INSTITUTIONAL REALITY AND NORMS: SPECIFYING AND MONITORING AGENT ORGANIZATIONS

HENRIQUE LOPES CARDOSO

LIACC - NIAD&R, Faculty of Engineering, University of Porto
R. Dr. Roberto Frias, 4200-465 Porto, Portugal

EUGÉNIO OLIVEIRA

LIACC - NIAD&R, Faculty of Engineering, University of Porto
R. Dr. Roberto Frias, 4200-465 Porto, Portugal

Norms and institutions have been proposed to regulate multi-agent interactions. However, agents are intrinsically autonomous, and may thus decide whether to comply with norms. On the other hand, besides institutional norms, agents may adopt new norms by establishing commitments with other agents. In this paper we address these issues by considering an *electronic institution* that monitors the compliance to norms in an evolving normative framework: norms are used both to regulate an existing environment and to define contracts that make agents' commitments explicit. In particular, we consider the creation of *virtual organizations* in which agents commit to certain cooperation efforts regulated by appropriate norms. The supervision of norm fulfillment is based on the notion of institutional reality, which is constructed by assigning powers to agents enacting institutional roles. Constitutive rules make a connection between the illocutions of those agents and institutional facts, certifying the occurrence of associated external transactions. Contract specification is based on conditional prescription of obligations. Contract monitoring relies on rules for detecting the fulfillment and violation of those obligations. The implementation of our normative institutional environment is supported by a rule-based inference engine.

Keywords: Electronic institution; norm; contract; virtual organization.

1. Introduction

An increasingly important dichotomy in multi-agent systems (MAS) research confronts concepts such as autonomy and openness to regulation efforts[1]. While agent theory[31] describes agents as autonomous self-interested entities, preferably interacting in open environments (hence the efforts towards agent communication standards, such as FIPA-ACL[13]), the application of MAS in real-world scenarios raises an important question: how to ensure cooperative behavior in scenarios populated with heterogeneous and self-interested agents? There are (at least) two possible approaches to this problem[4]. One is to impose constraining infrastructures to agents, dictating coordination rules and making it impossible for agents to deviate from the desired behavior. However, this approach severely limits agent autonomy; agents are

1

left to stick to a fixed coordination protocol or do nothing, since any other alternatives are externally constrained by practical impossibility. The other possibility is to regulate the environment, providing incentives for cooperative behavior through normative constraints, and allowing agents to choose whether to obey or to violate them.

A real-world scenario where these concerns clearly apply is in business relationships. In particular, we are concerned with environments in which agents may agree on cooperation efforts, involving specific interactions during a certain time frame. By this way agents compose a virtual organization (VO), which is regulated by the specific norms agreed. Agents may represent different business units or enterprises, which come together to address new market opportunities by combining skills, resources, risks and finances no partner can alone fulfill[9]. The subject of virtual organizations/enterprises is gaining increasing importance in the B2B world, where players are becoming more focused on their core business and rely on outsourcing and dynamic consortiums.

Any cooperation activity requires trust between the involved partners. When considering open environments, previous performance records of potential partners may not be assessable. If such information is available, reputation mechanisms can be put in place. If not, we must take into account that it is possible for a VO to include agents that have never worked together in the past. In this case, trust must be based on third-parties, which verify that agents comply with their promises. It is therefore necessary first to explicitly state through a formal and legally binding contract what an agreement is about, and second to provide an environment for enforcing those contracts. If no such environment can be put into place, then agents are on their own to decide whether they are willing to take the risk of being cheated: mutual trust can only be formed by experience over time.

Electronic Institutions (EI) have been proposed to mimic real-world institutions, by regulating the interactions between agents. In our view, the EI concept addresses the above concerns[19], as it consists of a coordination framework facilitating the establishment of contracts and providing a level of trust by offering an enforceable normative environment. This is accomplished by providing a number of agent-based institutional services. The EI encompasses a set of norms regulating the environment. This normative environment evolves as a consequence of the establishment of agents' agreements formalized in contractual norms. Therefore, an important role of the EI is to monitor and enforce, through appropriate services, both predefined institutional norms and those formalizing contracts that result from a negotiation process. Agents rely on the EI to monitor their contractual commitments.

The EI is thus a regulated environment where agents interact. When considering software agents, interactions are based on the exchange of messages (illocutions). Inside the EI, agents' illocutions are the starting point towards the formation of institutional reality (inspired by Ref. 27). This reality is composed of both new organizational structures (VOs) and actions performed concerning the compliance to norms. Within institutional boundaries, some agents perform specific institutional

roles, being certified by the EI as legitimate to achieve certain institutional facts. External agents may also announce themselves as performing certain external roles. Instead of providing institutional services, these are general roles (such as supplier or customer) which may, when performed inside the EI, have a set of attached norms. By assuming those roles, agents become committed to these norms.

This paper is structured as follows. Section 2 describes the use of norms within our EI framework, assigning contexts to them. Afterwards, section 3 develops on the notion of institutional reality and its creation, by detailing the adaptation of Ref. 27 to our approach; we imported concepts such as brute and institutional facts, and constitutive rules, and this section relates them to institutional roles. Section 4 proceeds to the specification of contracts through norms and to their monitoring and enforcement, taking into account the normative structured presented in section 2. We emphasize on the prescription of obligations and on rules for detecting their fulfillment or violation. We also show how to deal with VO cooperation agreements. Section 5 addresses the issue of implementing the normative core of our EI. Section 6 concludes and presents related work.

2. Norms

The concept of social structure is a central issue in the social sciences: it is assumed to exist in order to impose a sense of order and predictability within a group of individuals. Social structure typically involves a framework of norms attached to roles played by members of a society. A role represents the way someone is expected to behave in particular situations.

From a social perspective, different types of norms can be identified, with different abstraction levels. At one hand we have values as general principles, conventions, and abstract norms that are implicitly adhered to and may not have a formally defined social response in case of deviation. On the other side of the spectrum we may have formal (legal) norms and prescriptions that include explicitly defined punishments to deal with infractions. Within MAS research, norm classifications have also been made (e.g. Ref. 29 and 21).

Norms play an important role in open artificial agent systems; they have been said to improve coordination and cooperation[7]. As in real-world societies, norms provide us a way to achieve social order[4] by controlling the environment and making it more stable and predictable.

2.1. *Norms vs. rules*

The concept of *norm* has been used interchangeably with the term *rule*. A subtle distinction might be that a norm is a rule with a wide applicability, i.e. to a group of agents. Advanced properties of multi-agent systems, such as the possibility for norm emergence[30], might lead us to say that when a group of agents adheres to a rule, this rule becomes a norm.

Another important source of confusion regards the representation of norms in a declarative and computational fashion. Norms can be implemented with rule-based systems – this fact contributes to an undifferentiated use of the terms *norm* and *rule*. However, we believe that not all rules are norms. Norms have an intrinsic property: they *prescribe behavior*.

2.2. *Contextualized norms*

Norms can be classified according to different criteria. Considering our EI framework, we find it important to classify norms according to their scope. Accordingly, we distinguish between institutional, constitutional, and operational norms[20].

Institutional norms regulate the behavior of every agent inside the EI. As behavior prescription is typically dependent on agents' commitments, institutional norms represent the commitments of agents towards the EI as a whole: by assuming general roles, agents become committed to their associated norms. Institutional norms also include general ways of dealing with contract-independent occurrences, such as policies for handling norm violations. Institutional norms set up the normative ground on which cooperation commitments can be established.

Constitutional norms are used to describe the constitution of agents' virtual organizations, which thereby commit to a certain cooperation agreement. The terms of such an agreement are specified by means of norms regulating the created consortium, which usually exists for a period of time.

Finally, *operational norms* specify contracts by indicating actions to be performed by contractual agents. Operational contracts may be proposed and signed within the context of a specific VO, or else may comprise a stand-alone deal.

