Normative Multi-Agent Organizations: Modeling, Support and Control. Draft Version

Benjamin Gâteau[1],[2] and Olivier Boissier[2]

[1] CITI/CRP Henri Tudor
29 Av. John F. Kennedy L-1855 Luxembourg – G.-D. of LUXEMBOURG
{benjamin.gateau}@tudor.lu
[2] SMA/G2I/ENSM Saint-Etienne
158, Cours Fauriel F-42023 Saint-Etienne Cedex 02 – FRANCE
olivier.boissier@emse.fr

Abstract. In the last years, social and organizational aspects of agency have become a major issue in multi-agent systems' research. Recent applications of MAS enforce the need of using these aspects in order to ensure some social order within these systems. Tools to control and regulate the overall functioning of the system are needed in order to enforce global laws on the autonomous agents operating in it. This paper presents a normative organization system composed of a normative organization modeling language $\mathcal{M}$OISE^{Inst} used to define the normative organization of a MAS, accompanied with $\mathcal{S}$YNAI, a normative organization implementation architecture which is itself regulated with an explicit normative organization specification.

1 Introduction

Nowadays, current IT applications show the large scale interweaving of human and technological communities (e.g. Web Intelligence, Ambient Intelligence, Interactive TV). Using Multi-Agent System (MAS) technology introduces software entities that act on behalf of users and cooperate with those infohabitants. 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 [1]. As stated in [2], "Openness without control may lead to chaotic behavior". Being composed of heterogeneous and autonomous agents, tools to control and regulate the overall functioning of the system are required in order to enforce global laws on the autonomous agents operating in the system.

In this paper we present a multi-agent normative organization environment composed of $\mathcal{S}$YNAI, multiagent organization infrastructure, interpreting normative declarative organizations programmed with $\mathcal{M}$OISE^{Inst}, a normative organization modeling language. $\mathcal{M}$OISE^{Inst} is an extension of the $\mathcal{M}$OISE^{+} developped by [3]. $\mathcal{S}$YNAI is composed of generic *supervisor* agents, aiming at controlling and enforcing the rights and duties of autonomous "domain" agents operating in an normative organization expressed with $\mathcal{M}$OISE^{Inst} ($\mathcal{M}$OISE^{Inst} extending $\mathcal{M}$OISE^{+}, $\mathcal{S}$YNAI extends

Dagstuhl Seminar Proceedings 07122
Normative Multi-agent Systems
http://drops.dagstuhl.de/opus/volltexte/2007/902

$\mathcal{S}$-$\mathcal{M}$OISE$^+$ [4]). Whereas supervisor agents are dedicated to the control of the system, the domain agents implement the functionalities of the application. $\mathcal{M}$OISE^{Inst} is also used at a meta-level since the supervisor agents themselves operate under the control of a normative organization that structures and constrains their control behaviours on the domain agents. All along the paper, we illustrate the use of this environment with an iTV game issued from the European ITEA Jules Verne Project.

This paper is organised as follows: section 2 gives a global overview of our framework for defining normative multi-agent organizations. Its use is illustrated with an iTV application. The succeeding sections presents the organization modeling language $\mathcal{M}$OISE^{Inst} and then the infrastructure supporting it, $\mathcal{S}$YNAI. Finally, before concluding, section 5 positions our work with respect to other approaches.

2 Global view

In the recent past, multiagent technologies have been developed and deployed in different applications. Most of these efforts have been largely supported by the existence of multiagent platforms like JADE [5] or FIPA-OS [6]. These platforms have demonstrated the needs and utility of generic services for supporting the execution of multiagent applications like for instance Agent Management System, Directory Facilitator. The recent developments in the domain (e.g. electronic commerce [7]) have shown the requirement to enrich those services to provide multiagent applications with what we call organization oriented programming [8]. Such an approach provides the possibility to express and make explicit one or more patterns of cooperation installed in a top-down approach on the agents, that constrains and drive their actions and interactions towards some purpose. Current multiagent approaches on normative organizations [9] propose to enrich those patterns of cooperation with the explicit modelling of rules stating the norms directing the functioning of the system. Agents interpret these norms and are enforced to comply with their specified behaviours. However, agents can still practise organizational autonomy, in the sense that they are able to read, to represent, and to reason about the organization and may decide whether to follow the constraints stated by the organization or not. They may also decide to adapt and change the organization in a bottom-up process, installing a new pattern/structure. Such a functioning corresponds to the combination of agent-centred organized MAS and organization oriented MAS approaches [8]

Considering the programming of normative organizations has led us to introduce norms in the $\mathcal{M}$OISE^{Inst} *organization modeling language* (OML) used to *define* the organization(s) of an MAS. It is used to collect and express specific constraints and cooperation patterns that the designer (or the agents) have in mind, resulting in an explicit representation that we call *organization specification* (OS). Finally the OS is executed and interpreted on an *Organization Implementation Architecture* (OIA) to install a collective entity in the MAS that we call *Organization Entity* (OE): a set of agents building the organization specified with an OS. Once created, the OE's history starts and runs by events like other agents entering and/or leaving it, group creation, role adoption, goal commitment, etc. The OIA may be further split into an agent part (such as, for instance, in [10]) and into an organization infrastructure part, the $\mathcal{S}$YNAI system. Implied by the

introduction of the normative dimension in the OML, $\mathcal{S}$YNAI has been enriched with different mechanisms to deal with it.

