Operationalisation of Norms for Usage in Electronic Institutions*

Huib Aldewereld[†]
Frank Dignum
Institute of Information and Computing Sciences
Utrecht University
The Netherlands
{huib, dignum}@cs.uu.nl

Andres Garcˇa-Camino[‡]
Pablo Noriega
Juan Antonio Rodrˇguez-Aguilar
Carles Sierra
Artificial Intelligence Research Institute, IIIA
Spanish Council for Scientific Research, CSIC
Campus de la UAB, Barcelona, Spain
{andres, pablo, jar, sierra}@iiia.csic.es

ABSTRACT

Agent-mediated electronic institutions belong to a new and promising field where interactions between a group of agents are regulated by means of a set of explicit norms. Current implementations of such open-agent systems are, however, mostly using constraints on the behaviour of the agents, thereby severely limiting the autonomy of the agents. To increase the autonomy of agents and possibly boost the efficiency of the overall system, a more flexible norm enforcement is required. However, as norms make extensive use of vague and ambiguous concepts and lack operational meaning (not expressing how the norm should be enforced), translating norms for usage with such a flexible enforcement mechanism might be difficult. In this paper we propose an extension to electronic institutions to allow for a flexible enforcement of norms, and manners to help overcome the difficulties of translating abstract norms for the use of implementation.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*multiagent systems*

General Terms

Theory, Legal Aspects, Design

Keywords

Norms, Electronic Institutions, Normative Systems

1. INTRODUCTION

Agent-mediated institutions, introduced in [?, ?], are open agent systems that allow heterogeneous agents to enter and perform tasks. Because of this heterogeneous nature of the agents joining the electronic institution (e-institution), measures have to be taken to control and regulate the behaviour of these agents. These measures are needed to improve and guarantee the safety and stability of the system, as agents joining the institution might, (un)intentionally, brake the system by behaving in non-expected or non-accepted manners. It has been widely accepted that norms can be used to ensure this safety, since norms, which are vague and abstract in order to express various different circumstances without the need for change, can be used for defining the legality and illegality of actions (and states) in e-institutions [?].

For these norms to be used in the e-institutions, thereby regulating the agents joining the institution, enforcement mechanisms must be devised to implement the norms in the institution. Although the norms should be available to the agents joining the institution to allow them to work more efficiently in the regulated domain, it is not the agents reasoning and trying to adhere to the norms that provide the safety that the institution needs; it is the institution itself that has to ensure that this safety exists. As discussed in [?], the enforcement of norms comes down to either: 1) defining constraints on unwanted behaviour, or 2) detecting violations of norms and reacting to these violations. The former manner is, however seriously reducing the autonomy of agents, used by current implementations of e-institutions [?]. However, to allow the agents in e-institutions more freedom and flexibility, while still complying to the norms, we would like to extend the implementation of e-institutions with the second manner of enforcing norms.

Previous work on normative systems (mainly focussed on deontic frameworks [?, ?]) is mostly *declarative in nature*, while the implementation of norms and norm enforcement in e-institutions, as mentioned above, require norms to have an *operational semantics* as well. Where the declarative nature of norms is necessary for reasoning about norms (reasoning about what is and what is not accepted), the operational semantics define how norms are to be implemented (e.g. what

[*] This is a student paper.

[†] The first author of this paper was supported by the Netherlands Organisation for Scientific Research (NWO) under project number 634.000.017

[‡] A. García-Camino was supported by the Spanish Council for Scientific Research (CSIC) by means of an I3P grant.

to do when norms are violated). Recent approaches on normative systems have begun to research and express this operational meaning of norms, as seen in [**?**, **?**, **?**, **?**]. These approaches represent norms and their operational meaning, but are not conclusive on how the implementation in an agent system, such as an e-institution, should be obtained. In this paper we are trying to bridge this gap, by proposing a translation from the operational approach proposed in [**?**] to elements usable for norm enforcement in AMELI. Moreover we will show that the approaches from [**?**] and [**?**] can be translated in this formalism as well.

In this paper we assume institutions to be defined as a set of norms, which are to be enforced by a distributed set of (internal) agents. Secondly we assume that the norms can sometimes be violated by agents in order to keep their autonomy, which can also be functional for the system as a whole as argued in [**?**]. The violation of norms is handled from the organisational point of view by violation and sanction mechanisms. And finally, we assume that the internal state of agents is neither visible, nor controllable from an institution's point of view, which, basically, means that enforcement of norms needs to be done by the detection of violations and the reacting to these violations, and that we can only use the observable behaviour of agents to detect the violations.

The remainder of this paper is organised as follows. In the next section we give a short discussion on a formal view of electronic institutions. In sections 3 and 4 we introduce the syntax and semantics of the mechanism used for expressing and handling the violation of norms, while in section 5 we give a translation from the norm frame of [**?**] into this enforcement mechanism. In section 6 we give a tentative comparison on how this enforcement method can be applied to other normative approaches, and in section 7 we explain how norms can be operationalised before being implemented.

2. ELECTRONIC INSTITUTIONS

Electronic institutions, as we consider them [**?**, **?**, **?**], shape agent environments that restrict the behaviour of agents to ensure that agents interact in safe conditions. E-institutions constrain agent behaviour by defining the valid sequences of (dialogical) interactions that agents can have to attain their goals. We differentiate two types of norms in e-institutions: protocol-based and rule-based. Protocol-based norms are defined by a group of scenes, a performative structure, and a dialogical framework that establish the permitted actions at each instant of time taking into account the past actions of agents. Rule-based norms are defined by a certain type of first-order formulas that establish a dependency relation between actions. Some actions under certain conditions fire normative rules which produce new commitments, establishing new pending obligations (actions to be carried out by agents).

The dialogical framework defines all the conventions required to make interaction between two or more agents possible. Moreover, it defines what the participant roles within the e-institution and the relationships among them will be. We take interactions to be a sequence of speech acts between two or more parties. Formally, we express speech acts as illocutionary formulas of the form: $\iota(speaker, hearer, \phi, t)$. The speech acts that we use start with an illocutionary particle (declare, request, promise) that a *speaker* addresses to a *hearer*, at time t, whose content ϕ is expressed in some object language whose vocabulary stems from an e-institution's ontology.