The described classification suggests that different types of norms are created at different moments. Thus, institutional norms may be pre-existent, while constitutional norms are created when agents reach a cooperation agreement, and operational norms come into existence when executable contracts are signed. However, this needs not be the case. Norms with limited scopes may be predefined for a number of reasons, as follows.

An important concept in contract law theory is the use of "default rules"[8]. These exist with the intent of facilitating the formation of contracts, allowing them to be underspecified by defining default clauses or default values. Therefore, constitutional or even operational norms may be institutionally defined: together with institutional norms, they provide a normative background in which agents can rely to build their contractual commitments.

Furthermore, just as real-world legislations are organized through hierarchies of laws, it is natural to have predefined regulations devoted to particular contexts, such as the VO setting. Norms regulating specific aspects of this type of contracts are, naturally, predefined. Agents can rely on these regulations as a ground basis to raise specific virtual organizations.

Another situation where norms with a limited scope are pre-existing is when they

regulate predicted coordination scenarios, as in the case of negotiation protocols. In such situations, agents agreeing to coordinate their negotiation efforts according to a certain protocol simply adhere to the norms implementing it. Notice that, differently from "default rules", these norms apply as they are to every adhesion to the protocol; they do not make up mere fill-in prescriptions in case of absence of explicitly created norms.

According to this setting, it is possible, therefore, to predefine scoped norms that are to be imposed when the activity they regulate is adhered to by agents. This methodology may be applied to negotiation protocols, to standard cooperation commitments among a group of agents, or even to norms attached to roles. Although having a limited scope, these norms can be seen as institutional in the sense that they are institutionally predefined.

Independently of the circumstances of their creation, we consider that norms define, within a given context, what ought to be done in certain circumstances. As such, norms can be represented as:

$$[Context]\ Situation \rightarrow Prescription$$

The *Context* indicates the scope of the norm; as such, we may have norms applicable inside the whole institution (in which case the context may be omitted), while other norms may be defined, e.g., inside a particular agents' VO. The *Situation* describes when the norm is in place. The *Prescription* specifies what should be accomplished in order for the norm to be fulfilled. Contexts therefore allow us to structure existing norms and, as we shall see in the next section, permit the contextualization of occurring facts, facilitating the assessment of their consequences. Contexts will also support a hierarchical composition of the normative environment.

Using the above representation, norms obviously lend themselves to a rule-based implementation. The situation part refers to the rule conditions, whereas the prescription part corresponds to the rule's effects.

3. Institutional Reality

Considering an EI as a place where social relationships are created and enforced, it is necessary to establish how and when such relationships are in place, and how and when their underlying commitments are fulfilled. If we design a closed environment with a well-defined performative structure (as in Ref. 11), then agents' actions and their effects are restricted. If, however, we design an open environment where autonomous agents interact, we must relate those interactions with the (emergence of) social structures defining interdependencies between agents, that is, specifying commitments among them. This represents a much more flexible approach towards the development of a normative framework.

Following Searle's theory on "the construction of social reality"[27], inside the EI we distinguish between *brute facts* and *institutional facts*. The latter are obtained from the former, through rules defining "counts-as" relations (*constitutive rules,*

according to Searle). In our case, brute facts do not refer to physical reality, but rather to agents' actions. In a society of artificial agents, the only observable actions are the utterance of illocutions: these are our basic building blocks, that is, the brute facts we build on to infer institutional reality. Rules regulating the creation of institutional facts will verify not only what is uttered, but also who perpetrates such actions – the effects of illocutions may depend on who utters them.

3.1. *Brute facts*

Agents' illocutions are stored as brute facts in an EI component responsible to interpret them: the "institutional reality engine". Notice, however, that only messages sent to the EI are recordable. Agents are free to exchange messages with other agents, as no restriction on behavior is imposed.

Considering that agents use the FIPA-ACL message structure specification[13], we store their EI-directed illocutions in the form:

   illocution(Sender, Content)

where *Sender* indicates the agent affirming the illocution and *Content* represents the content of a FIPA-ACL message. Illocutions stored as brute facts include only informative messages (e.g., *inform*[14]); they are assumed to use a well-defined institutional ontology.

3.2. *Institutional facts*

Institutional facts are inferred using constitutive rules. These ascribe meaning to certain happenings within a certain social context. In our case, such happenings correspond to agent illocutions (brute facts), which inside the institutional context may produce institutional reality through constitutive rules (more on this in subsection 3.3.3).

Just as norms have a context in which they apply, we associate institutional facts with the *context* within which they occur. Searle argues that there are many situations in which the context of an institutional fact needs not exist anymore after that fact has been created (using the "x counts as y in context c" formula). In our case, the main context consists of the EI itself, inside which certain illocutions count as certain institutional facts. However, if we consider facts denoting agent behavior related to deontic notions such as obligations or roles, these facts occur within their context (e.g. a contract or an organization). Therefore I may, for instance, pay a given amount of money referring to an invoice issued within a contract. Besides, institutional reality may be governed by institutionally defined rules, while the facts themselves may refer to social structures that are of a smaller scope, i.e., representing a narrower reality (e.g. a contractual relationship between some of the agents, or an organization including a subset of all the existing agents). Therefore, the context in which the "counts-as" relation applies may be broader than the sub-context in

which the fact occurs. This emphasizes the difference between fact recognition (in an institutional context) and fact contextualization (e.g. contractual).

An important issue to consider in contracting scenarios is time. Every fact must occur at a certain instant, provided by an institutional clock. We use the following representation for institutional facts:

> *[Context] ifact(IFact, Timestamp)*

where each part speaks for itself.

3.3. *The institutional reality engine*

Norm creation and fulfillment supervised by the EI depends on the recording of brute facts, which are then processed by "systems of constitutive rules"[27]. As explained above, brute facts correspond to agents' illocutions. In our perspective, institutionally relevant facts include those events that are important in respect to commitment creation (implying the establishment of contracts) and fulfillment. Therefore, rules regulating how these facts come about are needed.

3.3.1. *Institutional roles*

Illocutions are stored as brute facts in our institutional reality engine. The effects of such illocutions may depend on the actors uttering them. To deal with this we identify a set of *institutional roles* enacted by agents providing institutional services. Thus, some institutional facts may come into existence only if agents performing certain institutional roles execute some predetermined actions (that is, utter appropriate illocutions). Authoritative relations are established in this way between roles and institutional reality: an agent performing a given role is said to be empowered to achieve the effects expressed in its role-related constitutive rules. This, however, does not imply that illocutions of agents which do not perform institutional roles have no relevance. We may still define rules empowering them. Typically, however, those agents will need to interact with agents performing institutional roles, in order to obtain certain institutional facts.

As we are concerned with the application of our model to business scenarios involving transactions, we identify three main institutional roles providing a connection to the real-world. A *messenger* role provides certified information exchange facilities. A *banking* role enables acknowledging monetary value transfers. A *delivery tracker* role certifies product delivery. Different domains may require different types of roles; whenever a third-party is needed to certify the occurrence of a fact, an institutional role with its associated constitutive rules should be in place.

3.3.2. *An ontology for brute and institutional facts*

Although agents are free to utter any illocutions, only a subset of these has a meaning inside the EI. The institutional ontology may include the messages (brute

facts) presented in table 1.

Table 1. An ontology for brute facts

Content	Meaning
currency_transfered(Ctx, Ag1, Am, Ag2, T)	a currency transfer of amount Am was made at time T between agents $Ag1$ and $Ag2$, within context Ctx
paid(Ctx, Am, Ag2, T)	I have paid amount Am at time T to agent $Ag2$, within context Ctx
collected(Ctx, Am, Ag1, T)	I have collected amount Am at time T from agent $Ag1$, within context Ctx
delivered(Ctx, Ag1, Item, Qt, Ag2, T)	a delivery of Qt units of $Item$ was made at time T from agent $Ag1$ to $Ag2$, within context Ctx
sent(Ctx, Item, Qt, Ag2, T)	I have sent Qt units of $Item$ at time T to agent $Ag2$, within context Ctx
received(Ctx, Item, Qt, Ag1, T)	I have received Qt units of $Item$ at time T from agent $Ag1$, within context Ctx
msg_delivered(Ctx, Ag1, Msg, Ag2, T)	message Msg was delivered at time T from agent $Ag1$ to $Ag2$, within context Ctx
msg_sent(Ctx, Msg, Ag2, T)	I have sent message Msg at time T to agent $Ag2$, within context Ctx
msg_received(Ctx, Msg, Ag1, T)	I have received message Msg at time T from agent $Ag1$, within context Ctx

When uttered by agents enacting appropriate roles, these illocutions produce the institutional facts presented in table 2.