Let's illustrate this sketch of a normative system with our Interactive Games application (see Fig. 1): a "questions – answers" TV game show opposing a real players' team present on the TV scene, to a televiewers' team interacting from home into the game with the help of avatars, i.e. the domain agents. Each avatar is under the control of its respective televiewer. The quizmaster is also supported by a virtual assistant, aiming at regulating the game. As in all collective games, the aim is to promote a collective behaviour among the players of a same team. The OS defined with the $\mathcal{M}$OISE^{Inst} organization modeling language states the structure and functioning of the game, with a set of *norms* defining the game rules, the sanctions and rewards in use during the game. However, since avatars are autonomous agents, they can be autonomous with respect to these constraints, e.g. a televiewer is able to decide to answer whereas it is not his turn and to take the risk to be punished. The OIA has been defined with $\mathcal{S}$YNAI as a *normative system* in order to control, regulate and reward/punish avatars when they respect or not the OS. Supervisor agents of the OIA are dedicated to the management of the organization and to the enforcement of the game rules on the avatars. Both kinds of agents (supervisor and domain) are organised and constrained according to the OS defined with the $\mathcal{M}$OISE^{Inst} normative OML [11]. Agents are thus able to reason on the organization and constraints. They have the possibility to decide to take it into account or not. The OIA reads this specification in order to supervise and control the agents as well as to be informed about its own organization specification.

3 Normative Organization Modeling Language

$\mathcal{M}$OISE^{Inst} [11] is used to define what we call an organization specification (OS) with the help of four dimensions[1]: structural specification (SS), functional specification (FS), contextual specification (CS) and normative specification (NS).

3.1 Structural specification

The *structural specification* (SS) defines the MAS structure with the notions of *roles*, *groups* and *links*. A *role* consists in a label to which constraints on the playing agents' behavior. Roles are also used as anchors to the links. A *group specification* consists in a set of links and roles. The Fig. 2 shows the structural specification of the iTV application: a "Team" group is composed of the roles corresponding to the expertises mobilized for the game ("History", "Geo", "Sport", "Science") with a special role "Chief". These roles are specialization – inheritance link – of "BasicPlayer" or "Player" abstract roles, i.e. roles which cannot be played by agents. All roles inherit of the abstract root role "Soc". Well formed attributes may be ascribed to groups. They concern intra/extra group compatibility of roles among them, minimum and maximum number of role players inside a group, minimum and maximum number of subgroups. *Cardinality* and *compatibility links* express constraints on the way agents play roles in groups. For

[1] Formal definitions of SS and FS are available in [12], CS and NS in [11]

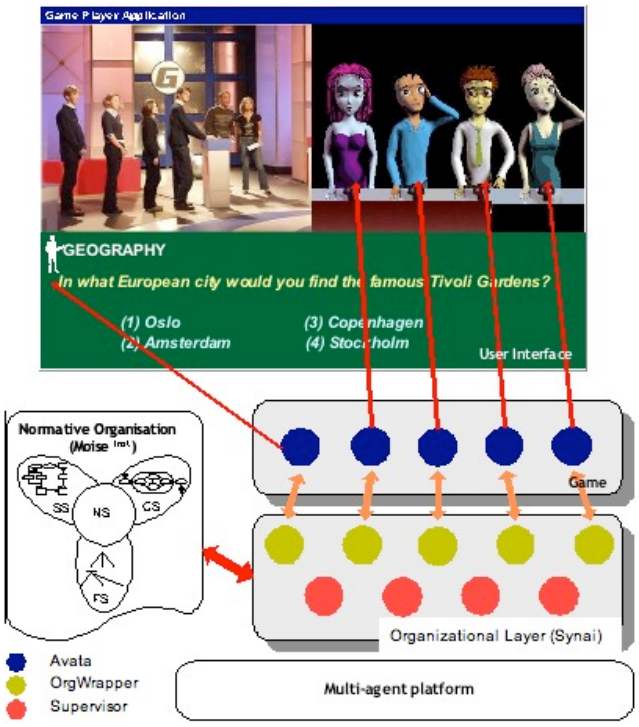


Fig. 1. Global view of normative organization environment for iTV application.

instance, cardinality '1..1' on the composition link ensures that, in an group instance of a "Team", roles can be adopted by only one agent at the same time. A compatibility link between "BasicPlayer" and "Chief", allows the same agent to play those two roles or specialization of those roles. Thus, according to this specification, one agent may have the possibility to play at most two of those five roles. *Links* have direct effect on the agents' behavior. They can be: *acquaintance* links (i.e. agents playing the source role are allowed to have a representation of the agents playing the destination role), *communication* links (i.e. agents are allowed to communicate with the target agents), *authority* links (i.e. source agents are allowed to control target agents). For instance, all roles inheriting from "Player" can communicate between them, and the "Chief" has the authority on all "BasicPlayer". It means that all roles inheriting from this role are under the authority of the "Chief".

3.2 Functional specification

In the *functional specification* (FS), goals that are to be achieved by the organization are structured according to different *social schemes*. A social scheme is a goal decomposition tree where the root is the Scheme's goal. The operators that may appear in these

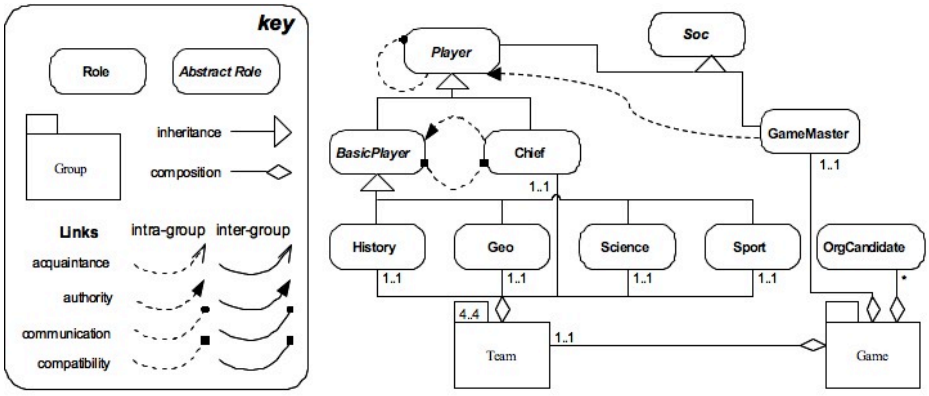


Fig. 2. Structural specification for the domain agents of the iTV application.

