the paper (identified by the goal *fdv* in the scheme of Fig. 4) that has the following sub-goals: write a title, an abstract, the introduction, and the section names. Other agents then "fill" the paper's sections to get a submission version of the paper (identified by the goal *sv*). The goals of this scheme are distributed in three missions: $mMan$ (general managing of the process), $mCol$ (collaborate in the paper writing the content), and $mBib$ (get the references for the paper). A mission defines all goals an agent commits to when participating in the execution of a scheme, for example, commit to the mission $mMan$ is indeed a commitment to achieve six goals of the scheme. The deontic relation from roles to missions is specified in Fig. 5. For example, any agent playing the role editor is permitted to commit to the mission $mMan$. The structural, functional, and deontic specifications briefly described here form an Organisational Specification (OS) where, for example, some agents can "instantiate" an Organisational Entity (OE).

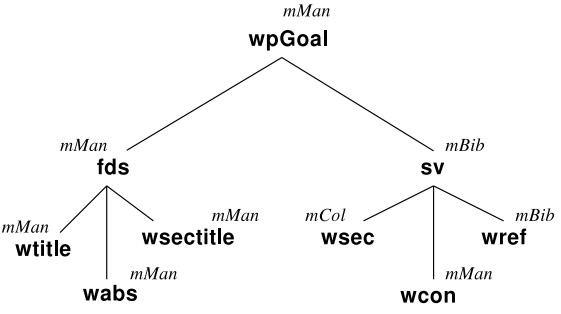

Fig. 4. Functioning

role	deontic relation	mission	cardinality
editor	*permission*	$mMan$	1..1
writer	*obligation*	$mCol$	1..5
writer	*obligation*	$mBib$	1..1

Fig. 5. Deontic relations

Organisational Agents and Artifacts Based on $\mathcal{M}$OISE$^+$. We exploit here the $\mathcal{M}$OISE$^+$ model to identify and shape a basic set of organisational artifacts (kind) and agents that constitute the basic infrastructure building blocks of ORA4MAS, being a sort of "reification" of the structural specification (SS), functional specification (FS), and deontic specification (DS) (Fig. 6). This basic set accounts for: an OrgBoard artifact —used to keep track of the structure of organisation in the overall; a GroupBoard artifact —used to manage the life-cycle of a specific group; a SchemeBoard type — used to support and manage the execution of a social scheme. Here we consider just a core set, skipping most details that would make heavy the overall understanding of the approach: the interested reader is forwarded on this technical report [15] to get further details.

In the following we briefly describe the basic characteristics of these kinds of artifact. In the description, the operations (commands) enlisted in artifact usage interface are

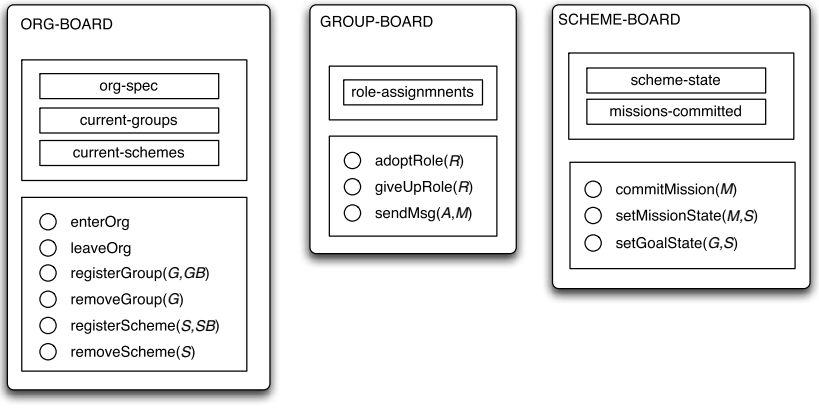

Fig. 6. Basic kinds of artifacts in ORA4MAS, with their usage interface, including operations and observable properties

abstractly described by a name with input parameters, followed (optionally) by a set of the observable events possibly generated by the operation execution (only events significant for artifact specific functionalities are considered, skipping those generated by default by the artifact). Observable properties are represented just by a name, which corresponds to the name of the property.