A *dialogical framework* encompasses all the illocutions available to the agents in a given institution. Formally,

Definition 1. *A dialogical framework is a tuple* $DF = \langle O, L_O, P, R, R_S\rangle$ *where* O *stands for an ontology (vocabulary);* L_O *stands for a content language to express the information exchanged between agents using ontology* O*;* P *is the set of illocutionary particles;* R *is the set of roles;* R_S *is the set of relationships over roles.*

For each activity in an institution, interactions between agents are articulated through agent group meetings, which we call *scenes*. A scene is a role-based multi-agent protocol specification. A scene defines the valid sequences of interactions among agents enacting different roles. It is defined as a directed graph where each node stands for scene state and each edge connecting two states is labelled by an illocution scheme. An illocution scheme is an illocutionary formula with some unbound variables. At run-time, agents playing different roles make a scene evolve by uttering illocutions that match the illocution schemes connecting states. Each scene maintains the context of the interaction, that is how the dialogue is evolving, i.e. which have been the uttered illocutions and how the illocution schemes have been instantiated.

Definition 2. *A scene is a tuple* $S = \langle s, R, DF, W, w_0, W_f, \Theta, \lambda, min, Max\rangle$ *where* s *is the scene identifier;* R *is the set of scene roles;* DF *is a dialogical framework;* W *is the set of scene states;* $w_0 \in W$ *is the initial state;* $W_f \subseteq W$ *is the set of final states;* $\theta \subseteq W \times W$ *is a set of directed edges;* $\lambda : \theta \longrightarrow \mathcal{L}^*_{DF}$ *is a labelling function, which maps each edge to an illocution scheme in the pattern language of the DF dialogical framework* $\mathcal{L}^*_{DF}$*;* $min, Max; \mathbb{R} \longrightarrow \mathbb{N}$ $min(r)$ *and* $Max(r)$ *are, respectively, the minimum and the maximum number of agents that must and can play each role* $r \in R$*.*

The activities in an e-institution are the composition of multiple, distinct, possibly concurrent, dialogical activities, each one involving different groups of agents playing different roles. A performative structure can be seen as a network of scenes, whose connections are mediated by transitions (a special type of scene), and determines the role-flow policy among the different scenes by showing how agents, depending on their roles and prevailing commitments, may get into different scenes, and showing when new scenes will be started. The performative structure defines the possible order of execution of the interaction protocols (*scenes*). It also allows agent synchronisation, and scene interleaved execution.

Definition 3. *A performative structure is a tuple* $PS = \langle S, T, s_0, s_\Omega, E, f_L, f_T, f^O_E, \mu\rangle$ *where* S *is a finite, non-empty set of scenes;* T *is a finite, non-empty set of transitions;* $s_0 \in S$ *is the initial scene;* $s_\Omega \in S$ *is the final scene;* $E = E^I \cup E^O$ *is a set of edge identifiers where* $E^I \subseteq S \times T$ *is a set of edges from scenes to transitions and* $E^O \subseteq T \times S$ *is a set of edges from transitions to scenes;* $f_L : E \longrightarrow DNF_{2^{V_A \times R}}$ *maps each edge to a disjunctive normal form of pairs of agent variable and role identifier representing the edge label;* $f_T : T \longrightarrow \mathcal{T}$ *maps each transition to its type;* $f^O_E : E^O \longrightarrow \mathcal{E}$ *maps each edge to its type;* $\mu : S \longrightarrow \{0, 1\}$ *sets if a scene can be multiply instantiated at execution time;*

The institutional state consists of the list of scene executions (described by their participating agents and interaction context) along with the participating agents' state (represented by their observable attributes).

3. INTEGRITY & DIALOGICAL CONSTRAINTS

As mentioned in the introduction, we want to extend the ISLANDER formalism with mechanisms to implement norms by means of a distributed set of agents. To achieve this we need mechanisms to detect violations and react to these violations. This is accomplished by using, respectively, integrity constraints and dialogical constraints. The main idea is that integrity constraints are checked by the institution to detect and register all violations, i.e. the passing from a legal state to an illegal state. Besides that, the dialogical constraints express the obligation of the enforcing agents to act according to the violations detected, i.e. sanction the responsible agent. The dialogical constraints themselves are part of the internal enforcing agents.

Due to the fact that the internal agents should be designed to follow the norms of the institution, we might assume that internal agents will always act according to the dialogical constraints specified. However, the internal agents might not be responsible for the enforcement of all the norms in the system, we can specify integrity constraints that express when a dialogical constraint (which is in a sense an obligation to enforce) has been violated, i.e. a violation has occurred, but no action has been taken by the enforcing agent to punish the violator. In theory, complex hierarchical structures of enforcement chains (institutions enforcing the enforcement within another institution, etc.) are possible with the approach presented in this paper, but we are not going to discuss them in this paper.

Before enforcement can take place, norm violations have to be detected. This is done by specifying integrity constraints, extracted from previous work [**?**]:

Definition 4. *Integrity constraints are first-order formulas of the form*

$$\left(\bigwedge_{i=1}^{n} uttered(s_i, w_{k_i}, \boldsymbol{i}_{l_i}) \wedge \bigwedge_{j=0}^{m} e_j \right) \rightarrow \bot$$

where s_i are scene identifiers or variables, w_{k_i} is a state k_i of scene s_i or a variable, $\boldsymbol{i}_{l_i}$ is an illocution scheme l_i matching the schema labelling an outgoing arc from w_{k_i} and e_j are boolean expressions over variables from uttered predicates.

These integrity constraints define sets of states that *should not* occur within an e-institution. The meaning of these constraints is that if grounded illocutions matching the illocution schemes $\boldsymbol{i}_{l_1}, \ldots, \boldsymbol{i}_{l_n}$ are uttered in the corresponding scenes states, and expressions $e_1, \ldots, e_m$ are satisfied, then a violation occurs ($\bot$).

Since agents can violate norms, the integrity constraints are not enough. We need to specify which actions are to be taken by the enforcers after the violation has been detected. In a sense, the violation of a norm by agents within the e-institution obliges the enforcers to perform actions, namely to punish the agent breaking the norm. This "obligation to enforce" is expressed by means of a dialogical constraint:

Definition 5. *Dialogical constraints are first-order formulas of the form:*

$$\left(\bigwedge_{i=1}^{n} uttered(s_i, w_{k_i}, \boldsymbol{i}^*_{l_i}) \wedge \bigwedge_{j=0}^{m} e_j \right) \Rightarrow$$

$$\left(\bigwedge_{i=1}^{n'} uttered(s'_i, w'_{k_i}, \boldsymbol{i}'^*_{l_i}) \wedge \bigwedge_{j=0}^{m'} e_j \right)$$

where s_i, s'_i are scene identifiers or variables, w_{k_i}, w'_{k_i} are variables or states of scenes s_i and s'_i respectively, $\boldsymbol{i}^*_{l_i}$, $\boldsymbol{i}'^*_{l_i}$ are illocution schemes l_i matching the schema labelling an outgoing arc from w_{k_i} of scenes s_i and s'_i respectively, and e_j, e'_j are boolean expressions over variables from uttered predicates. These boolean expressions can include functions to check the state of the institution.