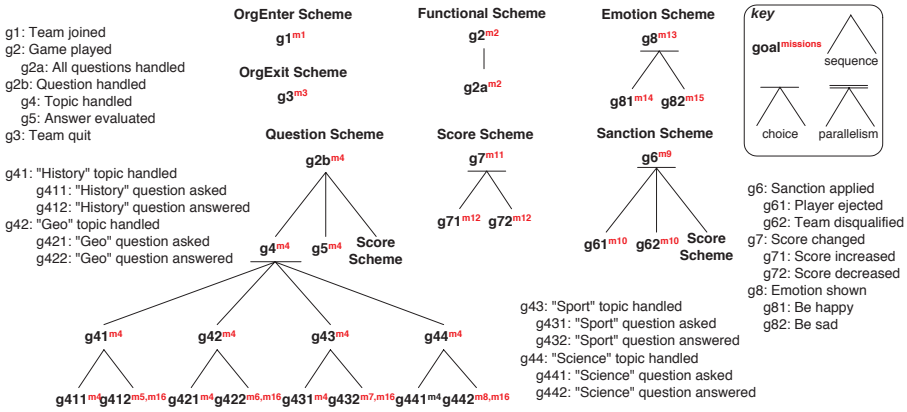


Fig. 3. Functional Specification of the organization for iTV application

plans express the execution in sequence/parallelism and the possibility of choice. All the goals of a social scheme (root goal and subgoals) are structured into *missions*: set of coherent goals that are to be assigned to roles and that an agent can commit to. More precisely, if an agent accepts a mission m_i, it commits to all goals of m_i ($g_j \in m_i$) and the agent will try to achieve a g_j goal only when the goal precondition for g_j is satisfied. In Fig. 3, the main social scheme has a goal "X pts scored" that can be satisfied by the achievement in sequence of "g4", "g5" and of the goals of the "Score Scheme". The "Emotion Scheme" deals with the specification of the emotional behaviour of avatars as: to show either an happy face or a sad one. The "OrgEnter Scheme" (resp. "OrgExit Scheme") defines the principal behaviours for entering (resp. leaving) an organization. We also define a scheme dedicated to sanctions which has to be considered by the supervisor agents (see below). For instance, a sanction consists in a choice between the ejection of a player, the disqualification of the team or the modification of the score.

3.3 Contextual specification

To tackle with the situatedness of applications in evolving environment, a *contextual specification* (CS) captures design-time a priori constraints on the evolution of the organization as a set of contexts and transitions between them (cf. Figure 4).

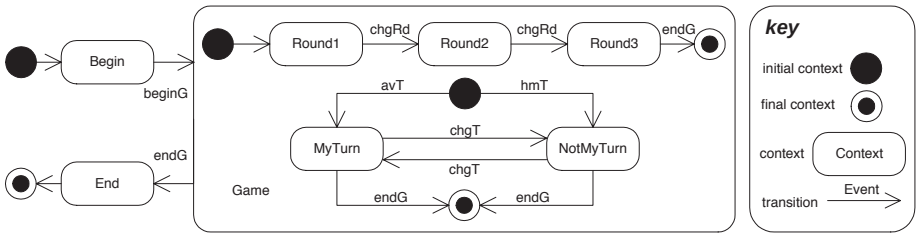


Fig. 4. Contextual Specification for the organization of the iTV application.

A context expresses a state in which agents playing role have to respect specific norms. Transitions define change from one context to an other context given the occurrence of different events. For instance, in the iTV application, the CS is used to express the different rounds of the game that impose changes to the enacted rules. Here the CS starts with a synchronous state "Begin" which allows the televiewer to connect to the system. A macro-context "Game" is decomposed into three rounds sub-contexts. This global context will be used to define the basic rules of the game while the three round sub-contexts will be used to define the corresponding specific rules. The "Game" context is also decomposed into two sub-contexts defining the turn of the players. A round sub-context and a turn sub-context can be active at the same time. Let's notice that the macro-context is active in all of its sub-contexts. The rules defined in the "Game" context are thus inherited in sub-contexts where they keep their status. Finally the last state is the context in which Avatars quit their team.

3.4 Normative specification

Finally, the *normative specification* (NS) glues all specifications (SS, FS and CS) in a coherent and normative organization with the help of norms (see Fig. 5). In $\mathcal{M}\text{OISE}^{Inst}$, norms define rights (i.e. *permission*), duties (i.e. *obligation*, *prohibition*) for agents while playing a role or being member of a group, to execute a *mission* in a particular *context* and during a given *time*. The fulfillment of a norm is supervised by an *issuer* which can apply a *sanction* on the *bearer* of the norm. Norms are represented as the following expression (φ, $context$, $sanction$, $weight$ and time constraint tc are optional):

$$norm : \varphi \rightarrow op(context, issuer, bearer, mission, sanction, weight, tc)$$

φ and $context$ refer respectively to the validity and activation conditions (see below), considered as true if not specified in the expression. op is a deontic operator

($op \in \{O, P, F\}$) which defines a norm as an obligation (O), a permission (P) or a prohibition (F). These operators concern the $mission$ expressed in the FS. Missions that are not prohibited or obliged are considered as permitted. The normative expressions don't refer directly to agents but to groups or roles of the organization in which agents are situated (fields $issuer$, $bearer$). This way, expressions are independent of the kinds of agents that could populate the system at one time. The $issuer$ refers to the role or group that check the status of the norm (fulfilled, violated), whereas the $bearer$ refers to the structural entities on which the norm is applied. Users who specify their own application modelling don't know how the supervision of the normative organization works. That's why they have to set the issuer up to "Supervisor" role (root role of the supervision SS, see below). The $\mathcal{S}$YNAI layer decides automatically what agents supervise what norms. Composition and inheritance that are defined in the SS among groups and roles have consequence on norms:

- When the $bearer$ (resp. $issuer$) is a group, all roles contained in this group, are considered as $bearer$ (resp. $issuer$). of this norm. For instance, the prohibition for the "Team" group to answer a question when it is not its turn, is applied on all the roles being part of this group ("History", "Science", "Geo", "Sport", "Chief").
- If the $bearer$/$issuer$ of a norm is a role r all roles inheriting from r are also concerned by the norm as $beared$/$issuer$. For instance, if a norm obliges the role "Player" to answer a question, all the inheriting roles are obliged to answer a question ("BasicPlayer", "Chief", ...).
- If the $bearer$/$issuer$ of the norm is a group gt then all sub-groups composing gt are concerned by the norm. For instance, if a norm concerns the "Game" group, the norm concerns also the "Team" group. As a consequence, if a norm concerns "Game" and "Team" groups, it concerns also roles belonging to both groups i.e. "History", "Science", "Geo", "Sport", "Chief", "GameMaster" and "OrgCandidate".

A $sanction$ is another norm appearing in the NS that is considered as a "sanction" to apply in case of norm violation[2]. The $weight$ defines a priority used for solving conflicts between norms in case of incoherence, when for instance an agent could be constrained by two contradictory norms[3] (e.g. $N9$ and $N14$ in Fig. 5). 1 is the highest priority.

A norm is *active* when the $context$ referred in the norm equals the current state specified in the CS. As a context can be composed of sub-contexts, if a norm is active in a context then it is also active in sub-contexts. For instance, if a norm is considered active as soon as the OE's state is equal to "Game", the norm will be considered active when the state of the organization will be either in the "Round1", "Round2", "Round3", "MyTurn" or "NotMyTurn" contexts. A norm is *valid* as long as its condition φ is satisfied. φ is the condition that defines the particular state of the OE in which the norm may be valid. As long as φ is satisfied, the norm stays valid. A norm condition could

[2] If a norm id specifies a $sanction$, then the condition of the $sanction$ contains the predicate $violated(id)$.

[3] Even, if this field is not satisfactory in case of two norms having the same weight, it was sufficient in our application. Future works will have to consider this issue.

be a conjunction/disjunction of sub-conditions. A primitive condition consists in one of the following expression:

- an application-dependant predicate (e.g. *sad* or *happy* which test if an avatar shows a sad or happy face);
- a predicate related to the life cycle of the organization such as *number* or *cardinalityMax* which respectively access the number of agents being part of a group and the maximum number of agents that a group may accept;
- a predicate related to the functioning of the normative organization itself such as *violated* which tests if the norm is violated.

A norm can be *fulfilled* or *violated* from the moment it is active *and* valid. The violation detection depends on the deontic operator op and on deadline tc -date or period- appearing in the expression of the norm. If no tc is specified, the end of the $context$ is considered by default.

- An obligation states that the $mission$ ought to be accomplished by the $bearer$ before, after or during a deadline expressed in tc - date or period -. The norm is considered fulfilled if the mission is accomplished by the bearer in term else violated.
- A prohibition hinders the norm's $bearer$ to accomplish the $mission$ in a tc. *deadline*. Contrary to an obligation, a prohibition is considered fulfilled as long as the mission is not accomplished by the bearer until the deadline is over, violated in the other case.
- A permission authorizes the $bearer$ of the norm to accomplish the $mission$ in a tc. In an organization specified with $\mathcal{M}\text{OISE}^{Inst}$, agents don't restrict their action to what is authorized or obliged but all that they are able to do and that is not prohibited.

In the iTV application, norms are used to define game rules as well as what happens before and after the game. For instance, norms $N01$ to $N04$ are related to the management of the organization: constraints on when it's possible to join/quit the team. $N01$ states that *any agent playing the "OrgCandidate" role is obliged to join a team (instance of "Team" group) in case there is still a role to play in this team* (condition $nb(Team) < max(Team)$ composed of two functions representing the number of agents already in the Team group and the maximum of agents allowed in the Team). According to the $context$ field, this norm is active as long as the OE is in the "Begin" context. The norm $N02$ manages the end of the game: *any agent playing a role in the "Team" group is obliged to quit the team (instance of "Team" group) when the organization is in the "End" context*. Moreover (see $NO3$ and $NO4$) in the "Game" context, agents playing the "OrgCandidate" role are forbidden to join a team and agents playing a role in the "Team" group are forbidden to quit the team. $N03$ has a sanction which is expressed as the norm $N17$: *in case of violation of N03, any agent playing the "GameMaster" role has to eject the agent playing the "OrgCandidate" role*. Let us notice that the mission expressed in this normative expression refers to a mission expressed in the "Sanction" scheme of the FS.

Other norms define the rules of the game and constrain its performance. For instance, according to $N05$ and $N06$, as long as the OE is in the "Game" context: *any*

context	id	w.	condition	issuer	bearer	deOp	mission	deadline	sanction
Begin	N01	1	nb(Team)<max(Team)	Supervisor	OrgCandidate	O	m1	---	---
End	N02	1	---	Supervisor	Team	O	m3	---	---
Game	N03	1	---	Supervisor	OrgCandidate	F	m1	---	N17
Game	N04	1	---	Supervisor	Team	F	m3	---	---
Game	N05	1	---	Supervisor	GameMaster	O	m2	---	---
Game	N06	1	---	Supervisor	GameMaster	O	m4	---	---
Game	N07	1	---	Supervisor	Team	P	m13	---	---
Game	N08	2	---	Supervisor	Team	F	m16	---	N18
Round1	N09	3	---	Supervisor	Team	P	m16	< answer_delay	---
Round2	N11	1	---	Supervisor	History	P	m5	< answer_delay	---
Round2	N12	1	---	Supervisor	Geo	P	m6	< answer_delay	---
Round2	N13	1	---	Supervisor	Sport	P	m7	< answer_delay	---
Round2	N14	3	---	Supervisor	Science	P	m8	< answer_delay	---
Round3	N10	1	---	Supervisor	Chief	P	m16	< answer_delay	---
NotMyTurn	N15	1	---	Supervisor	Team	F	m16	---	---
NotMyTurn	N16	1	---	Supervisor	Team	F	m14	---	---
Game	N17	1	violated(N02)	Supervisor	GameMaster	O	m9	---	---
Game	N18	1	violated(N08)	Supervisor	GameMaster	O	m11	---	---