If other kinds of facts are needed, this institutional ontology may be extended to consider them.

Institutional reality is not exclusively created from agents' illocutions. An important exception is the generation of institutional facts corresponding to time events, which are important for detecting violations of obligations (as described in section

Table 2. An ontology for institutional facts

IFact	Meaning
payment(Ag1, Am, Ag2)	a payment of amount *Am* took place from agent *Ag1* to *Ag2*
delivery(Ag1, Item, Qt, Ag2)	a delivery of *Qt* units of *Item* took place from agent *Ag1* to *Ag2*
msg(Ag1, Msg, Ag2)	a message *Msg* was delivered from agent *Ag1* to *Ag2*

4.2).

3.3.3. *Constitutive rules*

Constitutive rules, the central component of the institutional reality engine, make a connection between what is said and what is taken for granted and possible consequences of this. Many of these rules will define institutional roles and their powers. According to our EI rationale, we identify two main focuses for constitutive rules: (1) the recognition of action execution, and (2) the establishment of commitments through contracts. The former includes the fulfillment of contracts through the realization of appropriate transactions.

The verification of action execution is important because of trustworthiness issues. The way we address this problem is through the use of the aforementioned institutional roles.

Certifying action execution

Consider a situation in which an agent ought to make a certain payment to another. Although the agent may claim to have paid its debt, that does not make it the case. Still, if an independent financial third party agent (the receiver's bank), providing a certified institutional service, states that a currency transfer referring to a certain context (e.g. a purchase contract) has taken place, it would be safe to consider that the payment took place:

> *illocution(B, currency_transfered(Ctx, Ag1, Amount, Ag2, Time))* ∧
> *ibank(B)*
> → *[Ctx] ifact(payment(Ag1, Amount, Ag2), Time)*

We can also say that if both agents (the payer and the receiver) state that a payment took place, it would also be safe to conclude the associated institutional fact:

> *illocution(Ag1, paid(Ctx, Amount, Ag2, _))* ∧
> *illocution(Ag2, collected(Ctx, Amount, Ag1, Time))*

$\rightarrow$ *[Ctx] ifact(payment(Ag1, Amount, Ag2), Time)*

Another exemplifying case where physical actions must be checked concerns the delivery of products. We may trust on a delivery tracking service:

illocution(DT, delivered(Ctx, Ag1, Item, Quantity, Ag2, Time)) $\wedge$
idelivery_tracker(DT)
$\rightarrow$ *[Ctx] ifact(delivery(Ag1, Item, Quantity, Ag2), Time)*

And again rely on the confirmation of both involved parties:

illocution(Ag1, sent(Ctx, Item, Quantity, Ag2, _)) $\wedge$
illocution(Ag2, received(Ctx, Item, Quantity, Ag1, Time))
$\rightarrow$ *[Ctx] ifact(delivery(Ag1, Item, Quantity, Ag2), Time)*

The same methodology can be applied concerning the exchange of messages. If message delivery recognition is a must, an institutional messenger service may be provided. This way, interactions between agents through the exchange of messages can be recorded, as long as such a service intermediates the process. Agents may use this service by using the FIPA-ACL *proxy* communicative act[14].

The messenger service informs the EI that a given message was delivered. The following constitutive rule applies:

illocution(M, msg_delivered(Ctx, Ag1, Msg, Ag2, Time)) $\wedge$
imessenger(M)
$\rightarrow$ *[Ctx] ifact(msg(Ag1, Msg, Ag2), Time)*

The context (*Ctx*) of the message delivered may be obtained from the FIPA-ACL message *conversation-id* parameter[13].

Agents can also confirm such a message delivery:

illocution(Ag1, msg_sent(Ctx, Msg, Ag2, _)) $\wedge$
illocution(Ag2, msg_received(Ctx, Msg, Ag1, Time))
$\rightarrow$ *[Ctx] ifact(msg(Ag1, Msg, Ag2), Time)*

In principle, any information exchange could be treated in a similar way. This opens up the possibility to verify business-related activities such as order placement, invoice issuing, acceptance confirmations, or shipment notices. It also enables the follow-up of negotiation processes that require the exchange of proposals. As an example, we may acknowledge the placement of an order using the following rule:

[Context] ifact(msg(A1, Msg, A2), Time) $\wedge$
Msg = order(Item, Quantity)
$\rightarrow$ *ifact(order(A1, Item, Quantity, A2), Time)*

There is, though, a difference between this particular rule and the former ones: it is based on institutional facts. These may, therefore, be used to infer new institutional reality. The indicated context applies both to the antecedent and consequent *ifacts*, which report to the same context.

Establishing commitments

As for contract establishment, we must define relations between institutional facts (created from agents' illocutions) and commitment formation. The distinguishing feature of these rules is that they define how new institutionally enforceable norms can be created, where norms describe contractual relationships between agents.

Contractual relationships may rise from an appropriate exchange of messages. These may be logged, as above, using an institutional messenger service. Constitutive rules are used to detect situations in which contracts come into existence, as a result of a successful interaction. For instance, a contract may be recognized if an agent accepts the terms and conditions proposed by the other (that is, when they reach an agreement):

> *ifact(acceptation(Ag1, Ag2), TA)* $\wedge$
> *ifact(proposal(Ag2, Ag1, Proposal, Timeout), TP)* $\wedge$
> *TP< TA< TP+Timeout*
> $\rightarrow$ *register_new_contract(Ag1, Ag2, Proposal, TA)*

where *register_new_contract* would be an institutional procedure registering the contract between the involved agents. The proposal should embrace contractual details, such as contract clauses. This can be simplified by means of contract templates, over which agents negotiate by instantiating negotiable parameters.

While the above rule comprises a simple example, several researchers address the issue of commitment creation from interaction protocols. A survey may be found in Ref. 22.

If, however, negotiation protocols must be enforced, or if negotiation must be mediated (besides mere message forwarding), negotiation mediation services may be provided by an institutional agent (i.e., an agent providing an institutional service). This applies to negotiation protocols specifically devoted to formalizing VO cooperation agreements[23].

Agents may also use the EI for contract enforcement purposes only. Therefore, it is possible to conceive a notary service where agents can register contracts, which become in force upon all participants confirmation (either through digital authentication or simple illocutionary adhesion).

3.4. *Institutional reality and norms*

Institutional reality forms the basis for implementing a normative framework regulating the institutional environment. Figure 1 shows the connection between constitutive rules and norms. As explained above, besides creating institutional facts, constitutive rules also make it possible to generate new norms, which regulate a specific relationship amongst a group of agents by specifying commitments.

There is an important part of institutional reality that is not dependent on agent's illocutions: time. As we explain in the next section, norm prescriptions are frequently dependent on deadlines, which make it important to verify the passage
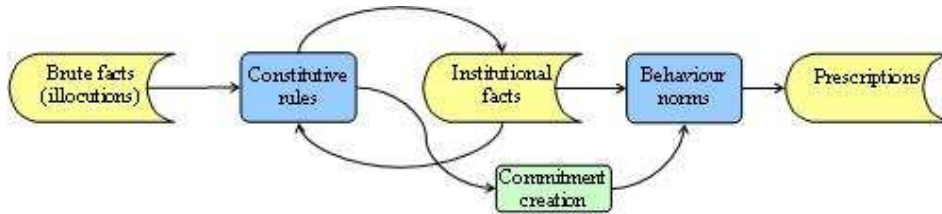
Fig. 1. Connection between constitutive rules and norms

of time using a synchronized (institutional) clock.