The intuitive meaning of a dialogical constraint is that if grounded illocutions matching $\boldsymbol{i}^*_{l_1}, \ldots, \boldsymbol{i}^*_{l_n}$ are uttered in the corresponding scene states, and the expressions $e_1, \ldots, e_m$ are satisfied, then, grounded illocutions matching $\boldsymbol{i}'^*_{l_1}, \ldots, \boldsymbol{i}'^*_{l_n}$ satisfying the expressions $e'_1, \ldots, e'_{m'}$ must be uttered in the corresponding scene states as well. Dialogical constraints assume a temporal ordering: the left-hand side illocutions must be uttered prior to the illocutions on the right-hand side, i.e. the illocutions on the left should have time stamps which precede those of the illocutions on the right.

The dialogical constraints point out the actions to perform in the enforcement of a violated norm. For instance,

$$uttered(S,W,inform(A,Role,all,Role2,smoke(),T)) \Rightarrow$$
$$uttered(S,W,inform(B,enforcer,A,Role,decrement(credit,50),T'))$$

shows an example of a dialogical constraint which expresses that every agent that smokes in a scene should be sanctioned (since smoking is illegal). Whenever an agent performs the action of smoke, an enforcer agent is obliged to decrement its credit by 50.

The integrity constraints are then implemented in the infrastructure of the e-institutions, thereby providing the means to detect violations of norms, where the dialogical constraints are implemented in the enforcing agents which use them to determine the illocutions that should be uttered when a norm has been violated.

4. SEMANTICS

In this section we present the semantics of the integrity constraints, used for detecting violations, and the dialogical constraints, used for specifying enforcement, which we introduced in the previous section. In the definitions below we rely on the concept of *substitution*, that is, the set of values (first-order terms denoted τ) for variables (denoted x, y, z) in a computation [**?**, **?**]:

Definition 6. *A substitution $\sigma = \{x_0/\tau_0, \ldots, x_n/\tau_n\}$ is a finite and possibly empty set of pairs x_i/τ_i, $0 \leq i \leq n$, $n \in I\!N$.*

Definition 7. *The application of a substitution follows:*
1. $c \cdot \sigma = c$ for a constant c;
2. $x \cdot \sigma = \tau \cdot \sigma$ if $x/\tau \in \sigma$; otherwise then $x \cdot \sigma = x$;
3. $p^n(\tau_0, \ldots, \tau_n) \cdot \sigma = p^n(\tau_0 \cdot \sigma, \ldots, \tau_n \cdot \sigma)$.

We conceive the notion of state in an electronic institution as the set of illocutions uttered and the boolean expressions that hold during its enactment. The execution of the institution would be divided into two different, alternating rounds: event addition and processing. Firstly, we start the execution with a (possibly empty) initial state where agents' illocutions are added. Secondly, the rules are executed evolving the state adding inconsistency marks or obligations. Then, we again start the event addition round and so on. The semantics of the integrity constraints are defined as relationships ($\mathbf{s}_{IC}$) between the current state Δ and the next state Δ'. Let us first look at the utterances and boolean expressions that are used in the constraints. An utterance holds iff it is uttered in the current state:

Definition 8. $\mathbf{S}(\Delta, \mathit{uttered}(s, w, \boldsymbol{i}), \sigma)$ *holds iff* $\mathit{uttered}(s \cdot \sigma, w \cdot \sigma, \boldsymbol{i} \cdot \sigma) \in \Delta$

Conjunctions used in the constraints are satisfied in the normal method:

Definition 9. $\mathbf{S}(\Delta, (\bigwedge_{i=1}^{n} \tau_i), \sigma)$ *holds iff* $\mathbf{S}(\Delta, \tau_i, \sigma)$, $1 \leq i \leq n, n \in \mathbb{N}$, *hold.*

We now define when boolean expressions hold:

Definition 10. $\mathbf{S}(\Delta, \tau_1 \rhd \tau_2, \sigma)$ *holds iff* $\tau_1 \cdot \sigma \rhd \tau_2 \cdot \sigma$ *holds. Where* $\rhd \in \{=, \neq, >, <, \geq, \leq\}$ *with their usual meaning.*

Integrity constraints define the violations of the norms. An integrity constraint is applicable to the institutional state (Δ), and thus introducing a violation ($\perp$), iff the conjunction of utterances and boolean expressions holds in Δ:

Definition 11. $\mathbf{s}_{IC}(\Delta, ((\bigwedge_{i=1}^{n} \mathit{uttered}(s_i, w_{k_i}, \boldsymbol{i}_{l_i}) \wedge \bigwedge_{j=0}^{m} e_j) \rightarrow \perp), \Delta \cup \{\perp\})$ *holds iff* $\mathbf{S}(\Delta, (\bigwedge_{i=1}^{n} \mathit{uttered}(s_i, w_{k_i}, \boldsymbol{i}_{l_i})), \{\sigma_1, \ldots, \sigma_p\})$ *and* $\mathbf{S}(\Delta, (\bigwedge_{j=0}^{m} e_j), \{\sigma_1, \ldots, \sigma_p\})$, $1 \leq i \leq n, 0 \leq j \leq m, n, m \in \mathbb{N}$, *hold.*

An integrity constraint does not introduce a violation, if either the utterances or the boolean expressions does not hold in Δ, i.e. the integrity constraint is not applicable:

Definition 12. $\mathbf{s}_{IC}(\Delta, ((\bigwedge_{i=1}^{n} \mathit{uttered}(s_i, w_{k_i}, \boldsymbol{i}_{l_i}) \wedge \bigwedge_{j=0}^{m} e_j) \rightarrow \perp), \Delta)$ *holds iff* $\mathbf{S}(\Delta, (\bigwedge_{i=1}^{n} \mathit{uttered}(s_i, w_{k_i}, \boldsymbol{i}_{l_i})), \{\sigma_1, \ldots, \sigma_p\})$, $1 \leq i \leq n, n \in \mathbb{N}$, *does not hold or* $\mathbf{S}(\Delta, (\bigwedge_{j=0}^{m} e_j), \{\sigma_1, \ldots, \sigma_p\})$, $0 \leq j \leq m, m \in \mathbb{N}$, *does not hold.*

