OMNI: Introducing Social Structure, Norms and Ontologies into Agent Organizations

Virginia Dignum, Javier Vázquez-Salceda , Frank Dignum
Institute of Information and Computing Sciences
Utrecht University, The Netherlands
{virginia, javier, dignum}@ cs.uu.nl

Abstract

In this paper, we propose a framework for modelling agent organizations, OMNI*, that allows the balance of global organizational requirements with the autonomy of individual agents. It specifies global goals of the system independently from those of the specific agents that populate the system. Both the norms that regulate interaction between agents, as well as the contextual meaning of those interactions are important aspects when specifying the organizational structure.* OMNI *integrates all this aspects in one framework. In order to make design of the multi-agent system manageable, we distinguish three levels of abstraction with increasing implementation detail. All dimensions of* OMNI *have a formal logical semantics, which ensures consistency and possibility of verification of the different aspects of the system.* OMNI *is therefore utmost suitable for the modelling of all types of MAS from open to closed environments.*

1. Introduction

In closed domains, the design of MAS can suffice with the idea that agents are mere performers of organizational roles or functions, interacting according to fixed protocols and unable to deviate from expected behavior [20]. As such, agent autonomy is rather limited. In open domains, agents are self-governed autonomous entities that pursue their own individual goals based only on their own beliefs and capabilities [1].

Comprehensive models for MAS must, on the one hand, be able to specify global goals and requirements of organizations but, on the other hand, cannot assume that participating agents will act according to the needs and expectations of the system design. Concepts as organizational rules [19], norms and institutions [6], [7], and social structures [13] arise from the idea that the

effective engineering of MAS needs high-level, agent-independent concepts and abstractions that explicitly define the organization in which agents live [20]. These are the rules and global objectives that govern the activity of an enterprise, group, organization or nation.

Given that agents might deviate from expected behavior, open societies need mechanisms to systematize, defend and recommend right and wrong behavior, which can inspire trust into the agents that will join them. Norms are commonly used means to describe such expected behavior. Finally, organizational models must provide means to represent concepts and relationships in the domain that are rich enough to *cover* the necessary contexts of agent interaction while keeping in mind the *relevance* of those concepts for the global aims of the system.

In this paper we propose a framework for agent organizations, OMNI(Organizational Model for Normative Institutions) presenting a first attempt to cover all the above mentioned aspects in a way that is usable for both open and closed systems.

The paper is organized as follows: In §2 we present a generic description of the OMNI framework. In §3, we discuss the abstract level of an organization. Then we will focus on the description of the Organizational dimension (§4), the Normative dimension (§5) of the e-Organization and present an outline in §6 of the kind of ontologies and communication languages needed in the Ontological Dimension. In §7 we compare our framework with other well known models. We end this document with our conclusions and outline future lines of research. Throughout the paper the different components of a society are illustrated using an organization that has as main global objective the realization of conferences.

2. The OMNI framework

OMNI is an integrated framework for modelling a

whole range of MAS, from closed systems with fixed participants and interaction protocols, to open, flexible systems that allow and adapt to the participation of heterogeneous agents with different agendas. This approach is rather unique, as most existing frameworks concentrate in a specific type of MAS. OMNI is composed by three dimensions: **Normative**, **Organizational** and **Ontological** that describe different characterizations of the environment. The model is based on two recent MAS models, OperA [5], and HARMON*IA* [17]. Figure 1 depicts the different modules that compose our proposed framework organized into three levels of abstraction:

- the **Abstract Level**: where the statutes of the organization to be modelled are defined in a high level of abstraction. This step is similar to a first step in the requirement analysis. It also contains the definition of terms that are generic for any organization (that is, that are incontextual) and the ontology of the model itself.

- the **Concrete Level**: where all the analysis and design process is carried on, starting from the abstract values defined in the previous level, refining their meaning in terms of norms and rules, roles, landmarks and concrete ontological concepts.

- the **Implementation Level**: where the design in the Normative and Organizational dimensions is implemented in a given multi-agent architecture, including the mechanisms for role enactment and for norm enforcement.
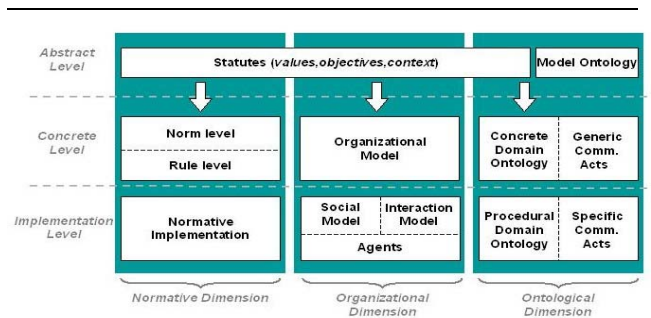
Figure 1. The OMNI **framework.**

The division of the system into these three levels aims to ease the transition from the very abstract statutes, norms and regulations to the very concrete protocols and procedures implemented in the system. Different domains have different requirements concerning normative, organizational and communicative char-

acteristics, which means that not always all three modules have the same impact or are even needed: in those domains with none or small normative components, design is mainly guided by the Organizational Dimension, while in highly regulated domains the Normative Dimension is the most prominent.