4. Specifying and Monitoring Contracts through Norms

Behavior norms prescribe the expected behavior of agents. Usually, such norms specify states of affairs that *must* be brought about by an agent before a certain deadline. Therefore, we consider *obligations* as the means to express the prescription of behavior norms. Obligations have the following structure:

[Context] obligation(Bearer, InstitutionalFact, Deadline)

Notice that instead of dictating the exact action an agent must perform, we prescribe the institutional fact that he must bring about. This fits with our model of institutional reality, where we specify by means of constitutive rules how an institutional fact may be accrued. It also enables an agent to delegate or outsource tasks conducting to the accomplishment of such state of affairs, while still being responsible before the institution for the (un)fulfillment of the obligation.

Contracts are specified by describing what obligations hold in which situations. They comprise the exchange of resources (e.g. products and money) between participants. While simple purchase contracts are composed of a predetermined normative path (that is, a sequence of fixed obligations), VO cooperation agreements are more complex, in that they have an ongoing nature. They describe cooperation efforts of participating agents, whose performance is dependent on appropriate requests. Therefore, VO agreements include repetitive yet unscheduled interactions.

4.1. *Simple contracts*

General contracts for purchase operations between a customer and a vendor can be specified by an institutional fact stating that such a contract is in place, together with a set of behavior norms representing contract clauses that prescribe the expected behavior of contracting partners. Concerning behavior norms, there are two complementary approaches: the institution may provide a predefined set of norms to which agents limit themselves to agree, or the agents may define their own set of norms.

4.1.1. *Using a predefined set of norms*

An institutional fact introducing the purchase contract describes the involved parties and the resources to be exchanged. It is an instantiation of:

> *[] ifact(purchase_contract(IdPC, Customer, Vendor, Item, Qt, Price),*
> *PCTime)*

The contract's behavior norms can be predefined and applicable to any purchase contract adhering to them (the contract id is left unspecified in the norms):

> *[purchase_contract:IdPC]*
> *obligation(Vendor, delivery(_, Item, Quantity, Customer), PCTime+10)*
>
> *[purchase_contract:IdPC] fulfilled(Vendor, delivery(_, Item, Quantity,*
> *Customer), TD)*
> → *obligation(Customer, payment(_, Price, Vendor), TD+30)*

The syntax is simplified for greater readability: in this representation, we assume variable names have contract scope (as opposed to Prolog's clause scope). The absence of a designated agent in institutional facts indicates that such tasks may be delegated (that is, the agent achieving the indicated state of affairs may be other than the obligation's bearer). Within a norm, literals either in the situation or in the prescription part are contextualized. Although these norms have a context embracing all purchase contracts, they concretize their context (by means of instantiation) upon their "run-time" appliance to a specific purchase contract.

4.1.2. *Defining customized contracts*

The set of predefined norms as defined above makes it easier to establish a new purchase contract. However, these norms are fixed. If different purchase contracts are needed, they may be negotiated using contract templates. Different templates may be defined, with different flexibilities. At the extreme, agents can define the behavior norms that will specify their contractual relationship. However, starting a negotiation where nothing is fixed represents a too ill-structured problem to consider automating. The importance of a contract template resides on its ability to provide a structure on which negotiation can be based.

Therefore, contract templates with different degrees of flexibility may be provided, allowing agents to negotiate their details by instantiating parameters. Still, the resulting contract is represented as an institutional fact:

> *[] ifact(custom_purchase_contract(IdCPC, C, V, I, Qt, P), CPCTime)*

Depending on the template used, different sets of norms may be obtained. The difference to the previous approach is that these norms will be applicable only to the contract at hand, and will thus be instantiated with customized contract data.

14   *Henrique Lopes Cardoso, Eugénio Oliveira*

4.1.3. *Sanctioning norms*

Behavior norms may be complemented with sanctioning norms that prescribe new obligations in case of violation. The general structure of a sanctioning norm is as follows:

> *[Context] violated(Bearer, IFact, Deadline)*
> *→ obligation(Bearer, NewIFact, NewDeadline)*

This norm states that if the agent identified as the bearer for the achievement of an institutional fact fails to do so by a certain deadline, then a new obligation is assigned to him. Thus, a violation imposes a new obligation on the prevaricator. Considering the two described approaches for norms in purchase contracts, the *Context* may be either general purchase contracts or a specific customized purchase contract.

Sanctioning norms are useful to discourage norm violation. If the new prescribed obligation is violated again, other enforcement mechanisms must be put into place, such as reputation updates (see below), or other coercive measures.

4.2. *Fulfillment and violation of obligations*

Institutional facts (which are contextualized) are used to verify the fulfillment of obligations. For this, we define an *obligation fulfillment rule* applicable to all contexts (that is, to any contractual relationship monitored by the EI):

> *[Context] ifact(IFact, T) ∧ obligation(Bearer, IFact, Deadline) ∧*
> *T<Deadline*
> *→ fulfilled(Bearer, IFact, T)*

This rule indicates that if an institutional fact prescribed by an obligation is achieved before its deadline, then that obligation is fulfilled. As with behavior norms, literals within the rule are dependent on its context. That is, if an obligation within a certain contract is accomplished, the fulfillment of such obligation occurs, obviously, inside that same contract. However, this rule is institutional, in that it applies to all contractual relations; it thus has un-instantiated context.

This rule is fundamental for enabling the chaining of obligations within a contractual relationship. It establishes a connection between the institutional facts that are added and the pending obligations.

The connection between behavior norms and sanctioning norms is achieved through violation detection mechanisms. These are based on *violation detection rules*, which fire when deadlines have elapsed. For this we consider time events, which are generated as institutional facts corresponding to the time when obligations are due.

> *[Context] ifact(time, Deadline) ∧*
> *obligation(Bearer, IFact, Deadline) ∧ not(fulfilled(Bearer, IFact, _))*
> *→ violated(Bearer, IFact, Deadline)*

This violation detection rule states that in any context, if a deadline referring to an obligation was reached, and such obligation was not fulfilled, then a violation occurred (in the same context).

The resulting fact stating that a violation has occurred may be used to apply punishment measures, either direct (sanctions, restrictions, interdictions) or indirect (reputation). We may connect a violation with an update on the agent's reputation:

>*[Context] violated(Bearer, IFact, Deadline)*
>*→ update_reputation(Bearer, ReputationPunishmentValue)*

Violations can constrain the effects of further actions from the prevaricator's part, thus making it possible to define prohibitions based on violations. Since we cannot restrict the actions taken by truly autonomous agents, one way of defining such constraints is by affecting the constitutive rules, which relate agent's illocutions to institutional reality. By this way, we forbid an agent of achieving the effects of uttering appropriate illocutions (e.g., signing a new contract). Another possibility is for violations to interdict agent access to institutional services, such as message delivery, negotiation mediation, or contract registration through a notary service.

Violations may also be used in a less severe way, simply by prescribing new obligations to be attended, as specified in the contract. This approach allows us to distinguish violation detection from sanction imposition mechanisms. While the detection of violations is a general and institutionally defined concept, the prescription of sanctions may be contract-specific.

4.3. *Virtual organization cooperation agreements*

A cooperation agreement aggregates the organization's constitutional information, including the cooperation effort agents commit to, and their general business process flow. As with simple contracts, we can choose to have predefined norms applicable to typical cooperation agreements, or let agents customize their norms (which may be done using templates, so as to keep enabling the automation of the VO set-up phase).