Dialogical constraints introduce obligations to enforce, based on the violations detected by integrity constraints. We define the semantics of dialogical constraints as relationships ($\mathbf{s}_{DC}$) between current state Δ and the next state Δ'. A dialogical constraint is applicable to a state Δ, thus introducing an obligation to enforce, iff the conjunction of utterances and boolean expressions holds in Δ:

Definition 13. $\mathbf{s}_{DC}(\Delta, ((\bigwedge_{i=1}^{n} \mathit{uttered}(s_i, w_{k_i}, \boldsymbol{i}^{*}_{l_i}) \wedge \bigwedge_{j=0}^{m} e_j) \Rightarrow (\bigwedge_{i=1}^{n'} \mathit{uttered}(s'_i, w'_{k_i}, \boldsymbol{i}'^{*}_{l_i}) \wedge \bigwedge_{j=0}^{m'} e_j)), \Delta \cup \{ \bigwedge_{i=1}^{n'} \mathit{uttered}(s'_i, w'_{k_i}, \boldsymbol{i}'^{*}_{l_i}) \wedge \bigwedge_{j=0}^{m'} e_j)\}\})$ *holds iff* $\mathbf{S}(\Delta, (\bigwedge_{i=1}^{n} \mathit{uttered}(s_i, w_{k_i}, \boldsymbol{i}_{l_i})), \{\sigma_1, \ldots, \sigma_p\})$ *and* $\mathbf{S}(\Delta, (\bigwedge_{j=0}^{m} e_j), \{\sigma_1, \ldots, \sigma_p\})$, $1 \leq i \leq n, 0 \leq j \leq m$, $n, m \in \mathbb{N}$, *hold.*

A dialogical constraint does not introduce an obligation to enforce iff the conjunction of utterances or the conjunction of boolean expression does not hold in Δ:

Definition 14. $\mathbf{s}_{DC}(\Delta, ((\bigwedge_{i=1}^{n} \mathit{uttered}(s_i, w_{k_i}, \boldsymbol{i}^{*}_{l_i}) \wedge \bigwedge_{j=0}^{m} e_j) \Rightarrow (\bigwedge_{i=1}^{n'} \mathit{uttered}(s'_i, w'_{k_i}, \boldsymbol{i}'^{*}_{l_i}) \wedge \bigwedge_{j=0}^{m'} e_j)), \Delta)$ *holds iff* $\mathbf{S}(\Delta, (\bigwedge_{i=1}^{n} \mathit{uttered}(s_i, w_{k_i}, \boldsymbol{i}_{l_i})), \{\sigma_1, \ldots, \sigma_p\})$, $1 \leq i \leq n$, $n \in \mathbb{N}$ *does not hold or* $\mathbf{S}(\Delta, (\bigwedge_{j=0}^{m} e_j), \{\sigma_1, \ldots, \sigma_p\})$, $0 \leq j \leq m, m \in \mathbb{N}$, *does not hold.*

From this semantics we can straightforwardly implement an interpreter in Prolog as done in [**?**]. This interpreter would evolve the state of enactment of an institution by adding inconsistency marks, based on violations detected through the integrity constraints, or obligations to enforce, based on the specified dialogical constraints.

In the current AMELI framework, agent interactions are mediated by a special kind of agents called *governors*. These governors regulate the agents' illocutions following the specification of electronic institutions, i.e. they only forward illocutions that match the illocution scheme of an outgoing arc of the current state of the scene. By including the interpreter mentioned above, we improve the governors by allowing them to regulate more expressive and flexible specifications of electronic institution.

The semantics given above provide a basis for the implementation of our interpreter for integrity and dialogical constraints. We show such an interpreter in figure 1 as a logic program, interspersed with built-in Prolog predicates; for easy referencing, we numbered each of the clauses.

1. $\mathbf{s}(\Delta, \alpha', \sigma) \leftarrow \mathtt{member}(\alpha' \cdot \sigma, \Delta)$
2. $\mathbf{s}(\Delta, (\mathit{LHS} \wedge \mathit{LHS}'), \sigma) \leftarrow \mathbf{s}(\Delta, \mathit{LHS}, \sigma'), \mathbf{s}(\Delta, \mathit{LHS}', \sigma''),$
   $\mathtt{union}(\sigma', \sigma'', \sigma)$
3. $\mathbf{s}(\Delta, e_j, \sigma) \leftarrow \mathtt{call}(e_j \cdot \sigma)$
4. $\mathbf{s}_{IC}(\Delta, LHS, [\perp|\Delta]) \leftarrow \mathbf{s}(\Delta, LHS, \sigma)$
5. $\mathbf{s}_{IC}(\Delta, (LHS \Rightarrow \perp), \Delta) \leftarrow \neg\mathbf{s}(\Delta, LHS, \sigma)$
6. $\mathbf{s}_{DC}(\Delta, (LHS \Rightarrow RHS), [RHS|\Delta]) \leftarrow \mathbf{s}(\Delta, LHS, \sigma)$
7. $\mathbf{s}_{DC}(\Delta, (LHS \Rightarrow RHS), \Delta) \leftarrow \neg\mathbf{s}(\Delta, LHS, \sigma)$

Figure 1: An Interpreter for Integrity and Dialogical Constraints

Clauses 1-7 are, respectively, adaptations of the cases depicted in definitions 8 to 14. Clause 1 makes use of the built-in predicate `member/2` that checks if the first argument is in the list provided as the second argument. Clause 3 makes use of the built-in predicate `call/1` that executes the expression provided as argument.

5. IMPLEMENTING NORMS

The operational approach to norms expressed in [**?**] that tries to implement norms from an institutional perspective, that is to say enforcing norms by means of detecting violations and reacting to such violations, views norms as a manner to describe how someone should behave, i.e., they define obligations, permissions and prohibitions also known as the *declarative meaning of norms* (cf. [**?**, **?**]). Since a system needs responses to the violations that occur, the norms in this approach are viewed as a frame which includes not only this declarative meaning of the norm but also a definition of the responses to violations to the norms, which are known as sanctions and repairs (also known as the *operational meaning of the norm*). In [**?**] this norm frame is defined as follows:

Definition 15 (Norms).

$$\begin{aligned}
\mathsf{NORM} &:= \mathsf{NORM\_CONDITION}\\
&\ \mathsf{VIOLATION\_CONDITION}\\
&\ \mathsf{DETECTION\_MECHANISM}\\
&\ \mathsf{SANCTION}\\
&\ \mathsf{REPAIRS}\\
\mathsf{VIOLATION\_CONDITION} &:= formula\\
\mathsf{DETECTION\_MECHANISM} &:= \{action\ \ expressions\}\\
\mathsf{SANCTION} &:= \mathsf{PLAN}\\
\mathsf{REPAIRS} &:= \mathsf{PLAN}\\
\mathsf{PLAN} &:= action\,expression \mid action\,expression\ ;\mathsf{PLAN}
\end{aligned}$$

The norm condition is the declarative norm, as obtained from, for instance, the legal domain (see definition 16 for a description of what these norm conditions can be. The other fields in this norm description are; 1) the *violation condition* which is a formula defining when the norm is violated, 2) the *detection mechanism* which describes the mechanisms included in the agent platform that can be used for detecting violations, 3) the *sanction* which defines the actions that are used to punish the agent(s) violating the norm, and 4) the *repairs* which is a set of actions that are used for recovering the system after the occurrence of a violation.