A simple abstract model for the OrgBoard artifact is depicted in Fig. 6 (left). The usage interface is composed by operations to:

- Register / de-register a new group: registerGroup(G,GB), removeGroup(G)—where G is an identifier for a group and GB is the identifier of the related group board artifact;
- Register / de-register a new scheme: registerScheme(S,SB), removeScheme(S) where S is the identifier for a schema and SB is the identifier of the scheme board.

Among the observable properties: list of current groups; list of current schemes; and the organisation specification (including SS, FS, DS).Generally speaking, the observable properties of the artifact make it possible —for agents observing an OrgBoard—to monitor and be aware of which are the schemes and groups created. Also, this artifact can be inspected to know which are the SS, FS, DS currently adopted in the organisation.

The GroupBoard artifact type (see Fig. 6, center) is instantiated upon a specific SS, and provides functionalities to manage a group in terms of set of available roles and agents participation, according to the specific structure and strategy specified in the SS. The usage interface accounts for the following operations:

- Adopt a new role: adoptRole(R):{role_adoption_ok,role_adoption_failed}, where R is the identifier for a role;
- Give up a role: giveUpRole(R):{role_giveup_ok,role_giveup_failed};

– Sending a message to a specific agent or all the agents part of the group: sendMsg(A,M), sendMsg(M), where A is the identifier for the receiver agent, m is the message content.

Among the observable properties, we have only the role assignments. By observing a GroupBoard artifact, an agent can thus monitor and be aware of the role-agent assignments inside the group.

The GroupBoard interprets the structural specification and maintains a consistent state of the group so that some important organisational constraints are not violated — the remaining constraints are enforced by organisational agents. For instance, when some agent asks for a role adoption in the group, the GroupBoard ensures that: (1) the role belongs to its group specification; (2) the number of players is lesser or equals than the maximum number of players defined in the group's compositional specification; (3) each role ρ_i that the agent already plays is specified as compatible with the new role.

The SchemeBoard artifact type (see Fig. 6, right) is instantiated upon a specific FS and DS, and provides functionalities to manage the execution of a social scheme, coordinating the commitments to missions and the achievement of goals. It is essentially a coordination artifact, automating the management of the dependencies between the missions and the goals as described by the social scheme, and embedding such part of the deontic specification concerning permissions and obligations for agents to commit to missions. The usage interface provides commands to:

– Commit to a mission: commitMission(M):{commit_ok, commit_failed}, where M is the identifier for a mission;
– Set mission state: setMissionState(M,S), where M is the identifier for a mission and S can be either *completed* or *failed*;
– Set goal state: setGoalState(G,S), where G is the identifier for a goal and S can be either *satisfied* or *impossible*.

Among the observable properties, we have: the scheme dynamic state, that includes all the goals of the scheme and their state; the list of the current missions committed. By observing a SchemeBoard artifact, an agent can monitor then the overall dynamics concerning the scheme execution, and the be aware of which missions are assigned to which agents, which goals are achieved and which can be pursued.

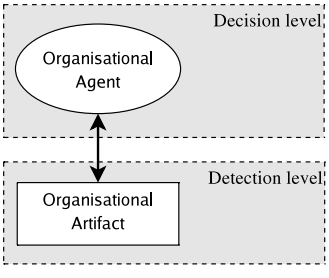

Fig. 7. Agent & Artifact

Organisational Agents. The organisational agents are essentially managers responsible to create and manage the organisational artifacts described previously (Fig. 7). Such activities typically include observing artifacts dynamics and possibly intervening, by changing / adapting artifacts or interacting directly with agents, so as to improve the overall (or specific) organisation processes or taking some kinds of decisions when detecting violations. As an example, one or multiple *scheme managers* agents can be introduced, responsible of monitoring the dynamics of the execution of a scheme by observing a specific SchemeBoard instance. The SchemeBoard artifact and