Considering the first case, an institutional fact registers that a cooperation agreement is in effect within a group of agents, each ensuring a specific cooperation effort. The following templates are used to specify this information:

>*[] ifact(cooperation_agreement(IdCA, Participants, Resources),*
>*   CATime)*

>*[cooperation_agreement:IdCA]*
>*coop_effort(Participant, Resource, MinQt, MaxQt, Frequency,*
>*   UnitPrice)*

>*[cooperation_agreement:IdCA] business_process(From, Resource, To)*

Cooperation efforts indicate quantity ranges for the supply of resources, within a given frequency, together with agreed prices. They represent the commitments

of participants. Business process entries indicate the resources that are supposed to flow between participants. Their effective transfer, however, is dependent on appropriate requests. That is, business process statements do not indicate specific dates or quantities; these will be specified at "run-time", as needed according to the business performance.

A minimum of one *coop_effort* fact per participant and per resource should be present. Also, there should be at least one *business_process* fact for each *coop_effort*, and their agent-resource linkage must conform to the latter. The cooperation agreement contract should also verify that all participants and resources specified within cooperation effort and business process sections are included in the participants and resources declared in the institutional fact. These validations can be performed by a contract validation facility that checks the contract before appending it to the normative environment.

The central norm in respect to contractual promises represents the fact that each agent is committed to its cooperation effort. As with other behavior norms, this translates to an obligation prescription:

> *[cooperation_agreement:IdCA]*
> *ifact(request(Requester, Resource, Qt, Answerer), TR)* $\land$
> *business_process(Answerer, Resource, Requester)* $\land$
> *coop_effort(Answerer, Resource, MinQt, MaxQt, Freq, _)* $\land$
> *calculate_performed_effort(Answerer, Resource, Freq, TR, PE)* $\land$
> *PE+Qt* $\leq$ *MaxQt*
> $\rightarrow$ *obligation(Answerer, acknowledge(Answerer, Resource, Qt,*
>   *Requester), TR+10)*

This norm is institutional: it applies to all cooperation agreements created inside the institution. Thus, as before, its context remains unbound until it is in use. Briefly, it states that if a contractually predicted request (considering the stated business process and cooperation effort) is made in the context of a cooperation agreement, then the envisaged agent is obliged to accept it. An institutional procedure (*calculate_performed_effort*) is invoked for calculating the effort already performed by the agent within the time frame indicated in the cooperation effort frequency, taking into account the request time. If the agent does not exceed its promised efforts (that is, if the previously supplied amount of resource is sufficiently below the agent's stated commitment), the obligation comes into effect.

4.3.1. *Operational contracts*

Operational contracts are established in the context of a cooperation agreement. Institutional facts register their creation:

> *[cooperation_agreement:IdCA]*
> *ifact(operational_contract(IdOC, Requester, Answerer, Resource, Qt),*
>   *OCTime)*

Considering parties' cooperation commitments, operational contracts come into existence through a constitutive rule of the form:

> *[cooperation_agreement:IdCA]*
> *fulfilled(Answerer, acknowledge(Answerer, Resource, Qt, Requester),*
>   *TA)*
> → *register_new_operational_contract(IdCA, Requester, Answerer,*
>   *Resource, Qt, TA)*

This rule applies to every cooperation agreement, and states that when an agent fulfils its obligation to acknowledge a given request, a new operational contract comes into existence. The acknowledgement is meant to inform the requester that the partner (as it ought) will indeed stick to its promise, and the operational contract can be put into place.

The cooperation agreement may also specify how operational contracts are managed, that is, what obligation chains implement such activity. Norms prescribing behavior on operational contracts may thus be pre-specified when the cooperation agreement is created.

One possibility is to consider operational contracts as purchase contracts using a predefined set of norms:

> *[cooperation_agreement:IdCA]*
> *ifact(operational_contract(IdOC, Requester, Answerer, Resource, Qt),*
>   *OCTime)* ∧
> *coop_effort(Answerer, Resource, _, _, _, UnitPrice)*
> → *ifact(purchase_contract(IdOC, Requester, Answerer, Resource, Qt,*
>   *UnitPrice*Qt), OCTime)*

Another option is to specify norms applicable to all operational contracts within a cooperation agreement. These would be defined, once again, using appropriate templates. For instance, if delivery and payment should occur within an operational contract:

> *[cooperation_agreement:IdCA, operational_contract:IdOC]*
> *obligation(Answerer, delivery(_,Resource, Quantity, Requester),*
>   *OCTime+10)*

> *[cooperation_agreement:IdCA, operational_contract:IdOC]*
> *fulfilled(Answerer, delivery(_, Resource, Quantity, Requester), TD)* ∧
> *coop_effort(Answerer, Resource, _, _, _, UnitPrice)*
> → *obligation(Requester, payment(_, UnitPrice*Quantity, Answerer),*
>   *TD+30)*

in which *IdOC* remains unbound, as these norms apply to all operational contracts which will be created in the future within agreement *IdCA*.

If, however, nothing is pre-specified, then agents must negotiate the terms of operational contracts, again assisted by templates. If they do not, then an institutional default policy may be defined, applicable to all operational contracts of all cooperation agreements.

5. Implementing a Normative Environment

Most work on normative systems relies on deontic formalizations, and focuses on defining formal semantics. However, recent attention is being given towards norm implementation in MAS[28]. An explicit declarative representation of norms is needed. If norms are enclosed inside agents' code, we get a static environment where norms cannot change over time. This prevents agents from adopting norms by establishing commitments through contracts.

According to Ref. 28, an operational semantics for norms comes down to either (i) defining constraints on unwanted behavior, or to (ii) detecting violations and reacting to these violations. In MAS, the absence of control over autonomous agent behavior leads us to the latter practice. Taking this into account, in our EI environment norms are based on obligation prescription. Agents' actions may be constrained as an outcome of violation conducts.

As obligations have associated deadlines, an institutional clock is essential for violation detection. It is responsible for generating institutional facts corresponding to time instants when deadlines are due.

Fulfillment detection is based on the assumption that it is in the best interest of agents to publicize the fulfillment of obligations. They do so by provoking the achievement of corresponding institutional facts (which are accrued by authorized agents). If agents fail to convince the EI that they complied, then they expose themselves to punishments, either direct (e.g. sanctions) or social (e.g. reputation records). This means that the active part of norm enforcement concerns the detection of violations, while their fulfillment is verified passively, upon the occurrence of external events (that is, events triggered by external agents).

Norms lend themselves to a rule-based implementation. Since the normative environment is based on the occurrence of events, the obvious solution towards its implementation is by using a forward-chaining production system (e.g. JESS[17]). The knowledge base consists of norms, rules and contracts; the working memory includes facts, obligations and sanctions (see figure 2).

The notions of templates and default norms demand for a default reasoning approach. An expert system shell that supports this kind of reasoning is thus appropriate (JESS[17] incorporates features enabling both rule and frame-based approaches, including also the possibility to define modules, which may be used to organize sets of rules – norm contexts in our case).

We now develop on our implementation using the JESS shell [JESS]. We describe our use of Jess templates and rule definitions. Jess uses a Lisp-like notation, in that a relation is defined using a list whose head is the name of the relation. It is also possible to define templates for facts, where they can be structured with slots. We explore this facility to structure institutional reality, and we rely on the simpler list representation for handling ontology-related slots.
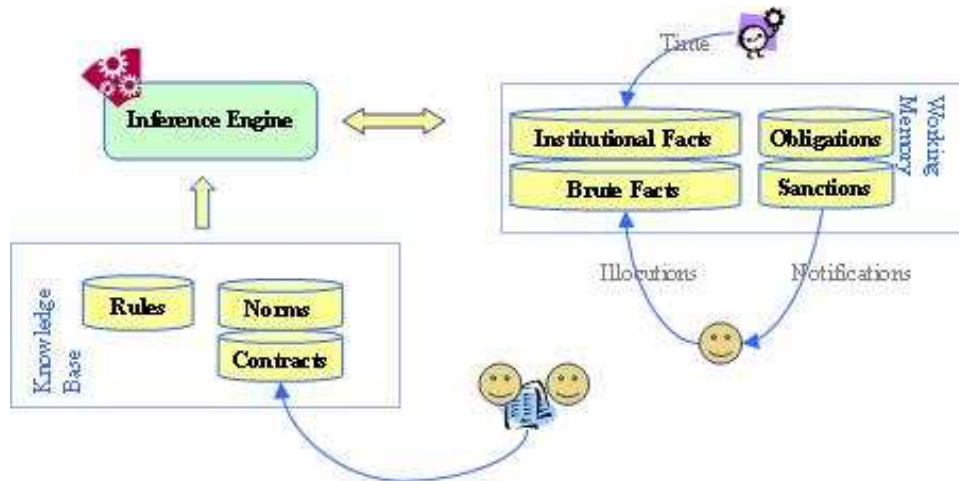
Fig. 2. Normative environment as a rule-based production system

5.1. *Representing institutional reality*

We start by defining templates for brute and institutional facts, and for roles. When a message arrives, it is asserted as an instance of the following template:

```
(deftemplate illocution
   "Brute facts"
   (slot sender)
   (multislot content)
)
```

The ontology for brute facts is subsumed inside multislot `content`.