3. The Abstract Level

Statutes indicate, at the most abstract level, the main *objective* of the organization, the *values* that direct the fulfilling of this objective and they also point to the *context* where the organization will have to perform its activities.

In the conference scenario, we can take as example a research consortium such as the IFMAS (International Foundation for Multi-Agent Systems). The statutes state: *"IFMAS is a non-profit corporation whose purpose is to promote science and technology. In pursuit of its purposes, IFMAS will engage in activities including, but not limited to: (1) Coordinating and arranging seminars on artificial intelligence and multi-agent systems; (2)..."*. In this statement we can find:

1. the *objectives*: the main objective of this organization is to promote science and technology. Another objective is the organization of seminars.

2. the *context*: IFMAS states that it operates in the area of artificial intelligence and multi-agent systems.

3. the *values*: The IFMAS is a *non-profit* organization. Implicit in the latter part, it also says that sharing is also a value of the organization.

The *objectives* of the organization express the overall goals of the society. As far as the organization has control over the actions of the agents acting within that organization, it will try to ensure that they perform actions leading to the overall goal of the society. We will see in §4.1 and §4.2 how these objectives influence the design process in the *Organizational Dimension*.

The *values* of the organization are beliefs that individuals have about *what* is important, both to them and to the society as a whole. A value, therefore, is a belief (right or wrong) about the way something should be. Values define beliefs about, for instance, what is acceptable or unacceptable, good or bad. In our framework, values are the basis of the *Normative Dimension*.

The environment, or *context*, of an organization can be seen as consisting essentially of other societies or organizations that are interdependent and each influence the other.

4. The Organizational Dimension

The design of agent organizations must capture on the one hand, the structure and requirements of the society owners, and on the other hand, must assume that participating agents must be available that are able and interested in enacting society roles. In OMNI, the Organizational Dimension consists of a 3-layered model: based on the concerns identified in the Abstract Level, the Concrete Level specifies the structure and objectives of a system as envisioned by the organization, and the Implementation Level describes the activity of the system as realized by the individual agents. This separation enables OMNI models to respect the autonomy of individual agents while ensuring conformance to organizational aims.

4.1. The Organizational Abstract Level

The abstract level of the Organizational Dimension describes which are the aims and concerns of the organization with respect to the social system. At the abstract level, as we saw in §3, this is defined by means of a list of the organization's externally observable *objectives*, that is, the desired states of affairs in the life of the society.

A common way to express the objectives of an organization is in terms of its expected functionality. The determination of the overall objectives of the society follows a process of elicitation of functional (*what*) and interaction (*how*) requirements. For example, how should a conference be organized, in terms of program, location, co-located workshops, etc. To identify the objectives of an organization, it is important to characterize the different stakeholders (*who*) of the organization, their requirements, expectations, constraints and relationships to each other. Stakeholders in the conference scenario are researchers, organizers, etc. Stake holders are the basis for the identification of *roles* in the concrete level of specification of an organization (see §4.2.1).

4.2. The Organizational Concrete Level

The Concrete Level of the organizational dimension specifies the *means* to achieve the objectives identified in the the abstract level as an *Organizational Model* (OM). The OM describes the structure and global characteristics of a domain from an organizational perspective. That is, from the premise that it is the society goals that determine roles and interaction norms. Organizational characteristics of an agent society are specified in the OM in terms of its *Social* (§4.2.1) and *Interaction Structures* (§4.2.2).

The definition of these structures alone is not enough, as specification of a society should include the description of concepts and relations holding in the domain, and of those behaviours accepted as 'good'. Therefore, these structures should be linked with the *role norms*, *scene norms* and *transition norms*, defined in the concrete level of the Normative Dimension (see §5.2), and with the ontologies and communication languages defined in the concrete level of the Ontological Dimension (see §6).

The organization design is also guided by the coordination needs of the domain. These determine the type of roles and tasks necessary to facilitate the tasks of the organization. We identify three basic coordination types: *market*, *hierarchy* and *network* that are defined as *Architectural Templates*.

4.2.1. The Social Structure The social structure of an organization describes the roles holding in the organization. It consists of a list of role definitions, *Roles* (including their objectives, rights and requirements), such as PC-member, program chair, author, etc.; a list of role groups' definitions, *Groups*; the relations between roles by a *Role Hierarchy* graph, and a *Role Dependencies* graph.
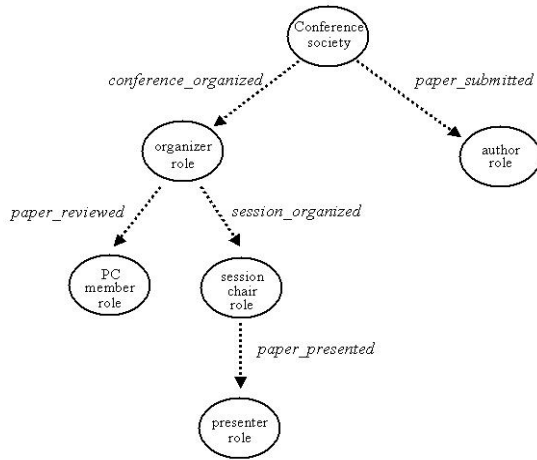
Roles are the main element of the *Social Structure*. From the society perspective, role descriptions should identify the activities and services necessary to achieve society objectives and enable to abstract from the individuals that will eventually perform the role. From the agent perspective, roles specify the expectations of the society with respect to the agent's activity in the society. In OMNI, the definition of a role consists of an identifier, a set of role objectives, possibly sets of subobjectives per objective, a set of role rights, a set of norms and the type of role. An example of role description is presented in table 1. The meaning and relationships between the concepts used is formally specified in the Ontological Dimension, and the formal specification of norms in the Normative Dimension.