Fig. 5. Normative Specification of the organization of the iTV application. Column "context" refers to the states defined in CS, column "w" contains the weight of the norms, columns "issuer" and "bearer" refer to roles and groups defined in SS, column "deOp" contains *op*, column "mission" contains the missions id specified in FS, column "sanction" refers to the id of norms.

agent playing the "GameMaster" role is obliged to ask question and to evaluate the answer (see missions $m2$ and $m4$ in Functional Scheme). According to $N07$, *any agent playing a role belonging to the "Team" group is forbidden to answer a question during the game*. Exceptions to this prohibition are set by defining specific norms in the context of the different rounds occurring during the game: when OE is in the first and third rounds, $N09$ and $N10$ permit any agent playing respectively a role belonging to the "Team" group and the role "Chief" to answer all questions during the answer delay. When the Organization is in the second round, norms $N11$, $N12$, $N13$, $N14$ allow concerned roles to answer question. Exceptions are expressed by defining for same context, role and mission a different priority in the $weight$.

Finally, norms $N15$ and $N16$ forbid the team to answer a question or to show an happy face when the OE is in the "NotMyTurn" context (i.e. the question is asked to the opponent team).

4 Normative Organizational Layer

While the previous section was concerned with the presentation of $\mathcal{M}$OISE^{Inst}, normative OML, illustrated with the OS installed on the domain agents of the application, this section deals with the issues related to their support into $\mathcal{S}$YNAI, normative Organization Implementation Architecture. As it happens with organizational models [8], implementations can also take either an agent centred or an system centred point of

view [4] (in [13] these points of view are called agent and institutional perspectives). In the former point of view, the focus is on how to develop agent reasoning mechanisms to interpret and reason on the OS and OE. In the latter, the main concern is how to develop an Organization Infrastructure (OI) that ensures the satisfaction of the organizational constraints and norms (e.g. agents playing the right roles, committing to the allowed missions). This point of view is important in heterogeneous and open systems where the agents that enter into the system can 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 organizational representation that enable them to reason about it. However, the agents should follow the OS despite their organizational reasoning abilities.

Many implementations of the OI follow the general architecture depicted in Fig. 6. Domain agents are responsible to achieve organizational goals and use an *organizational proxy* component to interact with the organization (OS and OE). The *organizational layer* is responsible to bind all agents in a coherent system and provides some services for them. The communication layer is responsible for connecting all components of the infrastructure in a distributed and heterogeneous applications.

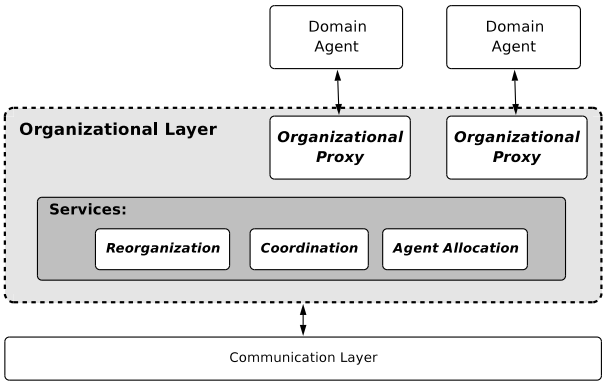


Fig. 6. Common Organization Implementation Architecture for open MAS

4.1 SYNAI

Domain agents playing the game evolve in the OE resulting of the OS specified by the designer with the OML $\mathcal{M}$OISE^{Inst}. The OE consists in the current states of SS (roles played by agents, existing instances of groups), FS (current committed / executed / waiting missions and goals), CS (current executed/active states) and NS (current active / valid / fulfilled / violated norms). Being autonomous (under the control of a user),

[4] We prefer here system-centred to organization-centred in order to avoid confusion even if, as we have seen, the organization is reified in OE

avatars can decide to not respect the constraints stated in the OS: adopting a role in the OE which is not authorized in the OS, violating a norm, ...).

$\mathcal{S}$YNAI aims at managing and controlling the functioning of this OE by the way of different events corresponding to the entry/exit of agents of the OE, adoption/leaving roles, change of context, commitment to missions, achievement of goals, etc. Receiving requests from agents, it detects if they violate or not constraints stated in the SS and the NS (cf. Fig. 7). For instance it verifies that an agent plays compatible roles or that it is authorized to commit on mission according to the role it is playing and to the current active and valid norms.

$\mathcal{S}$YNAI is composed of a set of different supervisor agents for the management of each entity deriving from the specification of the OS: *StructManagerAg* for the SS entity, *FunctManagerAg* for the FS entity, *ContextManagerAg* for the CS entity and *NormManagerAg* for the NS entity. The *OrgManagerAg* is able to manage the OE and to coordinate the other agents. Each domain agents is supported by an *OrgWrapperAg* which is a kind of facilitator for the domain agent to access and interact with the supervisor agents.