Institutional facts resulting from the application of constitutive rules are based on a template taking into account the context within which that fact occurred:

```
(deftemplate ifact
   "Institutional facts"
   (slot context (default MAIN))
   (slot when)
   (multislot fact)
   (slot processed (default FALSE))
)
```

The ontology for institutional facts is subsumed inside multislot `fact`. The indication of the `processed` state of an `ifact` is useful for the application of norms.

Roles performed by agents are defined using the following template:

```
(deftemplate agent-role
   "Roles for agents"
   (slot agent)
   (slot role)
)
```

We then add templates that allow us to create norm-related elements, such as obligations, their fulfillment or violation. These are based on the institutional fact template, adding the associated bearer:

```
(deftemplate obligation extends ifact
   "Obligation definition"
   (slot bearer)
)
(deftemplate fulfilled extends ifact
   "Obligation fulfillment"
   (slot bearer)
)
(deftemplate violated extends ifact
   "Obligation violation"
   (slot bearer)
)
```

For dealing with the normative framework, we define the notion of *context*, together with appropriate templates for dealing with different kinds of contractual structures. A context has a creation date and points to a super-context from which norms can be inherited:

```
(deftemplate context
   "Contexts for norms"
   (slot super-context (default MAIN))
   (slot id)
   (slot when)
)
```

A cooperation agreement extends the `context` definition by indicating the multiple participants, their cooperation efforts, and the business process to be implemented:

```
(deftemplate coop-agreement extends context
   "Virtual Organization cooperation agreement"
   (multislot who)
)
(deftemplate coop-effort
   "Cooperation effort of a VO partner"
   (slot context)
```

```
   (slot agent)
   (slot resource)
   (slot max-qt)
   (slot unit-price)
)
(deftemplate business-process
   "Resource flow between VO partners"
   (slot context)
   (slot from)
   (slot resource)
   (slot to)
)
```

An operational contract is a context involving two of the VO partners. It will have a cooperation agreement as its super-context:

```
(deftemplate op-contract extends context
   "Operational contract"
   (slot purchaser)
   (slot supplier)
   (slot resource)
   (slot quantity)
   (slot unit-price)
)
```

A purchase contract involves a vendor and a customer, the item sold and its price:

```
(deftemplate purchase-contract extends context
   "Purchase contract"
   (slot vendor)
   (slot customer)
   (slot item)
   (slot price)
)
```

5.2. *Defining rules*

Because Jess is a rule-based system shell, rule definition is straightforward.

5.2.1. *Implementing constitutive rules*

Constitutive rules are implemented taking into account the `illocution`, `agent-role`, and `ifact` templates. For example:

22   *Henrique Lopes Cardoso, Eugénio Oliveira*

```
(defrule payments
   "Constitutive rule for acknowledging payments"
   (illocution (sender ?a)
         (content currency-transfered ?ctx ?fr ?am ?to ?wh))
   (agent-role (agent ?a) (role ibank))
   =>
   (assert (ifact (context ?ctx) (fact payment ?fr ?am ?to)
               (when ?wh) ) )
)
```

5.2.2. *Implementing rules for monitoring*

Rules for detecting the fulfillment or violation of obligations take into account `ifact` and `obligation` templates. A fulfillment is detected when an obligation's institutional fact comes about before the obligation's deadline:

```
(defrule detect-fulfillment
   "Detect the fulfillment of an obligation"
   (ifact (context ?c) (fact $?f) (when ?t))
   (obligation (context ?c) (bearer ?b) (fact $?f)
         (when ?deadline) )
   (test (< ?t ?deadline))
   =>
   (assert (fulfilled (context ?c) (bearer ?b) (fact $?f)
               (when ?t) ) )
)
```

A violation is detected when a time fact corresponding to a non-fulfilled obligation is reached:

```
(defrule detect-violation
   "Detect the violation of an obligation"
   (ifact (context ?c) (fact time) (when ?t))
   (obligation (context ?c) (bearer ?b) (fact $?f)
         (when ?t) )
   (not (fulfilled (context ?c) (bearer ?b) (fact $?f)))
   =>
   (assert (violated (context ?c) (bearer ?b) (fact $?f)
               (when ?t) ) )
)
```

Several additional rules may be defined in order to make the system more responsive. For instance, notifications can be sent to agents whenever new obligations arise.

5.3. *Norm organization and application*

In order to address the possible application of default norms (defined as Jess rules), we chose to group the norms applicable in a given context in a Jess module. Modules allow for organizing a system with a potentially large number of rules. In our case, each context will correspond to a module, and will point to a super-context (see subsection 5.1) from which default rules can be applied.

5.3.1. *Defining norms*

Norms are defined as Jess rules and grouped inside appropriate modules. Default rules applicable in a given context are defined inside a super-context (that is, inside another module). The `MAIN` module subsumes institutionally defined norms.

For example, we may define how to handle a simple purchase contract by default. We may start by saying that when the contract's initial time is reached, an obligation to deliver the item is added:

```
(defrule MAIN::pc-initial-obligation-to-deliver
   "Start with an obligation to deliver the item"
   ?st <- (ifact (context ?ctx) (fact time) (when ?w)
               (processed FALSE) )
   (purchase-contract (id ?ctx) (when ?w) (customer ?cus)
         (vendor ?ven) (item ?it) )
   =>
   (assert (obligation (context ?ctx) (bearer ?ven)
               (when (+ ?w 10))
               (fact delivery ?ven ?it ?cus) ) )
   (modify ?st (processed TRUE))
)
```

After delivery, an obligation to pay for the item comes about:

```
(defrule MAIN::pc-obligation-to-pay-after-deliver
   "After delivery, create obligation to pay for the item"
   ?ff <- (fulfilled (context ?ctx) (bearer ?ven) (when ?w)
               (processed FALSE)
               (fact delivery ?ven ? ?cus) )
   (purchase-contract (id ?ctx) (customer ?cus)
         (vendor ?ven) (price ?pr) )
   =>
   (assert (obligation (context ?ctx) (bearer ?cus)
               (when (+ ?w 30))
               (fact payment ?cus ?pr ?ven) ) )
   (modify ?ff (processed TRUE))
)
```

24   *Henrique Lopes Cardoso, Eugénio Oliveira*

We may also define sanctions. If the delivery obligation is not met, maintain the obligation to deliver (with an extended deadline) and add a monetary penalty:

```
(defrule MAIN::pc-sanction-delivery-violation
   "If vendor fails delivery, add compensation obligation"
   ?vi <- (violated (context ?ctx) (bearer ?ven) (when ?w)
              (processed FALSE)
              (fact delivery ?ven ?it ?cus) )
   (purchase-contract (id ?ctx) (customer ?cus)
         (vendor ?ven) (price ?pr))
   =>
   (assert (obligation (context ?ctx) (bearer ?ven)
              (when (+ ?w 5))
              (fact delivery ?ven ?it ?cus) ) )
   (assert (obligation (context ?ctx) (bearer ?ven)
              (when (+ ?w 5))
              (fact payment ?ven (* ?pr 0.1) ?cus) ) )
   (modify ?vi (processed TRUE))
)
```