Groups provide means to collectively refer to a set of roles. Moreover, *groups* are used to specify norms that hold for all roles in the group. Groups are defined by means of an identifier, a non-empty set of roles, and the group norms. Norms of a group must be consistent with the norms of the roles in the group. An example of a group in the conference scenario is the organizing team consisting of the roles *program chair*, *local organizer*, and *general chair*.

Abstract society objectives form the basis for the definition of the objectives of roles. The distribution of objectives in roles is defined by means of the *Role Hierarchy*. Different criteria can guide the definition of *Role Hierarchy*. In particular, a role can be refined by

| Id	PC_member
Objectives	paper_reviewed(Paper,Report)
Sub-objectives	{read(P), report_written(P, Rep), review_received(Org, P, Rep)}
Rights	access-confmanager-program(me)
Norms & Rules	PC_member is OBLIGED to understand English
	IF paper_assigned THEN PC_member is OBLIGED
	to review paper BEFORE given deadline
	IF author of paper_assigned is colleague
	THEN PC_member is OBLIGED to refuse to review ASAP
Type	external

Table 1. PC member role description.

Scene	Review Process
Roles	Program-Chair (1), PC-member(2..Max)
Results	$r_1 = \forall$ P $\in$ Papers: reviews_done(P, review1, review2)
Interaction Patterns	PATTERN(r_1): *see figure 3*
Norms & Rules	Program-Chair is PERMITTED to assign papers
	PC-member is PERMITTED to review papers assigned before deadline

Table 2. Script for the Review Process scene.

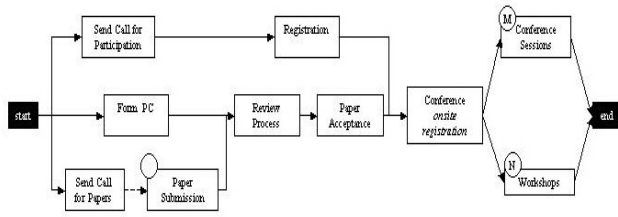**Figure 2. Role dependencies in a conference.**
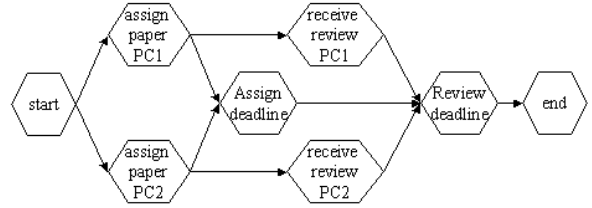
Figure 3. Landmarks pattern for review process.

4.2.2. The Interaction Structure

Interaction is structured in a set of meaningful scenes that follow predefined abstract scene scripts. Examples of scenes are the registration of participants in a conference, which involves a representative of the organization and a potential participant, or paper review, involving program committee members and the PC chair. A *scene script* describes an scene by its players (roles), its desired results and the norms regulating the interaction. In the OM, scene scripts are specified according to the requirements of the society. The results of an interaction scene are achieved by the joint activity of the participating roles, through the realization of (sub-)objectives of those roles. A scene script establishes also the desired *interaction patterns* between roles, that is, a desired combination of the (sub-) objectives of the roles.

In OMNI, interaction description is declarative in nature, rather than describing the exact activities. Interaction objectives can be more or less restrictive, giving the agent enacting the role more or less freedom to decide how to achieve the role objectives and interpret its norms. Following the ideas of [15], we call such expressions *landmarks*, that is, conjunctions of logical expressions that are true in a state. Figure 3 shows the informal landmark pattern for the *Review Process*.

Several different specific actions can bring about the same state, that is, landmarks actually represent families of protocols. The use of landmarks to describe activity enables the actors to choose the best applicable actions, according to their own goals and capabilities.

The relation between scenes is represented by the

decomposing it in sub-roles that, together, fulfill the objectives of the given role. This refinement of roles defines *Role Dependencies*. A dependency graph represents the dependency relations between roles. Nodes in the graph are roles in the society. Arcs are labelled with the objectives of the parent role for whose realization the parent role depends on the child role. Part of the dependency graph for the conference society is displayed in figure 2. For example, the arc between nodes PC-Chair and PC-member represents the dependency $PC\text{-}Chair \succeq_{paper-reviewed} PC\text{-}member$. The way the objective g in a dependency relation $r_1 \succeq_g r_2$ is actually passed between $r1$ and $r2$ depends on the coordination type of the society, defined in the Architectural Templates. In OMNI, we identify three types of role dependencies: *bidding*, *request* and *delegation*.