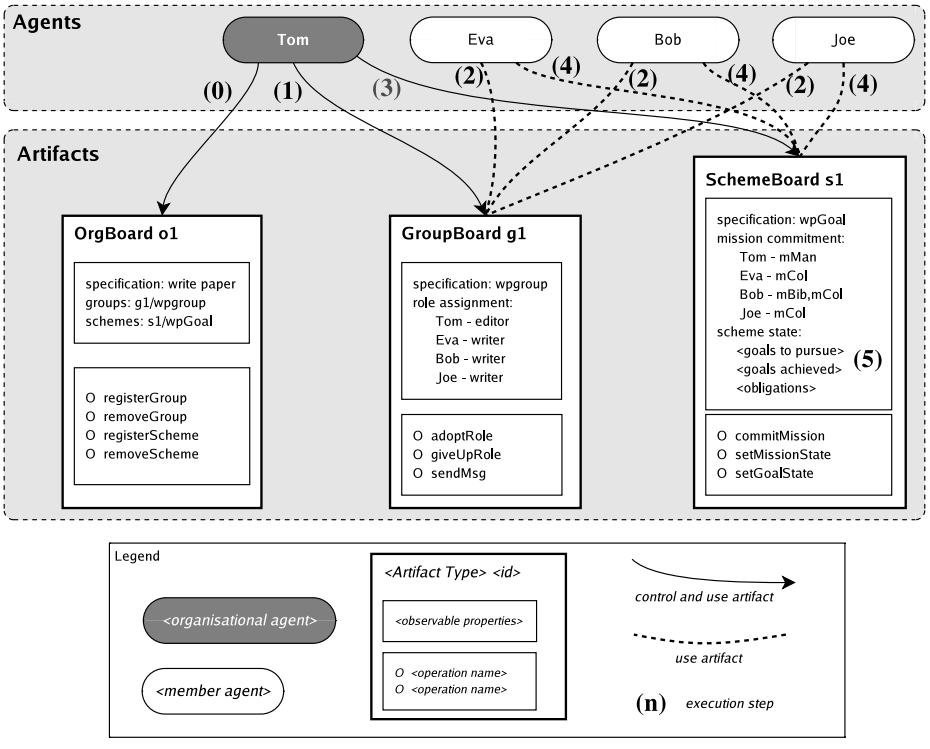

Fig. 8. Example of the construction and use of artifacts

scheme manager agents are designed so as that the artifact allows for violation of the deontic rules concerning the commitment of missions by agents playing some specific roles, and then the decision about what action to take—after detecting the violation—can be in charge of the manager agent.

The Writing Paper Example. We consider that four agents (Tom, Eva, Joe, and Bob) want to write a paper together using the proposed architecture described above. Among these agents, Tom is also an organisational agent and thus it may create the OrgBoard for the system. The creation of the OrgBoard is based on an organisational specification where the roles, groups, schemes, etc. are defined as presented in the Sec. 4. The following steps show how this system evolves until their goal of writing a paper is achieved (depicted in Fig. 8):

1. Tom creates the GroupBoard based on the specification of Fig. 3. Tom then register this new group in the OrgBoard using the registerGroup operation of the OrgBoard artifact. To succeed in this registering, the new group should satisfy all constraints defined in the OS (the group cardinality, for instance).
2. The new GroupBoard artifact is then perceived by all agents. While Tom decides to adopt the role editor, Eva, Bob, and Joe decide to adopt the role writer in this group.

To adopt the role, they use the adoptRole operation of the GroupBoard. Again, this operation may fail in case the agents do not fit in the requirements for the role (cardinality of the role in the group, compatibility of roles, etc.). The reasons for this role adoption is not covered here, but, for example, they may decide to become a writer because Tom has invited them to enter into the group.