If the payment obligation is not met, increase the value due and extend the deadline:

```
(defrule MAIN::pc-sanction-payment-violation
   "If customer fails to pay, increase payment due"
   ?vi <- (violated (context ?ctx) (bearer ?cus) (when ?w)
              (processed FALSE)
              (fact payment ?cus ?pr ?ven) )
   (purchase-contract (id ?ctx) (customer ?cus)
         (vendor ?ven) (price ?pr))
   =>
   (assert (obligation (context ?ctx) (bearer ?cus)
              (when (+ ?w 15))
              (fact payment ?cus (* ?pr 1.1) ?ven) ) )
   (modify ?vi (processed TRUE))
)
```

As for VOs, we may state that every VO partner must stick to its promise, being obliged to comply with a given request:

```
(defrule MAIN::ca-stick-to-promise
   "Each partner must stick with his coop-effort promise"
   ?if <- (ifact (context ?ctx) (when ?w) (processed FALSE)
              (fact msg ?ag1 request ?res ?qt ?ag2) )
   (coop-agreement (id ?ctx))
   (coop-effort (context ?ctx) (agent ?ag2) (resource ?res)
```

```
            (max-qt ?mqt))
    (test (< ?qt ?mqt))
    (business-process (context ?ctx) (from ?ag2)
          (resource ?res) (to ?ag1))
    =>
    (assert (obligation (context ?ctx) (bearer ?ag2)
                (when (+ ?w 10))
                (fact msg ?ag2 acknowledge ?res ?qt ?ag1) ) )
    (modify ?if (processed TRUE))
)
```

Also, a request acknowledgement leads to an operational contract:

```
(defrule MAIN::ca-fulfill-promise
    "Partner fulfills promise: new op-contract"
    ?ff <- (fulfilled (context ?ctx) (bearer ?ag2) (when ?w)
                (processed FALSE)
                (fact msg ?ag2 acknowledge ?res ?qt ?ag1) )
    (coop-agreement (id ?ctx))
    (coop-effort (context ?ctx) (agent ?ag2) (resource ?res)
          (unit-price ?upr) )
    =>
    (reg-new-op-contract ?w ?ctx ?ag1 ?ag2 ?res ?qt ?upr)
    (modify ?ff (processed TRUE))
)
```

Norms regulating operational contracts can be defined similarly to those of purchase contracts.

Although these examples are defined at an institutional level, overriding norms may be defined in specific contexts representing specific contracts. It is the environment's responsibility to decide which norms apply, as follows.

5.3.2. *Applying norms*

A rule for managing the application of norms will handle the execution focus for rules defined in different modules. Jess applies rules defined in the module having focus; if no rule is applicable, the module is removed from a focus-stack, and the next module gets focus. We define a rule for building a focus-stack each time a new institutional fact is created:

```
(defrule focus-on-new-irelement-contexts
    "When a new ifact is created, create its focus stack"
    (declare (salience 100))
    (ifact (context ?c))
    (context (id ?c) (super-context ?s))
```

```
        =>
        (focus ?s) (focus ?c)
    )
```

The salience value indicates that this rule should fire before any institutional norm. That is, through this rule, priority is given to norms defined at the innermost context.

5.4. *Example and integration*

Let us suppose that three agents, representing different organizations, create a VO, which is to be regulated by institutionally defined norms. Agent A accepts customer orders for product X (made of two components Y and Z), for which it requires the cooperation of agents B and C. These are responsible for supplying components Y and Z, respectively. The following definitions implement this scenario, and include further details:

```
    (defmodule caX)
    (coop-agreement
        (id caX) (when 162345) (who A B C)
    )
    (coop-effort
        (context caX) (agent B) (resource Y) (max-qt 10)
        (unit-price 3)
    )
    (coop-effort
        (context caX) (agent C) (resource Z) (max-qt 5)
        (unit-price 8)
    )
    (business-process
        (context caX) (from B) (resource Y) (to A)
    )
    (business-process
        (context caX) (from C) (resource Z) (to A)
    )
```

Having the institutional normative background, these definitions are enough for specifying the commitments between the different partners. The example can be extended to more complex scenarios, involving an arbitrary number of agents, resources and workflow definitions. Resources can represent product components, documents, or any item that is to be exchanged between partners. Different norm sets would be appropriate for different cases.

The realization of the EI framework involves the utilization of an agent platform (such as JADE[16]), where agents interact. Besides external agents representing the involved organizations, agents performing institutional roles need to be in place,

representing messenger, banking, and delivery-tracking facilities. These will inform the agentified normative environment of the occurring events, enabling it to take into account what is going on while it is monitoring norms. The normative environment will map received messages to the corresponding illocution facts.

A number of procedures are also needed in order to make the prototype work. Jess allows for an easy integration with Java programmed components that implement such functionalities, related to the creation of new contracts, the generation of context-relevant time events, or the sending of messages from the normative environment to agents addressed by norms. Our working prototype includes all these features.

6. Conclusions and Related Work

The regulation of multi-agent systems in environments with no central design (and thus with no cooperative assumption), is gaining much attention in the research community. Normative multi-agent systems[3] address this issue by introducing incentives to cooperation (or discouraging deviation). After initial research on norms as constraints on behavior, it is now accepted that autonomous agents are able to deliberate whether to comply with norms[5].

Searle's work on speech acts[26] and institutional reality[27] has inspired several researchers within the MAS field (e.g. Ref. 2, 6 and 15). In our case, we used this inspiration as a means to certify the occurrence of real-world actions, essential for contract monitoring purposes. Brute facts consist of agents' illocutions, which according to empowering relations are used to produce institutional reality.

The concept of electronic institutions is gaining importance inside MAS research. We identify the main goals of an EI as being twofold. First, it aims at supporting agent interaction as a coordination framework, making the establishment of business agreements more efficient. Furthermore, it serves the purpose of providing a level of trust by offering an enforceable normative environment.

Within the framework of our electronic institution[19] providing services for the achievement of contractually specified agreements (including VO scenarios), we described the use of contextualized norms, the specification of contracts and the implementation of monitoring and enforcement mechanisms. Norms are typically related with the deontic notions of obligation, permission and prohibition, and have been used to formalize contracts[25,18], or to specify interaction in agent societies[10,24]. In our case, we essentially rely on obligations to specify contractual commitments. Permissions are implicit with the definition of constitutive rules, which authorize certain illocutions to certified agents. In this sense, not permitted actions will be impossible. In the case of VO contracts, permissions can be seen as rights for requesting a partner's contribution. Prohibitions may be applied as a consequence of violation detection.

Previous approaches towards regulating agent behavior through EIs include Ref. 11. However, this model formally defines institutions as having rigid performa-

tive structures (consisting of scenes and transitions among these scenes) that guide agent interactions. Complementing that approach, a middleware was developed[12] which ensures that agents obey to the predefined protocol by constraining their activity. By restricting the actions agents are allowed to perform, this model does not cope well with a central property of agency: autonomy [31]. In our approach, we avoid imposing hard constraints on behavior. By enforcing norms, we do conduct and supervise the behavior of rational agents able to take into account their pertinent norms.

Top-down normative frameworks (i.e., frameworks where norms are pre-specified and imposed) are appropriate in situations where norms can be centrally designed, although regulating a distributed environment with autonomous self-interested agents. However, these approaches are not suitable to evolving normative environments, and are thus not amenable to contract handling. In such situations, agents make agreements that are to be monitored by a trusted third-party (the EI); hence the normative structure is modified by the corresponding contracts. Our work is focused on providing such an evolving normative framework. We have shown how norm monitoring and enforcement mechanisms can be put into place, taking into account the creation of institutional reality.