Figure 4. Interaction Structure in the Conference scenario.

Interaction Structure (see figure 4). In this diagram, *transitions* describe a partial ordering of the scenes, plus eventual synchronization constraints. Note that several scenes can be happening at the same time and one agent can participate in different scenes simultaneously. Transitions also describe the conditions for the creation of a new instance of the scene, and specify the maximum number of scene instances that are allowed simultaneously. Furthermore, the enactment of a role in a scene may have consequences in following scenes. In these cases the *evolution relations* between roles describe the constraints that hold for the role-enacting agents as they move from scene to scene.

4.3. The Organizational Implementation Level

OMNI assumes that individual agents are designed independently from the society to model goals and capabilities of a given entity. Individual agents will join a society as enactors of organizational role(s), as a means to realize their own goals [3].

4.3.1. Social Model Agent populations of the organizational model are described in the social model (SM) in terms of commitments regulating the enactment of roles by individual agents. Depending of the specific agents that will join the organization, several populations are possible for each organizational model.

When an agent applies, and is accepted, for a role, it will commit itself to the realization of the role goals and to act within the society according to the role constraints. The commitments are specified as social contracts. A *social contract* describes the conditions and rules applying to an agent enacting role(s) in the agent society. Given an agent society S, a social contract for agent s enacting role r is defined as a tuple

$$social\text{-}contract = \langle a, r, CC \rangle$$

where a is an agent, $r \in roles(S)$ is a role, and CC is a set of contract clauses (including (1) the time period the contract holds -either from date to date, or until certain states hold; (2) specific agreements and conditions governing the role enactment, and (3) the sanctions to take when norms are violated). In the conference scenario, when a researcher becomes PC member, her social contract will describe for example the agreements concerning number of papers to review (which possibly may deviate from the standard number desired by the conference).

Social contracts identify *role enacting agents* (*reas*) that compose the society. For each agent, the *rea* reflects the agent's own requirements and conditions concerning its participation in the society. Making agreements explicit and formal, allows the verification of whether the animated society behaves according to the design specified in the OM. Social contracts in OMNI are a two-sided agreement between agents and roles instead of a one-sided API description of role enactment, as have been proposed by other researchers [16, 12]. In the extreme, if all is expressed in the role definition and no room is left for negotiation, OMNI social contracts can function as these API's.

4.3.2. Interaction Model OMNI provides two levels of specification for interactions. While the OM provides a script for interaction scenes according to the organizational aims and requirements, the IM, realized in the form of contracts, provides the interaction scenes such as agreed upon by the agents. Due to the autonomous behavior of agents, the interaction model must be able to accommodate other interaction contracts describing new, emergent, interaction paths, to the extent allowed by the organizational and social models.

An interaction scene results from the instantiation of a scene script, described in the OM, to the reas actually enacting it and might include specializations or restrictions of the script to the requirements of the reas. An interaction contract describes the conditions and rules applying to interaction between agents in the agent society. That is, the clauses in an interaction contract specify actual instantiations of interaction scene scripts and must indicate the actors involved and the specific agreements and sanctions concerning the scene to be played. The contract must furthermore involve sufficient reas to cover all the needed roles in the scene. Besides the refinement of the script to the desires and characteristics of the agents participating in the scene instance, interaction contracts must describe the protocol agreed by those agents to fulfil the script landmarks. Interaction protocols are the concrete representation of the refinement of scene script landmarks with the par-

ticularities imposed by the participants to the specific communicative capabilities of those participants. Given a society S and a scene $s \in scenes(S)$, an *interaction contract* is defined as a tuple

$$interaction\text{-}contract = \langle A, s, CC, P \rangle$$

where the set of agents $A = \{a \in Agents : rea(a, r, s) | r \in roles(s)\}$ represents the set of all agents enacting reas participating in interaction scene s, CC is a set of contract clauses, that is, possible conditions and deadlines concerning the results and interaction patterns of scene s, and P is the protocol to be followed. Protocols describe the actual interaction between reas. A rea interaction protocol describes a communication pattern for reas that fulfills the scene script landmarks. In the conference scenario, an interaction contract for the Review Process scene can specify, for example, that actors will follow the *ConfMaster* protocol.

5. The Normative Dimension

In the same way as the Organizational Dimension, the Normative Dimension is composed by the different levels of abstraction. The translation steps from one level to the following are described in a formal way, as we aim to to be able to verify if a given organization complies to all the norms that are specified in the regulations. The connection between levels is very useful not only in its *top-down* direction (guiding the design), but also from *bottom up* (agents can trace the origin of a given protocol and reason in terms of the rules and norms the protocol implements).

5.1. The Normative Abstract level

At this highest level of abstraction, the *values* fulfill the role of norms in the sense that they determine the concepts that are used to determine the value or utility of situations. However, values do not specify *how*, *when* or in *which* conditions individuals should behave appropriately in any given social setup. This will be defined later by abstract norms, concrete norms and rules (see section 5.2).