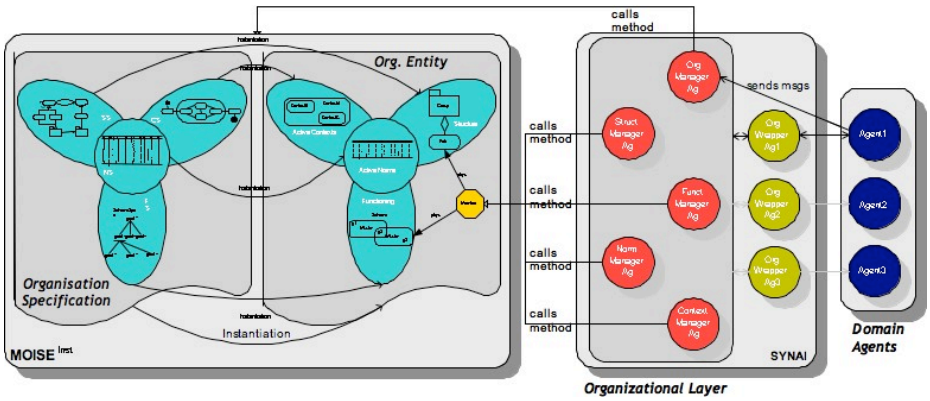


Fig. 7. Supervisor agents of the $\mathcal{S}$YNAI Organizational Layer.

4.2 Normative organization of the Organizational Layer

In order to supervise the enactment of the organization on the agents and to insure that the norms are fulfilled, the supervisor agents have to understand the $\mathcal{M}$OISE^{Inst} model. In order to make the implementation of the organizational layer independant of the structure of the supervisor agents, we chose to make explicit its organization, using the $\mathcal{M}$OISE^{Inst} OML. Supervisor agents are thus organised the same way as domain agents are, i.e. according to the OS defined with $\mathcal{M}$OISE^{Inst} in order to structure and to define their rights and duties (see Fig. 8).

The OS governing the supervisor agents is thus defined with a SS, FS, CS and NS as follows. The SS is composed of the only group "Supervision" containing the roles that supervisor agents would play in order to manage the domain agents OE. Since, all roles inherit of the "Supervisor" role, they can communicate with each other (communication link from "Supervisor" to itself). The cardinality '1..1' except for "OrgWrapper" ensures that only and only one supervisor agent will play a role in this group.

The FS defines the main goals of the supervision system which is to keep the organization in a coherent state: choice between correcting the violation (*gOC* goal) or blocking the violation intention (*gOB* goal) (see supervision scheme in Fig. 8). As expressed in the scheme, the steps of supervision -expressed as social schemas- are: violation detection, correction or not of the violation (according to the choice) and sanction of the culprit. Constraints to be checked come from the SS (cardinalities, links, etc.), from the FS (mission cardinalities) and from the NS (norms). Thus a violation detection is either a NS violation, a FS violation or a NS violation.

The CS defines the contexts that are used for the choice of the arbitration strategies in relation to the achievement of goals *gOC* or *gOB*. During the activity of the OE, an event can be created which causes the change of state implying, according to the norms, a change in the arbitration strategy: correct violations or block violations.

The norms of the NS (cf. the table of the Fig 8) express that the organization must be kept in a coherent state by correcting violations in the "CorrArb" context ($NA1$ to $NA5$) and by blocking actions with violation intention in the "BlocArb" context ($NA6$). They express that the detection must be done in whatever context ($NA7$ to $NA10$).

The supervision OS is integrated into the domain OS to compose the global normative organization as follows. The supervision SS is integrated into the domain SS by including the "Supervision" group into the "Game" group and by installing an authority link from the "Supervisor" role on the "Soc" role. As a consequence, agents from $\mathcal{S}$YNAI playing one role of the supervision SS have authority and can control activities of all domain agents playing roles of the domain SS belonging to group "Game". The supervision FS is just added to the domain FS. The "Sanction Scheme" defined for the domain is available and usable by the "Supervision Scheme" (see for instance the call to this scheme). This inclusion of the "Sanction Scheme" of the domain OS into the "Supervision Scheme" allows the use of domain specific sanction strategies into a generic supervision scheme that can be the same for all applications. The supervision CS is added to the domain CS as parallel transition state diagram. Both CS form a global CS. The supervision NS is added to the domain NS, composing the global NS of the normative organization.

Enacting this OS on the supervisor agents leads to an OE where *OrgManagerAg* plays "InstManager" and "Arbitrator", and each supervisor agent plays the role corresponding to its capabilities: *StructManagerAg* plays "StructManager", *FunctManagerAg* plays "FunctionalManager" and so on.

5 Related Works

Different organization modeling languages exist in the multi-agent domain. They use different modeling dimensions to cope with the complexity of the definition of multi-agent organizations. As in $\mathcal{M}$OISE^{Inst}, they exhibit either a *structural* dimension talking about the structure of the collective level of an MAS (e.g. AGR [14], ISLANDER [15], $\mathcal{M}$OISE^{+}), generally in terms of roles/groups/links or a *functional* one talking about the global functioning of the system (e.g. TMS [16], TEAMCORE [17], $\mathcal{M}$OISE^{+}). Some models as in ISLANDER, add a *dialogical* dimension talking about the interaction in terms of communications between the agents. Others introduce an environmental dimension allowing to constrain the anchoring of organization in an environment such as in AGRE [18]. Inspired of ISLANDER, $\mathcal{M}$OISE^{Inst} has introduced a contextual specification to define a-priori the transition between different configurations of norms, structures and plans. We won't compare all these OML, here, in terms of the primitives or modeling power each one can offer (refer for instance to [19] for a systematic comparison of these models). Depending on these different dimensions, their influence on the agents' behavior may be quite different. In models such as TMS where only the functional dimension is specified, the organization has nothing to "tell" to the agents when no plan or task can be performed. Otherwise, if only the structural dimension is specified as in AGR, the agents have to reason for a global plan every time they want to work together. Even with a smaller search space of possible plans, since the structure constrains the agents options, this may be a hard problem. Furthermore, the plans developed for a problem are lost, since there is no organizational memory to store these plans. Thus, in the context of open systems, we hypothesize that if the organization model specifies both dimensions as in $\mathcal{M}$OISE^{Inst} or TEAMCORE or a third one as in ISLANDER then the MAS that follows such a model can be more effective in leading the group behavior to its purpose. On the agents' side, they can develop richer reasoning abilities about the others and their organization. Agents may gain more information on the possible cooperation (in terms of roles, groups, but also on the possible goals under achievement or on the performative structures that can be used) that may be conducted with the other agents.