3. Tom creates the SchemeBoard to start the process of writing the paper and then registers it in the OrgBoard using the registerScheme operation. This artifact interprets the specification of Fig. 4.
4. Once the scheme is created some obligations are activated and perceived by the agents in the SchemeBoard. For instance, the agent Bob, that is playing the role writer, is obligated to commit to the missions $mCol$ and $mBib$ (cf. Fig. 5) and thus he decided to commit to both. Since the mission $mBib$ has a cardinality constraint that set the maximum number of commitments to one, the other agents are not obligated to this mission anymore. They commit thus only to the mission $mCol$. Tom, that plays editor, commits to $mMan$. We are assuming here that the agents are obedient and always commit to their obligations and pursue their organisational goals.
5. Having their missions, the agents can pursue the goals of the scheme. Initially only the goals *wp*, *fds*, and *wtitle* can be pursued. These goals belongs to the $mMan$ mission, so only Tom has something to do, the others will wait him to achieve these goals. To know which goal can be pursued and to set them as achieved, the agents perceive the SchemeBoard and act on it using the setGoalState operation. The SchemeBoard works therefore like a coordinating artifact.

In this example, the SchemeBoard simply *shows* the obligations for each agent and which goals they should pursue. As an artifact, it is maintaining the current state of the scheme execution. However, since it is not an agent, in case some agent does not commit to a mission it should do or does not achieve some goal, the SchemeBoard does nothing. An organisational manager agent, like Tom, must perceive this artifact and decide what to do when some violation occurs.

Towards a Concrete Architecture. ORA4MAS concrete architecture is realised on top of CARTAGO infrastructure, embedding algorithms used in $\mathcal{S}$-$\mathcal{M}$OISE$^+$. CARTAGO (Common ARtifact Infrastructure for AGent Open environment) is an MAS infrastructure based on the A&A meta-model, providing the capability to define new artifacts types, suitable API for agents to work with artifacts and workspaces, and a runtime supporting the existence and dynamic management of working environments. CARTAGO is meant to be integrated with existing cognitive MAS architectures and models / languages / platforms, so as to extend them to create and work with artifact-based environments. A first example of integration with the *Jason* agent programming platform is briefly described in [23]. CARTAGO is available as open-source projects freely downloadable from the project web sites (`http://www.alice.unibo.it/cartago`). The engineering of the first prototype of the ORA4MAS infrastructure upon CARTAGO is still a work in progress.

5 Conclusion and Perspectives

In this paper, we have followed the A&A approach to give back the power to agents in an organisational approach. From this perspective, we have defined on the one hand the organisational artifacts which encapsulate the functional aspects of an organisation and organisation management, and on the other hand the organisational agents, which encapsulated the decision and reasoning side of the management of organisations.

Although we already have some initial results of the ORA4MAS project, as those presented in this paper, we had concretely evaluated the proposal for only one OML (the $\mathcal{M}$OISE$^+$). The first future work of the project will therefore be an evaluation of its application for different OMLs such as ISLANDER [6], OMNI [5], $\mathcal{M}$OISE^{Inst} [11], or AGR [8]. Following this broadest application we can then better compare our approach with related works (e.g. [9]) and even others such as those managing the organisation with communication acts (e.g. RICA-J [26]) or exploiting the environment to coordinate and constrain the agents' behaviour [30].

Other extensions aim at taking benefit of the uniform concepts used to implement the environment and the organisation abstractions through the concept of artifacts. Such an homogeneous conceptual point of view will certainly help us to bind both concepts together in order to situate organisations in environment or to install the access to the environment into organisational models (in the same direction as proposed in [18]). Other points of investigation are (1) the study of the reorganisation process of an MAS using the ORA4MAS approach, (2) the impact of the reorganisation on the organisational artifacts, and (3) the definition of a meta-organisation for the ORA4MAS, so that we have special roles for organisational agents that give them access to the organisational artifacts.