The values of the Conference Society can be described as desires. For example:

- the *information sharing* value can be described as $D(share(info))$,

- the *non-profit* value can be described as $D(non\text{-}profit(organization))$,

However, besides a formal syntax, this does not provide any meaning to the concept of *value*. In our framework the meaning of the values is defined by the norms that contribute to this value. In an intuitive way we can see this translation process as follows:

$$\vdash_{org} D(\varphi) \mapsto O_{org}(\varphi)$$

meaning that, if an organization *org* values situations where φ holds higher than situation where φ does not hold, then such value can be translated in terms of a norm (an obligation of the organization *org*) to fulfill φ. In our framework a norm *contributes to a value* if fulfilling the norm always leads to states in which the value is more fully accomplished. So, each value has attached a list of norms that contribute to that value.

$$\vdash_{IFMAS} D(share(info)) \mapsto O_{IFMAS}(disseminate(research))$$

We define $ANorms$ (the language for abstract norms) to be a deontic logic that is temporal, relativized (in terms of roles and groups) and conditional, i.e., an obligation to perform an action or reach a state can be conditional on some state of affairs to hold, it is also meant for a certain type (or role) of agents and should be fulfilled before a certain point in time. For instance, the following norm might hold: *"The authors should submit their contributions before the deadline"*, which can be formalized as:

$$O_{author}(submit(paper) < Deadline)$$

The obligation is directed towards the author, assuming that she is responsible for fulfilling it.

5.2. The Normative Concrete Level

In order to check norms and act on possible violations of the norms by the agents within an organization, the abstract norms have to be translated into actions and concepts that can be handled within such organization. To do so, the definition of the abstract norms are iteratively concretized into more concrete norms, and then translated into the rules, violations and sanctions that implement them.

5.2.1. The Norm Level The norms at this level are described in $CNorms$ (the language for concrete norms), which we assume for the moment to be equal to $ANorms$, but which might use different predicates. In addition we define a function $\mathtt{I}: ANorms \rightarrow CNorms$ which is a mapping from the abstract norms to the concrete ones. For each abstract norm $\mathtt{I}$ indicates how it can be fulfilled by fulfilling concrete norms within the context of this organization. This function is based on the counts-as operator as developed in [8].

There are several ways in which norms can be *abstract* and thus several ways to make them more concrete [17]. As an illustration of this process, in the following we describe two kinds of abstractness.

Abstract actions: Actions that can be implemented in many ways. For example: *"submitting a paper"*. The translation in this case is a kind of definition of the abstract action in terms of concrete ones:

$$send\_mail(organizer, files) \cup send\_post(organizer, hard\_copies)$$
$$\Rightarrow_{IFMAS} submit(paper)$$

Temporal abstractness: Often there is an implicit deadline for obligations. E.g., the obligation of reviewing the paper occurs only if the paper is assigned, and if so the review should be done before the deadline:

$$done(assign\_paper(P,me,Deadline)) \rightarrow$$
$$O_{PC\_member}(review\_paper(P, Rep) < do(pass(Deadline)))$$

5.2.2. The Rule Level The translation from norms to rules in OMNI marks a transition from a normative perspective to a more descriptive one. Such translation also implies a change in the language, from deontic logic to a Propositional Dynamic Logic (a language more suitable to express actions and time constraints). Each norm can be translated into:
a) *a violation expression*: by using the following reduction rule by Meyer [10]:

$$O(\alpha) \mapsto [\neg\alpha] V$$

b) *a precedence expression*: in those cases where the norm expresses temporal relations among actions, such relation can be also expressed through the [ ] operator as follows:

$$O(\alpha < do(\beta)) \mapsto [\beta] done(\alpha)$$
$$O(\alpha < do(\beta)) \mapsto \neg done(\alpha) \rightarrow [\beta] V$$

The first reduction rule translates the temporal constraint of α being done before β with an expression in Dynamic Logic that states: *"once action β is performed, it should always be the case that action α has been done"*. The second reduction rule expresses the violation condition: *"if action α has not been done, once action β is performed it always is the case that violation V occurs"*.

By means of this refinement process, the designer can obtain all the norms and rules that apply in the system, and then include them in the organizational model. An example of the formalization of the role norms introduced in table 1 is given in table 3.

In OMNI, *violations* are the key concept in norm enforcement. We separate the violations coming from the behavior of external entities (*external violations*) from the ones related to the behavior of the internal agents (*internal violations*)

Id	PC_member
Objectives	paper_reviewed(Paper,Report)
Sub-objectives	{read(P), reported(P, Rep), review_received(Org, P, Rep)}
Rights	access-confman-program(me)
Norms	$O_{PC\_member}(understand(English))$ $done(assign\_paper(P,me,Deadline)) \rightarrow$ $\quad O_{PC\_member}(review\_paper(P, Rep) < do(pass(Deadline)))$ $done(assign\_paper(P,me,\_)) \wedge is\_a\_direct\_colleague(author(P)) \rightarrow$ $\quad O_{PC\_member}(review\_refused(P) < pass(TOMORROW))$
Rules	$done(assign\_paper(P,me,Deadline)) \wedge \neg done(review\_paper(P, Rep))$ $\rightarrow [pass(Deadline)]\ V4$ $done(assign\_paper(P,me,\_)) \wedge is\_a\_direct\_colleague(author(P)) \wedge$ $\neg done(review\_refused(P)) \rightarrow [pass(TOMORROW)]\ V5$
Type	external

Table 3. Formal specification of PC member role.