Definition 16 (Norm Condition).

$$\begin{aligned}
\mathsf{NORM\_CONDITION} &:= N(a,S\ \langle\mathsf{IF}\ C\rangle) \mid \mathsf{OBLIGED}(a\mathsf{ENFORCE}(N(a,S\ \langle\mathsf{IF}\ C\rangle)))\\
N &:= \mathsf{OBLIGED} \mid \mathsf{PERMITTED} \mid \mathsf{FORBIDDEN}\\
S &:= P \mid \mathsf{DO}\ A \mid P\ \mathsf{TIME}\ D \mid \mathsf{DO}\ A\ \mathsf{TIME}\ D\\
C &:= formula\\
P &:= predicate\\
A &:= action\ \ expression\\
\mathsf{TIME} &:= \mathsf{BEFORE} \mid \mathsf{AFTER}
\end{aligned}$$

As definition 16 shows, norms can either be permissions, obligations or prohibitions. Moreover, norms can be related to actions or to predicates (states). Through the condition (C) and deadline (D), norms can be made applicable to certain situations only (conditions and deadlines are considered optional).

Before we can use norms specified in the formalism described above, we need to translate the abstract and vague predicates and actions into corresponding concrete utterances and scenes that are specified in the definition of the institution. Since the norms specified in the formalism of definitions 15 and 16 are high-level translations of laws and regulations, they tend to be of a high level of abstraction. For instance, norms in this formalism would be expressed as

$$\mathsf{OBLIGED}((buyer\ \mathsf{DO}\ pay(Price,seller))\ \mathsf{IF}\ done(buyer,won(Item,Price)))$$

whereas, in e-institutions, utterances as the following are used:

$$uttered(payment,W,inform(A,buyer,B,payee,pay(Item,Price),T))$$
$$uttered(auction,w2,inform(C,auctioneer,A,buyer,won(Item,Price),T'))$$

The translation necessary for using these highly-abstracted norms in e-institutions can be considered a contextualisation, as the actions (and predicates) used in the abstract norms are linked to their corresponding meaning in the domain of the e-institution. We address this issue in section 7. For now it will suffice to say that some translation from, e.g., $\mathsf{OBLIGED}((a\ \mathsf{DO}\ A)\ \mathsf{IF}\ C)$ into $\mathsf{OBLIGED}(utter(S,W,I)\ \mathsf{IF}\ C)$ can be given, taking into account that the state S and world W of the institution will correspond to the applicable state meant by the norm, and that I is an illocution performed by a to implement action A.

Once the norms are contextualised, we can map them to integrity constraints, as specified in the previous section, which we use to check whether violations occur. This mapping of the contextualised norm conditions to integrity constraints can be done by the use of the following table:

Norm	Translation
$\mathsf{FORBIDDEN}(utter(s,w,\boldsymbol{i}))$	$uttered(s,w,\boldsymbol{i})\rightarrow\perp$
$\mathsf{OBLIGED}(utter(s,w,\boldsymbol{i})\ \mathsf{IF}\ C)$	$(C\wedge\neg uttered(s,w,\boldsymbol{i}))\rightarrow\perp$
$\mathsf{FORBIDDEN}(utter(s,w,\boldsymbol{i})\ \mathsf{IF}\ C)$	$(C\wedge uttered(s,w,\boldsymbol{i}))\rightarrow\perp$
$\mathsf{OBLIGED}(utter(s,w,\boldsymbol{i})\ \mathsf{BEFORE}\ D)$	$(\nexists T{:}uttered(s,w,\boldsymbol{i}(T))\wedge T<D)\rightarrow\perp$
$\mathsf{OBLIGED}(utter(s,w,\boldsymbol{i})\ \mathsf{AFTER}\ D)$	$(\nexists T{:}uttered(s,w,\boldsymbol{i}(T))\wedge T>D)\rightarrow\perp$
$\mathsf{FORBIDDEN}(utter(s,w,\boldsymbol{i})\ \mathsf{BEFORE}\ D)$	$(\exists T{:}uttered(s,w,\boldsymbol{i}(T))\wedge T<D)\rightarrow\perp$
$\mathsf{FORBIDDEN}(utter(s,w,\boldsymbol{i})\ \mathsf{AFTER}\ D)$	$(\exists T{:}uttered(s,w,\boldsymbol{i}(T))\wedge T>D)\rightarrow\perp$

This table indicates how the norm conditions from the framework described above are translated (automatically) into integrity constraints that can be used for checking whether the norms were violated. $\mathsf{OBLIGED}(utter(s,w,\boldsymbol{i})\ \mathsf{IF}\ C)$, for instance, is violated according to this table if in a state C holds but $\boldsymbol{i}$ has not been uttered.

An observant reader should note that permissions are left out of this translation, since permissions cannot be violated, and therefore cannot be specified as an integrity constraint. Unconditional obligations are also not in this table, since these would mean that agents are obliged to utter a certain illocution all the time, which is not meaningful. Likewise, obligations that should be satisfied after a specific point in time are not very useful either, since these can never be violated. This can, however, be adapted by including another deadline before which the obligation has to be fulfilled, which would mean that, in most cases, the obligation should be fulfilled before the institution ends.

Dialogical constraints are then used by the enforcing agents to determine which actions should be performed when a norm is violated. If a norm with sanction S and repairs R (which can be obtained from the norm framework), can be translated to the integrity constraint $IC \rightarrow \perp$, the following rule can be created (automatically) to oblige the enforcers to utter the illocution (DC) that is derived from contextualising S and R:

$$IC \Rightarrow DC$$

Since these dialogical constraints are considered obligations to the enforcers, we need the means to detect that this obligation has been violated, which is when the original norm was violated, but the enforcer did not punish the violating agent.

$$(IC\wedge \sim DC) \rightarrow \perp$$

And, if specified by means of the $\mathsf{ENFORCE}$ construct (thus including sanctions and repairs in the norm description), this violation can itself, by specifying a dialogical constraint, trigger an obligation to act (but now for a different enforcer).