References

1. Bellifemine, F.L., Caire, G., Greenwood, D.: Developing Multi-Agent Systems with JADE. Wiley, Chichester (2007)
2. Boissier, O., Hübner, J.F., Sichman, J.S.: Organization oriented programming from closed to open organizations. In: O'Hare, G.M.P., Ricci, A., O'Grady, M.J., Dikenelli, O. (eds.) ESAW 2006. LNCS (LNAI), vol. 4457, pp. 86–105. Springer, Heidelberg (2007)
3. Broersen, J., Dastani, M., Hulstijn, J., Huang, Z., der van Torre, L.: The BOID architecture: conflicts between beliefs, obligations, intentions and desires. In: Müller, J.P., Andre, E., Sen, S., Frasson, C. (eds.) Proceedings of the Fifth International Conference on Autonomous Agents, Montreal, Canada, pp. 9–16. ACM Press, New York (2001)
4. Castelfranchi, C., Dignum, F., Jonker, C.M., Treur, J.: Deliberate normative agents: Principles and architecture. In: Jennings, N.R. (ed.) ATAL 1999. LNCS, vol. 1757, Springer, Heidelberg (2000)
5. Dignum, V., Vazquez-Salceda, J., Dignum, F.: OMNI: Introducing social structure, norms and ontologies into agent organizations. In: Bordini, R.H., Dastani, M., Dix, J., El Fallah-Seghrouchni, A. (eds.) PROMAS 2004. LNCS (LNAI), vol. 3346, pp. 181–198. Springer, Berlin (2005)
6. Esteva, M., Rodriguez-Aguiar, J.A., Sierra, C., Garcia, P., Arcos, J.L.: On the formal specification of electronic institutions. In: Dignum, F., Sierra, C. (eds.) Proceedings of the Agent-mediated Electronic Commerce. LNCS (LNAI), vol. 1191, pp. 126–147. Springer, Berlin (2001)

7. Esteva, M., Rodríguez-Aguilar, J.A., Rosell, B., AMELI, J.L.: An agent-based middleware for electronic institutions. In: Jennings, N.R., Sierra, C., Sonenberg, L., Tambe, M. (eds.) Proceedings of the Third International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2004), pp. 236–243. ACM Press, New York (2004)
8. Ferber, J., Gutknecht, O.: A meta-model for the analysis and design of organizations in multi-agents systems. In: Demazeau, Y. (ed.) Proceedings of the 3rd International Conference on Multi-Agent Systems (ICMAS 1998), pp. 128–135. IEEE Press, Los Alamitos (1998)
9. García-Camino, A., Rodríguez-Aguilar, J., Vasconcelos, W.W.: A distributed architecture for norm management in multi-agent systems. In: Sichman, J.S., et al. (eds.) COIN 2007 Workshops. LNCS (LNAI), vol. 4870, pp. 171–186. Springer, Heidelberg (2008)
10. Gasser, L.: Mas infrastructure: Definitions, needs and prospects. In: Revised Papers from the International Workshop on Infrastructure for Multi-Agent Systems, London, UK, pp. 1–11. Springer, Heidelberg (2001)
11. Gâteau, B., Boissier, O., Khadraoui, D., Dubois, E.: Moiseinst: An organizational model for specifying rights and duties of autonomous agents. In: Third European Workshop on Multi-Agent Systems (EUMAS 2005), Brussels, Belgium, December 7–8, pp. 484–485 (2005)
12. IST Advisory Group: Ambient intelligence: From vision to reality. Technical report, IST (2003), `ftp://ftp.cordis.europa.eu/pub/ist/docs/istag-ist2003_consolidated_report.pdf`
13. Hübner, J.F., Sichman, J.S., Boissier, O.: A model for the structural, functional, and deontic specification of organizations in multiagent systems. In: Bittencourt, G., Ramalho, G.L. (eds.) SBIA 2002. LNCS (LNAI), vol. 2507, pp. 118–128. Springer, Berlin (2002)
14. Hübner, J.F., Sichman, J.S., Boissier, O.: $\mathcal{S}$-$\mathcal{MOISE}^+$: A middleware for developing organised multi-agent systems. In: Boissier, O., Padget, J.A., Dignum, V., Lindemann, G., Matson, E., Ossowski, S., Sichman, J.S., Vázquez-Salceda, J. (eds.) ANIREM 2005 and OOOP 2005. LNCS (LNAI), vol. 3913, pp. 64–78. Springer, Heidelberg (2006)
15. Kitio, R.: Organizational artifacts and agents for open multi-agent systems, Master Thesis report (June, 2007), `http://www.emse.fr/~boissier/kitio`
16. Luck, M., et al.: Agent Technology: Computing as Interaction (A Roadmap for Agent Based Computing). AgentLink (2005)
17. Nardi, B.A.: Context and Consciousness: Activity Theory and Human-Computer Interaction. MIT Press, Cambridge (1996)
18. Okuyama, F.Y., Bordini, R.H., da Rocha Costa, A.C.: Spatially distributed normative objects. In: G. Boella, O. Boissier, E. Matson, J. Vázquez-Salceda (eds.) Proceedings of the Workshop on Coordination, Organization, Institutions and Norms in Agent Systems (COIN), held with ECAI 2006, Riva del Garda, Italy (August 28, 2006)
19. Omicini, A., Ossowski, S., Ricci, A.: Coordination infrastructures in the engineering of multiagent systems. In: Bergenti, F., Gleizes, M.-P., Zambonelli, F. (eds.) Methodologies and Software Engineering for Agent Systems: The Agent-Oriented Software Engineering Handbook. Multiagent Systems, Artificial Societies, and Simulated Organizations, vol. 11, pp. 273–296. Kluwer Academic Publishers, Dordrecht (2004)
20. Omicini, A., Ricci, A., Viroli, M.: Agens Faber: Toward a theory of artefacts for MAS. Electronic Notes in Theoretical Computer Sciences 150(3), 21–36 (2006)
21. Omicini, A., et al.: Coordination artifacts: Environment-based coordination for intelligent agents. In: AAMAS 2004, vol. 1, pp. 286–293. ACM, New York (2004)
22. Ricci, A., Omicini, A., Denti, E.: Activity Theory as a framework for MAS coordination. In: Petta, P., Tolksdorf, R., Zambonelli, F. (eds.) ESAW 2002. LNCS (LNAI), vol. 2577, pp. 96–110. Springer, Heidelberg (2003)