Internal violations describe states that the organization should always avoid. As the designer has full control of the design of the organization's own agents, such internal agents will fully comply with the organizational objectives and follow its norms and rules. In this case the aim is not to create an *enforcement mechanism* but a continuous *safety control* of the system's behavior (i.e., avoid the system to enter in a non-desirable, *illegal* state because of a failure in one of the agents).

In our framework, *external violations* are the ones where the designer should pay more attention. As we cannot assume that agents entering into the organization will always follow the norms and rules imposed by its normative system, an active enforcement should be made by the internal agents. In our framework, internal agents do not have access to the internal beliefs, goals and intentions of the other agents, they can only check the agents' behavior, by detecting when those agents enter in states considered *illegal*. The way of doing so is by means of the list of definitions of external violations. Such list defines, for each violation, the condition that triggers it. This condition is extracted from the rule that defines the violation. As an example, let us take one of the rules identified for the PC member role in table 3:

$$done(assign\_paper(P,me,Deadline))$$
$$\wedge \neg done(review\_paper(P,Rep)) \rightarrow [pass(Deadline)]\ V4$$

we can create the condition for violation $V4$ by stating that the action inside [ ] has been *done* while the other preconditions are true. Then we should also add the *sanction* (the actions carried against the violator), the *side-effects* (the actions to be done to counter-act the violation) and the *enforcing roles* (the role or roles

that have the responsibility to detect this type of violations):

Violation:	$IFMASV4$
Pre-conditions:	$done(assign\_paper(P,me,Deadline))$
	$\wedge \neg\ done(review\_paper(P,\ Rep)) \wedge done(pass(Deadline))$
Sanction:	$delete\_from\_PC\_list(me))$
Side-effects:	$\{\ find\_new\_reviewer(P,r2);\ assign\_paper(P,r2,Deadline2)\ \}$
Enforcing roles:	$\{\ organizer,\ session\_chair\}$

5.3. The Normative Implementation level

There are two main approaches to implement the rules in the rule level: a) creating a *rule interpreter* that any agent entering the organization will incorporate, and b) translating the rules into *protocols* to be included in the interaction contracts

Note that in both cases it is not ensured that the agents will follow those descriptions. The violations in the rule level should also be translated in some detection mechanisms to check the behavior of the agents.
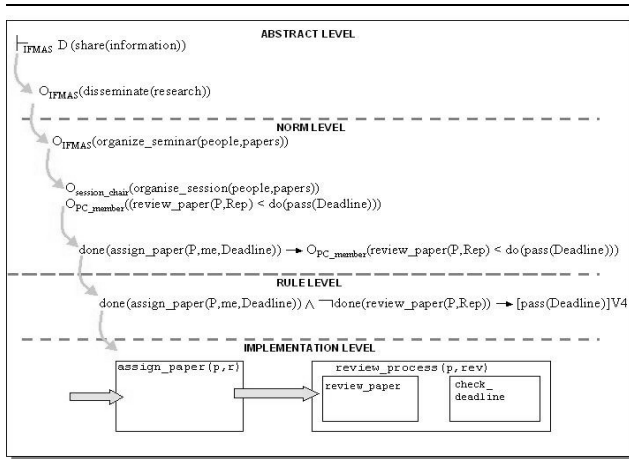

Figure 5. An example of the refinement process.

At Implementation Level, the organization model provides both the low-level protocols and the related rules. Agents that are only able to follow protocols will blindly follow them, while the ones that can also interpret the rules (*Deliberative Normative Agents* [4]) can choose between following the protocol or reasoning about the rules, or do both. With this approach the autonomy of the agents entering the organization is adapted to their reasoning capabilities, which makes OMNI utmost adequate to model open environments. *Norm Autonomous Agents* that are able to reason about norms, can switch from following low-level protocols to higher level rules and norms, by using the

links provided by OMNI from procedures to rules, and from rules to norms. An example is shown in figure 5.

6. The Ontological Dimension

The main challenge of coordination and collaboration in open environments is that of mutual understanding. Communication mechanisms include both the representation of domain knowledge (*what* are we talking about) and protocols for communication (*how* are we talking). Both content and protocol have different meanings at the different levels of abstraction (e.g. while at the abstract level one might talk of *disseminate*, such action will most probably not be available to agents acting at the implementation level). Specification of communication content is usually realized using ontologies, which are shared conceptualizations of the terms and predicates in a domain. Agent communication languages (ACLs) are the usual means in MAS to describe communicative actions. ACLs are wrapper languages in the sense that they abstract from the content of communication.

In OMNI, the Ontological Dimension describes both the content and the language for communication, at three different levels of abstraction. At the Abstract Level, the *Model Ontology* can be seen as a meta-ontology that defines all the concepts of the framework itself, such as norms, rules, roles, groups, violations, sanctions and landmarks.