6. OTHER NORMATIVE APPROACHES

In this section we give a tentative comparison between the approach just mentioned and the norm frameworks introduced in [?] and [?]. Given the concepts seemingly in those frameworks we show how we think norms from these frameworks can be implemented using the language given in section 3.

6.1 Norms in Z

In [?] Luck et al. proposed a framework for norms that could be integrated into their multiagent systems. This norm frame is composed of the elements shown in figure 2. Like the framework of the previous section it identifies

$$\begin{array}{|l}
\hline
\textit{Norm} \\
\textit{addressees}, \textit{benificiaries} : \mathbb{P}\,\textit{Agent} \\
\textit{normativegoals}, \textit{rewards}, \textit{punishments}, : \mathbb{P}\,\textit{Goal} \\
\textit{context}, \textit{exceptions} : \textit{EnvState} \\
\hline
\textit{normativegoals} \neq \varnothing; \textit{addressees} \neq \varnothing; \textit{context} \neq \varnothing \\
\textit{context} \cap \textit{exceptions} = \varnothing; \textit{rewards} \cap \textit{punishments} = \varnothing \\
\hline
\end{array}$$

Figure 2: Z definition of a norm in the framework of Luck et al.

the addressee, normative goal, punishments and context of norms (in the previous approach these were, respectively, the role a, the predicate P or action A, the sanctions and the (temporal) condition C or D). The norm frame of figure 2 expands this with the concepts of beneficiaries, exceptions and rewards, which were left implicit in the approach of the previous section. Additionally, the norm frame of figure 2 also specifies that for norms the inclusion of an addressee, a context and a normative goal are mandatory, and, moreover, it shows that the sets defining the context and the exceptions, as well as the sets of rewards and punishments, are disjoint. Note that punishments and rewards in this norm frame are specified as goals which are to be achieved by norm enforcing agents, that is to say, when the norm is violated the norm enforcing agents of the system are obliged to fulfil the punishment-goal to punish the agent violating the norm.

Using the language introduced in section 3 we can again show that norms specified in this norm frame can be operationalised for use in e-institutions. After contextualisation, the norms can be automatically translated to integrity constraints and inference rules.

The contextualisation of the norms as specified in figure 2 includes linking the addressee, beneficiaries (if present) and normative goal to the correct corresponding utterance, as well as identifying the predicates used in the e-institution to express the context and exceptions. After this contextualisation the norms can easily be translated into the following integrity constraint to detect violations of the norm:

$$(\textit{context} \wedge \sim \textit{exception} \wedge \neg \textit{goal}') \rightarrow \bot$$

where *context* and *exception* are predicates obtained through the contextualisation for specifying the context and exceptions mentioned in the norm, $goal'$ is the contextualised normative goal (thus including the addressee and possible beneficiaries), and the $\sim$ operator is for expressing negation-as-failure (since no exceptions might be given).

If punishments are specified, the following dialogical constraint is also obtained:

$$(\textit{context} \wedge \sim \textit{exception} \wedge \neg \textit{goal}') \Rightarrow \textit{punishment}$$

which defines that *punishment* should be executed by an enforcing agent when the specified condition (i.e. the violation of the norm) occurs. Similarly, rewards (if specified) are handled by the following dialogical constraint:

$$(\textit{context} \wedge \sim \textit{exception} \wedge \textit{goal}') \Rightarrow \textit{reward}$$

specifying that a reward should be given when agents comply to the norm, which is when the norm is active and the normative goal (included in $goal'$) has been achieved.

The obligations of the enforcing agents to punish violations or reward compliance can again be violated, which can be detected by the following integrity constraints:

$$(\textit{context} \wedge \sim \textit{exception} \wedge \neg \textit{goal}' \wedge \sim \textit{punishment}) \rightarrow \bot$$
$$(\textit{context} \wedge \sim \textit{exception} \wedge \textit{goal}' \wedge \sim \textit{reward}) \rightarrow \bot$$

Of course, if punishments and rewards for these violations were specified, these can be translated into new dialogical constraints.

6.2 Event Calculus Norms

Artikis et al. propose in [?] the use of event calculus for the specification of protocols. The event calculus is a formalism to represent reasoning about actions or events and their effects in a logic programming framework. It is based on a many-sorted first-order predicate calculus. The following figure shows the main predicates of the event calculus.

Predicate	**Meaning**
happens(Act,T)	Action Act occurs at time T
initially(F=V)	The value of fluent F is V at time 0
holdsAt(F=V,T)	The value of fluent F is V at time T
initiates(Act,F=V,T)	The occurrence of action Act at time T initiates a period of time for which the value of fluent F is V
terminates(Act,F=V,T)	The occurrence of action Act at time T terminates a period of time for which the value of fluent F is V

Predicates that change along time are called *fluents*. As shown in table below, obligations, permissions, empowerments, capabilities and sanctions are formalised by means of the following fluents: $obl(Ag, Act)$, $per(Ag, Act)$, $pow(Ag, Act)$, $can(Ag, Act)$ and $sanction(Ag)$. Prohibitions are not formalised in the example of [?] as a fluent since they assume that every action not permitted is forbidden by default.

Fluent	**Domain**	**Textual Description**
$requested(S{,}T)$	boolean	subject S requested the floor at time T
$status$	$\{free{,}granted(S{,}T)$	the status of the floor: $status = free$ denotes that the floor is free whereas $status = granted(S, T)$ denotes that the floor is granted to subject S at time T
$best\_candidate$	agent identifiers	the best candidate for the floor
$can(Ag{,}Act)$	boolean	agent Ag is capable of performing Act
$pow(Ag{,}Act)$	boolean	agent Ag is empowered to perform Act
$per(Ag{,}Act)$	boolean	agent Ag is permitted to perform Act
$obl(Ag{,}Act)$	boolean	agent Ag is obliged to perform Act
$sanction(Ag)$	$\mathbb{Z}^*$	the sanctions of agent Ag

The expression below shows an example of an obligation specified in Event Calculus extracted from [?]. The obligation that C revokes the floor holds at time T if C enacts the role of chair and the floor is granted to someone else different from the best candidate.