Besides those dimensions, the *deontic* and *normative* dimensions used respectively in $\mathcal{M}$OISE^{+} and ISLANDER or $\mathcal{M}$OISE^{Inst} address the agents autonomy problematic and consider organizations as normative constructs aiming at controling in an explicit manner the multi-agent system. While in other OML the agents are supposed to be benevolent and compliant (de-facto) to the OS, these two models add the possibility for agents to develop explicit reasoning on their autonomy with respect to the organizational constraints.

Turning now to the Organization Implementation Architecture that supports such normative OML, a few takes the same point of view of the normative organization layer developed in $\mathcal{S}$YNAI. AMELI [20] is the organization layer for ISLANDER. It provides a social layer which controls and helps the agents to participate in an e-institution with specialized governors. $\mathcal{S}$-$\mathcal{M}$OISE^{+} [21] is the organizational layer for managing $\mathcal{M}$OISE^{+} organizations. It provides the agents evolving in the organization with personal "OrgBoxes" giving a partial view of the organization. OrgBoxes serve as interface between heterogeneous agents and the organization. There is just one "OrgManager"

for controlling the access of the agents to the organization. The deontic expressions are enforced but not controlled. For instance, violation of an obligation is hardly detectable.

6 Conclusion and Perspectives

We have presented in this paper an ongoing work for the definition of normative organization environment. It is composed of an normative organization modeling language $\mathcal{M}$OISE^{Inst} with its accompanying organizational layer $\mathcal{S}$YNAI. Different modeling dimensions are mobilised to program rich organizational patterns to control or to help the cooperation of the agents in the system: structural, functional, contextual and normative. As noticed, these dimensions are not exclusive and some dimensions are still proposed in related works (e.g. environment, dialogical). In $\mathcal{M}$OISE^{Inst}, the agents' autonomy concern is considered with the explicit definition of norms that bind all the dimensions together. The agents' autonomy is also taken into account in the organizational layer that support $\mathcal{M}$OISE^{Inst} with the definition of supervisor agents aiming at controling and enforcing norms into the system. Two kinds of agents evolve in such organization: the domain agents and the supervisor agents. With $\mathcal{M}$OISE^{Inst} we expressed at a "meta-level" the supervision organization that aims at controlling the supervisor agents by defining roles that they will play, as well as the missions related to their ability to detect norms violations and to punish culprit domain agents.

However some challenges still need to be considered and solved: decentralization of the organization infrastructure to address the scaling problem, developping reasoning abilities in order to integrate top-down predefined organizations (organization-centred) with bottom-up emergent organizations (agent-centred), with eventually solving conflicts (e.g. what if some agent playing a role must interact with another agent X playing its role, but the agent knows that X can not perform some intended task and it even prefer to interact with agent Y?), to undertsand in more depth every dimension, leading to an organization ontology to enable interoperation, reorganization issues in general (how to evaluate? how to change?).

ACKNOWLEDGEMENT

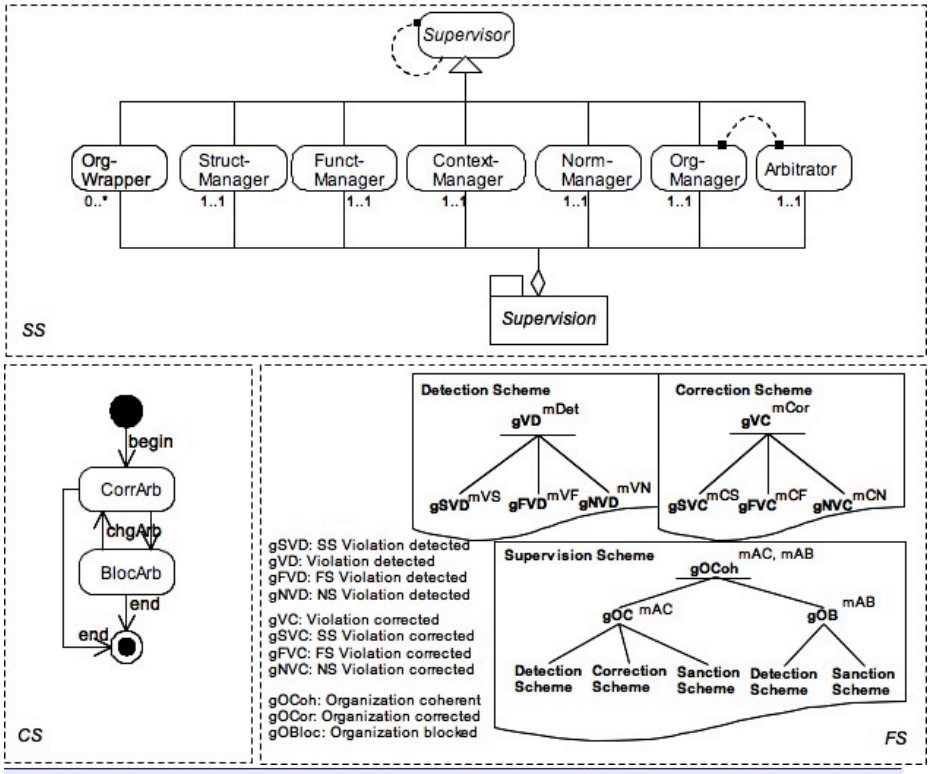
Part of this research has been funded by the University of Luxembourg through the LIASIT project.