The content aspects of communication, or domain knowledge, are specified by *Domain Ontologies*. In OMNI abstract concepts can be iteratively defined and refined in terms of more concrete concepts. The Concrete Domain Ontology includes all the predicates and elements that appear during the design of the Organizational and Normative Structure, and the Procedural Domain Ontology, with the terms from the domain that will be finally used in the implemented system. Concepts or predicates at a lower abstraction level, count as, or implement, concepts at the higher levels. For instance, the actual realization of the AAMAS'04 conference *counts-as* the IFMAS's objective *organize-conference* defined in the Organizational Model, which in turn *counts-as* the IFMAS's value of disseminate knowledge, described in its statutes.

Communication Acts define the language for communication, including the performatives and the protocols. At the Concrete Level, *Generic Communication Acts* define the interactions languages used in the Organizational Model, while the *Specific Communication Acts* covers the communication languages actually used by the agents as they agree in the interaction contracts. As with the content ontologies, communicative acts de-

fined at a lower level of abstraction implement those defined at a higher level.

7. Discussion: OMNI compared with other approaches

Development methods for multi-agent systems are currently a hot research topic and several approaches have been proposed. Comprehensive methodologies to design agent societies must be able to describe the characteristics of organizational environments. Such environments are best understood in terms of social concepts such as organization structures, norms and domain language. Furthermore, methodologies must support the development of open societies and the specification of formal institutions. These are issues covered by the OMNI framework. In the remainder of this section, we briefly discuss how some well known models support the social and normative concepts introduced by the OMNI framework.

GAIA Gaia [18] is one of the first agent-oriented software engineering methodologies that explicitly takes into account social concepts. Gaia models are situated in at the Concrete Level of OMNI (cf. figure 1). While the Implementation Level is explicitly and purposefully ignored, Gaia does not have any notion of an Abstract Level. Gaia is only concerned with the society level and does not capture internal aspects of agent design. However, societies are only considered from the perspective of the individual participants, and therefore Gaia does not deal with communication or other collective issues. Furthermore, normative aspects are reduced to static permissions, a sort of constraints or rules and behavior is fixed in protocols. Moreover, Gaia is not suited to model open domains, and cannot easily deal with self-interested agents, as it does not distinguish between organizational and individual aspects, and does not provide capabilities for agent interpretation of society objectives, norms or plans.

SODA SODA [11], is actually an extension to Gaia that enables open societies to be designed around suitably-designed coordination media, and social rules to be designed and enforced in terms of coordination rules. As Gaia, SODA distinguishes between an analysis and a design phase. As an attempt to include an higher abstraction level (cf. figure 1), SODA presents a notion of the context, or environment, of the society, albeit not explicit. However, even though SODA distinguishes between agent and collective spaces, it sees roles as the representation of the observable behavior of agents, and therefore cannot represent the difference between the organizational perspective on the activity

and aims of individuals (represented by the concept of role in OMNI) from the agent perspective on its own activity and aims (represented by the concept of agent in OMNI and linked to the role by a social contract). Role enactment is fixed in SODA as the agent model that maps roles to agent classes without any possibility to accommodate agent preferences or characteristics (agent classes are pure specifications of the role characteristics). There are no normative aspects in SODA further than the notion of permission to access infrastructure services. Communication primitives are limited to interaction protocols, and SODA provides no explicit representation for the domain ontology. Furthermore, SODA also does not have a clear and formal semantics.

ISLANDER The ISLANDER formalism [7] provides a formal framework for institutions [14] and has proven to be well-suited to model practical applications (e.g. electronic auction houses). This formalism views an agent-based institution as a *dialogical system* where all the interactions inside the institution are a composition of multiple dialogic activities (message exchanges). Furthermore, the e-INSTITUTOR platform and the ISLANDER API enable the animation of models and the participation of external agents. The activity of these agents is, however, constrained by *governors* that regulate agent actions, to the precise enactment of the roles specified in the institution model. In contrary to the other frameworks discussed here, ISLANDER provides a sound model for the domain ontology and has a formal semantics [14]. However, ISLANDER provides no primitives to model the Abstract Level of an organization and does not consider the normative aspects of organizations, further than the specification of constraints for scene transition and enactment (the only allowed interactions are those explicitly represented by arcs in scenes).

TROPOS TROPOS methodology [2] spans the overall development process. It distinguishes between an early and a late requirements phase, and between architectural design and detailed design. The models are implemented using Jack Intelligent Agents [9], which is an agent-oriented extension of Java. Tropos is a fairly complete methodology that considers all steps in system development, and it treats both inter-agent and intra-agent perspectives. The early requirement phase of Tropos, can be seen as a specification of the Abstract Level proposed by OMNI (cf. figure 1). The late requirements phase comes fairly close to the idea of Concrete Level in OMNI , except that it does not provide explicit concepts to capture norms, and ontological aspects are only implicitly described. At the Im-

plementation Level, Tropos provides a detailed implementation of organizational models into JACK agents. The main two shortcomings of Tropos are that a) it is not formal (although there is some ongoing work on providing a formal semantics for Tropos), and b) it is too organizational-centered in the sense that is does not consider that agents can have their own goals and plans, and not just those coming from the organization. Furthermore, Tropos has no concept representing the normative aspects of an organization.