$$\begin{array}{l}
\mathsf{holdsAt}(\textit{obl}(C, \textit{revoke\_floor}(C)) = \mathsf{true}, T) \leftarrow \\
\quad \textit{role\_of}(C, \textit{chair}) \\
\quad \mathsf{holdsAt}(\textit{status} = \textit{granted}(S, T'), T), (T \geq T'), \\
\quad \mathsf{holdsAt}(\textit{best\_candidate} = S', T), (S \neq S')
\end{array}$$

If we translate all the *holdsAt* predicates into *uttered* predicates, we can translate the obligations and permission of the example by including the rest of conditions in the LHS of the integrity constraints:

$$\begin{array}{l}
(\textit{uttered}(s, w, \textit{inform}(A, R, B, R', \textit{best\_candidate}(S'))\wedge \\
\textit{uttered}(s, w, \textit{inform}(C, \textit{chair}, S, \textit{candidate}, \textit{granted}(S)\wedge \\
S \neq S') \rightarrow \bot
\end{array}$$

$$\begin{array}{l}
(\textit{uttered}(s, w, \textit{inform}(A, R, B, R', \textit{best\_candidate}(S'))\wedge \\
\textit{uttered}(s, w, \textit{inform}(C, \textit{chair}, S, \textit{candidate}, \textit{granted}(S))\wedge \\
S \neq S') \Rightarrow \textit{utter}(s, w, \textit{inform}(C, \textit{chair}, A, R'', \textit{revoke\_floor}))
\end{array}$$

However, since there is no concrete definition of a norm, we cannot state that Artikis' approach is fully translatable into integrity constraints and dialogical constraints.

Although event calculus models time, the deontic fluents specified in the example of [?] are not enough to inform an agent about all types of duties. For instance, to inform an agent that it is obliged to perform an action before a deadline, it is necessary to show the agent the obligation fluent and the part of the theory that models the violation of the deadline.

7. CONTEXTUALISING NORMS

In previous sections we have mentioned that norms need to be contextualised in order to be used in e-institutions. This contextualisation is, in a sense, interpreting the abstract norm from the institution's point of view such that it is usable for implementation. In the example that we used earlier this interpretation was quite clear, as we translated the actions in the following norm:

$$\mathsf{OBLIGED}((buyer\ \mathsf{DO}\ pay(Price,seller))\ \mathsf{IF}\ done(buyer,won(Item,Price)))$$

into the utterances that would be used in an e-institution:

$$uttered(payment,W,inform(A,buyer,B,payee,pay(Item,Price),T))$$
$$uttered(auction,w2,inform(C,auctioneer,A,buyer,won(Item,Price),T'))$$

However, if we regard institutional norms that are derived (or translated) from human laws and regulations, the contextualisation becomes much harder, as laws contain vague and ambiguous concepts and cannot always be related to a single integrity constraint. In order to implement such norms with a high level of abstraction two steps must be taken: 1) interpreting the abstract concepts and link them to concrete concepts used in the institution, and 2) adding procedural information and artifacts to the institution to simplify (or allow) the enforcement of the norm. In this section we examine both these elements.

7.1 Ontological Interpretations of Concepts

The first and mandatory process of the contextualisation of norms with a high level of abstraction is solving this abstract characteristic of the norm, since agents and institutions cannot handle these abstract concepts (protocols and procedures are expressed in concrete concepts and lack any connection to abstract concepts in norms). Consider the following norm of an auction house, expressing that the obligation to identify oneself upon entering an auction:

$$\mathsf{OBLIGED}((participant\ \mathsf{DO}\ identify)\ \mathsf{IF}\ (participant\ \mathsf{DO}\ enter(auction)))$$

The action *identify* in this norm has an abstract meaning and can be implemented in various different manners. To implement this norm the meaning of this abstract action must be defined, which is done by connecting the abstract action to concrete action(s), e.g. through the use of a *counts-as* operator [?, ?]:

$$[participant\ \mathsf{DO}\ give(certificate,manager)\ \mathsf{AND}$$
$$manager\ \mathsf{DO}\ check(certificate)]\ \textsc{counts-as}\ participant\ \mathsf{DO}\ identify$$

describing that giving an identification certificate to the auction manager, and the manager checking this certificate is seen as an implementation of the *identify* action. These counts-as definitions, defining the scope (and applicability) of the abstract concept, are highly context dependent and not necessarily one-one definition, such as we used earlier in section 5.

Implementing these counts-as definitions is achieved by extending the existing ontology of the institution, which already consist of all the concrete concepts used in the institution, with the abstract concepts that are used in the norms and the relation between the abstract and concrete concepts. This relation is defined as a conceptual subset relations, specifying that the ontological meaning of the concrete concepts is included in the ontological meaning of the abstract concepts. In the case of our example, this would mean that the ontological meaning of the actions $give(certificate, manager)$ and $check(certificate)$ are included in the meaning of the abstract action $identify$:

$$give(certificate,manager)\sqcap check(certificate,manager)\sqsubseteq identify$$

7.2 Introducing Procedural Information

After interpreting the abstract concepts of the norm, the norm can be implemented by means of integrity and dialogical constraints as mentioned in sections 3 and 4. In some cases, though, trying to detect a violation would involves making lots of checks, which could be computationally hard or totally infeasible from the institution's point of view. For instance, checking whether every participant of the institution is able to identify oneself at any given time can be very hard, particularly in very crowded institutions.

Moreover, there might be norms in the institution which would have a severe impact on the institution if the norm would be violated. As recovery from such violations (normally done by sanctions and repairs, defined in the dialogical constraints, see section 5) would be (nearly) impossible, it would be wise to try to minimise such violations to an absolute minimum. For example, as murder is, in itself, a very severe violation of the norms of society, any measures that can be taken to limit the occurrence of this violation should be taken (e.g. the prohibition of owning (fire-)arms).

In both cases, one is trying to simplify the enforcement process such that it either becomes feasible to detect the violation, or protect the system from very harmful violations. This process of contextualising norms can be done in two ways. Either the norm is translated to smaller and simpler norms which are easier to check but ensure the compliance of the original norm, or the norm is translated to a set of constraints that ensure the compliance.