We are exploiting our EI prototype, testing its capabilities as a normative framework. Being an agent-based platform, the EI can be distributed over a number of computers. However, the normative environment, within which norms are monitored, seems to be a centralized component. This may represent a performance limitation when dealing with a considerable number of agents and their contractual commitments. In future work we may study how to address this limitation by implementing a distributed normative environment.

The explicit definition of permissions and prohibitions may be easily included with our norm specification. Most importantly, in business scenarios the definition of cross-contractual prohibitions is an important issue (e.g. antitrust or competition policies). We have yet to study how inter-contextual norms can be defined. Also, the illustration of VO settings will be enriched, to account for situations closer to real-world instances (as described in Ref. 20).

References

1. Agents, Norms and Institutions for Regulated Multiagent Systems (ANIREM), *Workshop in the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS05)*, Utrecht, The Netherlands, July 25th 2005, http://platon.escet.urjc.es/ANIREM2005/.
2. G. Boella and L. van der Torre, Regulative and Constitutive Norms in Normative Multiagent Systems, in *Proc. 9th Int. Conf. Principles of Knowledge Representation and Reasoning*, eds. D. Dubois, C. Welty and M.-A. Williams (Canada, Whistler, 2004), pp.255-266.
3. G. Boella, L. van der Torre and H. Verhagen, Introduction to Normative Multiagent Systems, in *1st Int. Symposium on Normative Multiagent Systems* (England, Hatfield, 2005).

4. C. Castelfranchi, Engineering Social Order, in *Engineering Societies in the Agents World*, eds. A. Omicini, R. Tolksdorf and F. Zambonelli (Springer, 2000), pp.1-18.
5. C. Castelfranchi, F. Dignum, C. Jonker and J. Treur, Deliberative Normative Agents: Principles and Architectures, in *Intelligent Agents VI: Agent Theories, Architectures, and Languages*, eds. N. Jennings and Y. Lesperance (Springer, 2000), pp. 364-378.
6. M. Colombetti and M. Verdicchio, An analysis of agent speech acts as institutional actions, in *Proc. 1st Int. Joint Conf. on Autonomous Agents and Multiagent Systems*: part 3 (ACM Press, New York, 2002), pp.1157-1164.
7. R. Conte, R. Falcone and G. Sartor, Introduction: Agents and Norms: How to fill the gap?, *Artificial Intelligence and Law*, **7**(1) (1999), 1-15.
8. R. Craswell, Contract Law: General Theories, in *Encyclopedia of Law and Economics, Volume III: The Regulation of Contracts*, eds. B. Bouckaert and G. De Geest (Edward Elgar, Cheltenham, 2000), pp.1-24.
9. V. Dignum and F. Dignum, Towards an Agent-based Infrastructure to Support Virtual Organizations, in *Collaborative Business Ecosystems and Virtual Enterprises*, ed. L. M. Camarinha-Matos (Kluwer, 2002), pp.363-370.
10. V. Dignum, J.-J. Meyer, F. Dignum and H. Weigand, Formal Specification of Interaction in Agent Societies, in *Formal Approaches to Agent-Based Systems*, eds. M. Hinchey, J. Rash, W. Truszkowski, C. Rouff and D. Gordon-Spears (Springer, 2003), pp.37-52.
11. M. Esteva, J. Padget and C. Sierra, Formalizing a language for institutions and norms, in *Intelligent Agents VIII*, eds. J.-J. Meyer and M. Tambe (Springer, 2002), pp.348-366.
12. M. Esteva, B. Rosell, J. A. Rodríguez-Aguilar and J. L. Arcos, AMELI: An Agent-based Middleware for Electronic Institutions, in *Proc. 3rd Int. Joint Conf. on Autonomous Agents and Multi-agent Systems*: volume 1 (IEEE Computer Society, New York), pp.236-243.
13. Foundation for Intelligent Physical Agents, *FIPA-ACL Message Structure Specification* (2002), http://www.fipa.org/specs/fipa00061/.
14. Foundation for Intelligent Physical Agents, *FIPA Communicative Act Library Specification* (2002), http://www.fipa.org/specs/fipa00037/.
15. N. Fornara, F. Viganò and M. Colombetti, Agent Communication and Institutional Reality, in *Agent Communication: International Workshop on Agent Communication*, eds. R. M. van Eijk, M.-P. Huget and F. Dignum, Lecture Notes in Computer Science, Volume 3396 (Springer, 2005), pp.1-17.
16. JADE, Java Agent DEvelopment Framework: http://jade.tilab.com/
17. JESS, the Rule Engine for the Java Platform: http://herzberg.ca.sandia.gov/jess/
18. M. J. Kollingbaum and T. J. Norman, Supervised Interaction – Creating a Web of Trust for Contracting Agents in Electronic Environments, in *Proc. 1st Int. Joint Conf. on Autonomous Agents and Multi-agent Systems*: part 1 (ACM Press, New York, 2002), pp. 272-279.
19. H. Lopes Cardoso, A. Malucelli, A.P. Rocha and E. Oliveira, Institutional Services for Dynamic Virtual Organizations, in *Collaborative Networks and Their Breeding Environments – 6th IFIP Working Conference on Virtual Enterprises (PRO-VE'05)*, eds. L.M. Camarinha-Matos, H. Afsarmanesh and A. Ortiz (Springer, 2005), pp.521-528.
20. H. Lopes Cardoso and E. Oliveira, Virtual Enterprise Normative Framework within Electronic Institutions, in *Engineering Societies in the Agents World V*, eds. M.-P. Gleizes, A. Omicini e F. Zambonelli (Springer, 2005), pp.14-32.
21. F. López y López and M. Luck, Modelling Norms for Autonomous Agents, in *Proc.*

4th Mexican Int. Conf. on Computer Science, eds. E. Chavez, J. Favela, M. Mejia and A. Oliart (IEEE Computer Society, 2003), pp.238-245.

22. N. Maudet and B. Chaib-draa, Commitment-based and dialogue-game based protocols – new trends in agent communication languages, *Knowledge Engineering*, **17**(2) (2002), 157-179.

23. E. Oliveira and A. P. Rocha, Agents Advanced Features for Negotiation in Electronic Commerce and Virtual Organisations Formation Process, in *Agent Mediated Electronic Commerce: The European AgentLink Perspective*, eds. F. Dignum and C. Sierra (Springer, 2000), pp.78-97.

24. O. Pacheco and J. Carmo, A Role Based Model for the Normative Specification of Organized Collective Agency and Agents Interaction, *Journal of Autonomous Agents and Multi-Agent Systems*, **6**(2) (2003), 145-184.

25. M. Sallé, Electronic Contract Framework for Contractual Agents, in *Advances in Artificial Intelligence: 15th Conference of the Canadian Society for Computational Studies of Intelligence*, eds. R. Cohen and B. Spencer (Springer, 2002), pp.349-353.

26. J. R. Searle, *Speech Acts: an Essay in the Philosophy of Language* (Cambridge University Press, Cambridge, 1969).

27. J. R. Searle, *The Construction of Social Reality* (Free Press, New York, 1995).

28. J. Vázquez-Salceda, H. Aldewereld and F. Dignum, Implementing norms in multiagent systems, in *Multiagent System Technologies*, eds. G. Lindemann, J. Denzinger, I. J. Timm and R. Unland (Springer, 2004), pp.313-327.

29. J. Vázquez-Salceda and F. Dignum, Modelling Electronic Organizations, in *Multi-Agent Systems and Applications III*, eds. V. Marik, J. Muller and M. Pechoucek (Springer, 2003), pp.584-593.

30. A. Walker and M. Wooldridge, Understanding the Emergence of Conventions in Multi-Agent Systems, in *Proc. 1st Int. Conf. on Multi-Agent Systems*, eds. V. R. Lesser and L. Gasser (MIT Press, 1995), pp.384-389.

31. M. Wooldridge and N. R. Jennings, Intelligent Agents: Theory and Practice, *Knowledge Engineering Review*, **10**(2) (1995), 115-152.