Conclusions

In this paper we introduced OMNI, a modelling framework for different types of MAS, from closed systems to open, flexible environments. This approach is rather unique, as most existing frameworks concentrate on a specific type of MAS. The modular structure of OMNI facilitates the adaptation of the framework to different types of domains. In those domains with none or small normative components, design is guided by the Organizational Dimension, while in highly regulated domains the Normative Dimension is more prominent and therefore guides the design.

All dimensions of OMNI have a formal logical semantics, which ensures consistency and possibility of verification of the different aspects of the system.

Currently we are taking the first steps towards implementing tools to use with the framework. We will be using ISLANDER as a basis for the support of the implementation level and build the other levels on top of that.

References

[1] G. Abdelkader. Requirements for achieving software agents autonomy and defining their responsibility. In *Proc. Autonomy Workshop at AAMAS 2003*.

[2] P. Bresciani, P. Giorgini, F. Giunchiglia, J. Mylopoulos, and A. Perini. TROPOS: An agent-oriented software development methodology. *Journal of Autonomous Agents and Multi-Agent Systems*, to appear.

[3] M. Dastani, V. Dignum, and F. Dignum. Role assignment in open agent societies. In *Proc. AAMAS'03*, 2003.

[4] F. Dignum. Autonomous Agents with Norms. *AI and Law*, 7:69–79, 1999.

[5] V. Dignum. *A Model for Organizational Interaction: based on Agents, founded in Logic*. SIKS Dissertation Series 2004-1. SIKS, 2004. PhD Thesis.

[6] V. Dignum and F. Dignum. Modeling agent societies: Coordination frameworks and institutions. In P. Brazdil and A. Jorge, editors, *Progress in Artificial Intelligence*, LNAI 2258, pages 191–204. Springer-Verlag, 2001.

[7] M. Esteva, J. Padget, and C. Sierra. Formalizing a language for institutions and norms. In J.-J.CH. Meyer and M. Tambe, editors, *Intelligent Agents VIII*, volume 2333 of *LNAI*, pages 348–366. Springer Verlag, 2001.

[8] D. Grossi and F. Dignum. Abstract and concrete norms in institutions. *Accepted in FAABS 2004*, 2004.

[9] Nick Howden, Ralph Rnnquist, Andrew Hodgson, and Andrew Lucas. Jack - summary of an agent infrastructure. In *5th International Conference on Autonomous Agents*. 2001.

[10] J.-J. Ch. Meyer. A different approach to deontic logic: Deontic logic viewed as a variant of dynamic logic. *Notre Dame J. of Formal Logic*, 29(1):109–136, 1988.

[11] A. Omicini. Soda: Societies and infrastructures in the analysis and design of agent-based systems. In P. Ciancarini and M. Wooldridge, editors, *Agent-Oriented Software Engineering*, volume 1957 of *LNAI*, pages 185–193. Springer Verlag, 2001.

[12] A. Omicini. Towards a notion of agent coordination context. In D. Marinescu and C. Lee, editors, *Process Coordination and Ubiquitous Computing*, pages 187–200. CRC-Press, 2002.

[13] H.V.D. Parunak and J. Odell. Representing social structures in uml. In M.Wooldridge, G.Weiss, and P. Ciancarini, editors, *Agent-Oriented Software Engineering II*, LNCS 2222. Springer-Verlag, 2002.

[14] J.A. Rodriguez. *On the Design and Construction of Agent-mediated Electronic Institutions*. PhD thesis, Institut d'Investigació en Intel.ligència Artificial (IIIA), 2001.

[15] I. Smith, P. Cohen, J. Bradshaw, M. Greaves, and H. Holmback. Designing conversation policies using joint intention theory. In *Proc. ICMAS-98*, pages 269–276. IEEE Press, 1998.

[16] W. Vasconcelos, J. Sabater, C. Sierra, and J. Querol. Skeleton-based agent development for electronic institutions. In *Proc. AAMAS'02*, 2003.

[17] J. Vázquez-Salceda and F. Dignum. Modelling electronic organizations. In V. Marik, J. Muller, and M. Pechoucek, editors, *Multi-Agent Systems and Applications III*, LNAI 2691, pages 584–593. Springer-Verlag, 2003.

[18] M. Wooldridge, N.R. Jennings, and D. Kinny. The Gaia Methodology for Agent-Oriented Analysis and Design. *Autonomous Agents and Multi-Agent Systems*, 3(3):285–312, 2000.

[19] F. Zambonelli. Abstractions and infrastructures for the design and development of mobile agent organizations. In M.Wooldridge, G.Weiss, and P. Ciancarini, editors, *Agent-Oriented Software Engineering II*, LNCS 2222, pages 245–262. Springer-Verlag, 2002.

[20] F. Zambonelli, N. Jennings, and M. Wooldridge. Organisational abstractions for the analysis and design of multi-agent systems. In P. Ciancarini and M.Wooldridge, editors, *Agent-Oriented Software Engineering*, LNCS 1957, pages 98–114. Springer-Verlag, 2001.