Consider the following norm in an auction house, expressing that as an agents bids on an item it has to pay for the item if it won the auction:

$$\mathsf{OBLIGED}((buyer\ \mathsf{DO}\ pay(Price,seller))\ \mathsf{IF}\ done(buyer,won(Item,Price)))$$

Violations of this norm occur, for instance, because the agent does not have enough money to pay, the agent does not want the item anymore or the agent simply disconnects (unintentionally or on purpose). Although the violation of this norm can be detected easily, sanctioning the agent and repairing the situation might be difficult (especially if the agent disconnects). To avoid these situations, one can choose to implement this norm by means of a constraint; upon entering the institution all agents have to deposit an amount of money (for security) that they will get back when leaving the institution if no violations have occurred:

$$\mathsf{OBLIGED}((agent\ \mathsf{DO}\ pay(Security\_Fee))\ \mathsf{IF}\ done(agent,enter(Institution)))$$

However, if a violation of the mentioned norm occurs, this money can be used to pay for the items, thereby sanctioning the agent. This means that our original norm has been implemented by introducing a norm that is easier to enforce (i.e. agents are obliged to pay security before entering), which generates the constraint (or mechanism) that is used for enforcing the original norm. Thus, instead of implementing one norm which was hard to enforce, we have implemented two norms (which were derived from the original norm) that are easily enforced.

8. CONCLUSIONS

With the development of electronic institutions, in their aim of implementing normative open-agent systems, comes the problem of ensuring the safety and stability of the system. Previous implementations of electronic institutions enforced norms by ensuring that the agents joining the system followed a pre-defined protocol, thereby guaranteeing norm compliance of the agents. As this approach severely limits the autonomy of the agents, a more flexible enforcement was desired. This paper proposes the use of integrity constraints and dialogical constraints to implement such a flexible enforcement of norms. This norm enforcement is based on the detection of and reacting to the violations of norms.

In order for any kind of norm enforcement to be implemented, norms need to be expanded with an operational meaning, as the declarative nature of norms only defines what is legal/illegal, but never expresses how this legality/illegality is obtained/averted. In [**?**] we introduced several mechanisms for operationalising norms, where we annotated norms (expressed in deontic logic) with operational aspects, like sanctions and repairs. In this paper we have used this normative frame to design an implementation scheme usable for implementing norm enforcement in electronic institutions. However, before norms can be implemented using this scheme, the norms need to be contextualised. This contextualisation is 1) connecting the abstract concepts of the norm to the concrete concepts used in the institution, and 2) extending the norm with additional procedural information before attempting to implement it. The contextualisation of the norms is, in fact, a further operationalisation of the norms, where, in contrast to declarative norms (which never change the world), the second step of this operationalisation changes the world in order to enforce the norm.

Acknowledgements

This paper was partially supported by the Spanish Science and Technology Ministry as part of the Web-i-2 project (TIC-2003-08763-C02-00).

9. REFERENCES

[1] K. R. Apt. *From Logic Programming to Prolog.* Prentice-Hall, U.K., 1997.

[2] A. Artikis, L. Kamara, J. Pitt, and M. Sergot. A protocol for resource sharing in norm-governed ad hoc networks. In *Proceedings of the Declarative Agent Languages and Technologies (DALT) workshop.* Springer, July 2004.

[3] C. Castelfranchi. Formalizing the informal?: Dynamic social order, bottom-up social control, and spontaneous normative relations. *Journal of Applied Logic*, 1(1-2):47–92, February 2003.

[4] F. Dignum. Abstract norms and electronic institutions. In *Proceedings of the International Workshop on Regulated Agent-Based Social Systems: Theories and Applications (RASTA '02), Bologna*, pages 93–104, 2002.

[5] F. Dignum, J. Broersen, V. Dignum, and J.-J. Ch. Meyer. Meeting the Deadline: Why, When and How. In *3rd Goddard Workshop on Formal Approaches to Agent-Based Systems (FAABS)*, Maryland, April 2004.

[6] M. Esteva. *Electronic Institutions: from specification to development.* Number 19 in IIIA Monograph Series. PhD Thesis, 2003.

[7] M. Esteva, J. Rodríguez-Aguilar, B. Rosell, and J. Arcos. AMELI: An Agent-based Middleware for Electronic Institutions. In *Third International Joint Conference on Autonomous Agents and Multi-agent Systems*, New York, US, July 2004.

[8] M. Esteva, W. Vasconcelos, C. Sierra, and J. Rodríguez-Aguilar. Verifying norm consistency in electronic institutions. In *Proc. of The AAAI-04 Workshop on Agent Organizations: Theory and Practice (ATOP)*, San Jose, California, July 2004.

[9] M. Fitting. *First-Order Logic and Automated Theorem Proving.* Springer-Verlag, New York, U.S.A., 1990.

[10] A. García-Camino and J. Rodríguez-Aguillar. Implementing norms in electronic institutions. In *Proceedings of the 4th Int. Joint Conf. on Autonomous Agents & Multi Agent Systems (AAMAS-05)*, Utrecht, The Netherlands, July 2005.

[11] A. García-Camino, J. Rodríguez-Aguillar, C. Sierra, and W. Vasconcelos. A distributed architecture for norm-aware agent societies. In *Proc. of the 3rd Int. Workshop on Declarative Agent Languages and Technologies (DALT 2005)*, Utrecht, The Netherlands, July 2005.

[12] D. Grossi, H. Aldewereld, J. Vázquez-Salceda, and F. Dignum. Ontological aspects of the implementation of norms in agent-based electronic institutions. *Computational and Mathematical Organization Theory*, to appear in 2006.

[13] D. Grossi, F. Dignum, and J.-J. Ch. Meyer. Contextual taxonomies. In J. Leite and P. Toroni, editors, *Proceedings of CLIMA V Workshop, Lisbon, September*, pages 2–17, 2004.

[14] A. Lomuscio and D. Nute, editors. *Proc. of the 7th Int. Workshop on Deontic Logic in Computer Science (DEON04)*, volume 3065 of *LNCS*. Springer Verlag, 2004.

[15] F. López y López and M. Luck. Towards a Model of the Dynamics of Normative Multi-Agent Systems. In G. L. et.al., editor, *Proc. of RASTA '02*, pages 175–194, Bologna, July 2002.

[16] P. Noriega. *Agent-Mediated Auctions: The Fishmarket Metaphor.* Number 8 in IIIA Monograph Series. PhD Thesis, 1997.

[17] J. A. Rodriguez-Aguilar. *On the Design and Construction of Agent-mediated Electronic Institutions.* Number 14 in IIIA Monograph Series. PhD Thesis, 2001.

[18] J. Vázquez-Salceda, H. Aldewereld, and F. Dignum. Implementing norms in multiagent systems. In G. Lindemann, J. Denzinger, I. Timm, and R. Unland, editors, *Multiagent System Technologies*, LNAI 3187, pages 313–327. Springer-Verlag, 2004.