23. Ricci, A., Viroli, M., Omicini, A.: A general purpose programming model & technology for developing working environments in MAS. In: Dastani, M., et al. (eds.) 5th International Workshop "Programming Multi-Agent Systems" (PROMAS 2007), AAMAS 2007, Honolulu, Hawaii, USA, pp. 54–69 (May 15, 2007)
24. Sairamesh, J., Lee, A., Anania, L.: Introduction. Commun. ACM 47(2), 28–31 (2004)
25. Schmidt, K., Simone, C.: Coordination mechanisms: Towards a conceptual foundation of CSCW systems design. International Journal of Computer Supported Cooperative Work (CSCW) 5(2–3), 155–200 (1996)
26. Serrano, J.M., Ossowski, S.: A compositional framework for the specification of interaction protocols in multi-agent organizations. In: Proceedings of the Third European Workshop on Multi-Agent Systems (EUMAS), Brussels, Belgium, pp. 375–386 (December 7–8, 2005)
27. Susi, T., Ziemke, T.: Social cognition, artefacts, and stigmergy: A comparative analysis of theoretical frameworks for the understanding of artefact-mediated collaborative activity. Cognitive Systems Research 2(4), 273–290 (2001)
28. Vázquez-Salceda, J., Aldewereld, H., Dignum, F.: Norms in multiagent systems: Some implementation guidelines. In: Proceedings of the Second European Workshop on Multi-Agent Systems (EUMAS 2004) (2004)
29. Weyns, D., Van Dyke Parunak, H.: Journal of Autonomous Agents and Multi-Agent Systems. Special Issue: Environment for Multi-Agent Systems, vol. 14(1). Springer, Netherlands (2007)
30. Weyns, D., Van Dyke Parunak, H., Michel, F. (eds.): E4MAS 2005. LNCS (LNAI), vol. 3830, pp. 1–17. Springer, Heidelberg (2006)