References

1. Omicini, A., Ricci, A., Goldin, D.: Introduction to the workshop. In: Second International-Workshop on Theory and Practice of Open Computational Systems (TAPOCS 2004). (2004)
2. Esteva, M., Rodríguez-Aguilar, J.A., Rosell, B., L., J.: AMELI: 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), New York, ACM (2004) 236–243

3. Hubner, 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.: 16th Brazilian Symposium on Artificial Intelligence (SBIA'02). Number 2507 in LNAI, Springer (2002) 118–128
4. Hübner, J.F., Sichman, J.S., Boissier, O.: $\mathcal{S}$-$\mathcal{M}$OISE$^+$: A middleware for developing organised multi-agent systems. In Boissier, O., Dignum, V., Matson, E., Sichman, J.S., eds.: Proceedings of the International Workshop on Organizations in Multi-Agent Systems, from Organizations to Organization Oriented Programming in MAS (OOOP'2005). Volume 3913 of LNCS., Springer (2006) 64–78
5. Bellifemine, F., Poggi, A., Rimassa, G.: Jade - a fipa-compliant agent framework. In: 4th International Conference and Exhibition on The Practical Application of Intelligent Agents and Multi-Agents, London (1999)
6. Poslad, S., Buckle, P., Hadingham, R.: The fipa-os agent platform: Open source for open standards. In: PAAM. (2000)
7. Dignum, F.: Agents, markets, institutions, and protocols. In Dignum, F., Sierra, C., eds.: Agent-Mediated Electronic Commerce - The European AgentLink Perspective. Volume 1991 of Lecture Notes in Artificial Intelligence., Springer Verlag (2001) ISBN 3-540-41671-4.
8. Boissier, O., Hbner, J.F., Sichman, J.S.: Organization oriented programming, from closed to open organizations. In: Engineering Societies in the Agents World VI, Sixth International Workshop, ESAW 2006, Dublin, Ireland, September, 2006, Revised Papers. Lecture Notes in Computer Science, Springer Verlag (2007)
9. Jones, A., Carmo, J.: Deontic logic and contrary-to-duties. In: Handbook of Philosophical Logic. Kluwer (2001) 203–279
10. Castelfranchi, C., Dignum, F., Jonker, C.M., Treur, J.: Deliberate normative agents: Principles and architecture. In: Proceedings of The Sixth International Workshop on Agent Theories, Architectures, and Languages (ATAL-99). (1999)
11. Gâteau, B., Boissier, O., Khadraoui, D., Dubois, E.: $\mathcal{M}$OISE^{Inst}: An organizational model for specifying rights and duties of autonomous agents. In van der Torre, L., Boella, G., eds.: 1st International Workshop on Coordination and Organisation (CoOrg 2005) affiliated with the 7th International Conference on Coordination Models and Languages, Namur - Belgium (2005)
12. Gâteau, B., Khadraoui, D., Dubois, E.: Architecture e-business sécurisée pour la gestion des contrats. In: 3ème Conférence sur la Sécurité et Architectures Réseaux (SAR), La Londe, Cote d'Azur - France (2004)
13. 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)
14. Ferber, J., Gutknecht, O.: A meta-model for the analysis and design of organizations in multi-agent systems. In: Third International Conference on Multi-Agent Systems (ICMAS98), Paris, France (1998) 128–135
15. Esteva, M., Padget, J., Sierra, C.: Formalizing a language for institutions and norms. In: Intelligent Agents VIII: 8th Intern. Workshop. Volume 2333 of LNAI. (2002) 348–366
16. Lesser, V., Decker, K., Wagner, T., Carver, N., Garvey, A., Horling, B., Neiman, D., Podorozhny, R., NagendraPrasad, M., Raja, A., Vincent, R., Xuan, P., Zhang, X.: Evolution of the gpgp/taems domain-independent coordination framework. Autonomous Agents and Multi-Agent Systems **9** (2004) 87–143 Kluwer Academic Publishers.
17. Pynadath, D.V., Tambe, M.: An automated teamwork infrastructure for heterogeneous software agents and humans. Autonomous Agents and Multi-Agent Systems **7** (2003) 71–100
18. Ferber, J., Michel, F., Baez, J.: AGRE: Integrating environments with organizations. In: E4MAS'04, 1st Int. Workshop. Volume 3374 of LNCS. (2005) 48–56

19. L. Coutinho, J.S. Sichman, O.B.: Modeling dimensions for multi-agent systems organizations. In Dignum, V., Dignum, F., Edmonds, B., Matson, E., eds.: Agent Organizations: Models and Simulations (AOMS), Workshop held at IJCAI 07. (2007)
20. Esteva, M., Rosell, B., Rodriguez-Aguilar, J.A., Arcos, J.L.: Ameli: An agent-based middleware for electronic institutions. In Jennings, N.R., Sierra, C., Sonenberg, L., Tambe, M., eds.: 3rd international joint conference on Autonomous Agents & Multi-Agent Systems (AAMAS). Volume 1., Columbia University, New York City - USA, ACM Press (2004) 236–243 ISBN 1-58113-864-4.
21. Hubner, J.F., Sichman, J.S., Boissier, O.: S-moise+: A middleware for developing organized multi-agent systems. In: International Workshop on Organizations in Multi-Agent Systems: From Organizations to Organization Oriented Programming (OOOP). (2005)



context	id	w.	condition	issuer	bearer	deOp	mission	deadline	sanction
CorrArb	NA01	1	---	OrgManager	Arbitrator	O	mAC	---	---
CorrArb	NA02	1	---	Supervisor	Arbitrator	O	mCor	---	---
CorrArb	NA03	1	---	Supervisor	StructManager	O	mCS	---	---
CorrArb	NA04	1	---	Supervisor	FunctManager	O	mCF	---	---
CorrArb	NA05	1	---	Supervisor	NormManager	O	mCN	---	---
BlocArb	NA06	1	---	Supervisor	Arbitrator	O	mAB	---	---
---	NA07	1	---	Supervisor	Arbitrator	O	mDet	---	---
---	NA08	1	---	Supervisor	StructManager	O	mVS	---	---
---	NA09	1	---	Supervisor	FunctManager	O	mVF	---	---
---	NA10	1	---	Supervisor	NormManager	O	mVN	---	---

Fig. 8. Organization Specification of $\mathcal{S}$YNAI