Formalising Agent Mediated Electronic Institutions

Marc Esteva, Juan A. Rodríguez-Aguilar, Josep Ll. Arcos, Carles Sierra, Pere Garcia
Artificial Intelligence Research Institute, IIIA
Spanish Council for Scientific Research, CSIC
08193 Bellaterra, Barcelona, Spain.
{marc,jar,arcos,sierra, pere}@iiia.csic.es
http://www.iiia.csic.es

Abstract

In this paper we argue that open agent organisations can be effectively designed and implemented as institutionalised electronic organisations (*electronic institutions*) composed of a vast amount of heterogeneous (human and software) agents playing different roles and interacting by means of illocutions. Here we take the view that the design and development of electronic institutions must be guided by a principled methodology. Along this direction, we advocate for the presence of an underlying formal method that underpins the use of structured design techniques and formal analysis, facilitating development, composition and reuse. For this purpose we propose a specification formalism for electronic institutions that founds their subsequent design, analysis and development.

Paraules clau: Sistemes multiagent, IA distribuïda

1 Introduction

Human organisations define the roles and responsibilities for organisational participants, who are expected to bring those into action depending on the task and environmental demands. Early work in DAI identified the advantages of organisational structuring as one of the main issues in order to cope with the complexity of designing DAI systems [6, 11, 3, 15]. However, DAI research, and MAS research in particular, have traditionally kept an individualistic character, and have focused on the principled construction of individual agents following an agent-centered view.

Nonetheless, recently there is a growing interest in incorporating organisational concepts into multi-agent systems, with the purpose of considering organisation-centered designs of multi-agent systems. For instance, in [16] we find a methodology for agent-oriented design and analysis founded on the view of a system as a computational organisation consisting of various interacting roles. Likewise [6], [1] and [8] propose agent development languages that allow to model organisations.

However, these organisational approaches do not conveniently handle the issues inherent to open multi-agent systems[7], namely *heterogeneity* of agents; *trust and accountability*; *exception handling* (detection, prevention and recovery from failures that may jeopardise the global operation of the system); and *societal change* (capability of accommodating structural changes).

As noted in [12, 9, 4], human societies have successfully coped with similar issues, by creating *institutions* that set and enforce laws, monitor and respond to emergencies, prevent and recover from disasters, etc. In this work we propose a similar approach for advocate for the adoption of a mimetic strategy for the realisation of open multi-agent systems founded on the deployment of *electronic institutions.*

Since we regard electronic institutions as highly complex and critical , we take the view that their design and development must be guided by a principled methodology. In this sense, we advocate for introducing formal methods that underpin the specification, analysis and development of electronic institutions. Thus, in this paper we propose a specification formalism for electronic institutions.

The rest of the paper is organised as follows. In Section 2 we describe a case study that will be employed for illustrative purposes and which have also motivated our experience in the development of practical electronic institutions, namely an agent-mediated auction house[12]. In Section 3 we introduce the core notions of an electronic institution. Next Section 4 details our formal view of electronic institutions. Finally some conclusions are presented in Section 5.

2 Actual-world Institutions. Case Studies

As a case study we choose a traditional trading institution: the fish market auction house. The actual fish market —and other similar auction houses— can be described as a place where several *scenes* take place simultaneously, at different places, but with some causal continuity. Each scene involves various agents who at that moment perform well-defined functions. These agents are subject to the accepted market conventions, but they also have to adapt to whatever has happened and is happening at the auction house at that time. The principal scene is the auction itself, in which buyers bid for boxes of fish that are presented by an auctioneer who calls prices in descending order —the downward bidding protocol. However, before those boxes of fish may be sold, fishermen have to deliver the fish to the fish market (in the *sellers' admission scene*) and buyers need to register for the market (at the *buyers' admission scene*). Likewise, once a box of fish is sold, the buyer should take it away by passing through a *buyers' settlements scene*, while sellers may collect their payments at the *sellers' settlements scene* once their lot has been sold.

The main activity within the marketplace is the auctioning of goods in the auction room. This scene is governed by the auctioneer making use of the downward bidding protocol (DBP). With a chosen good, the auctioneer opens a *bidding round* by quoting offers downward from the good's starting price, previously fixed by the sellers' admitter, as long as these price quotations are above a *reserve price* set by the seller. For each price called by the auctioneer, several situations might arise during the open round.

First, several buyers submit their bids at the current price. In this case, a collision occurs and the auctioneer restarts the round at a higher price. Nevertheless, the auctioneer tracks whether a given number of successive collisions is reached, in order to avoid an infinite collision loop. Second, if only one buyer submits a bid at the current price, the good is sold to him if his credit can support his bid. In case there is an unsupported bid the round is restarted at a higher price, the unsuccessful bidder is punished with a fine, and he is expelled out from the auction room unless such fine is paid off. Otherwise, if no buyer submits a bid at the current price, and the reserve price has not been reached yet, the auctioneer quotes a new price obtained by decreasing the current price according to the price step. If the reserve price is reached, the auctioneer declares the good *withdrawn* and closes the round.

3 Electronic Institutions. Fundamental Concepts

The role of any formal method is to provide a clear and precise description of *what* a system is supposed to do, rather than a description of *how* it operates [5]. The presence of an underlying formal model will support the use of structured design techniques and formal analysis, facilitating development, composition and reuse.

In this section we take such formal approach with the purpose of specifying software infrastructures for electronic institutions. Our formal specification will be based on a purposely devised specification language enabled to produce graphical specifications of infrastructures for electronic institutions. In other words, such specification language will serve to produce a sound and unambiguous specifications of the *rules* of an electronic institution.

We will solely focus on the specification of infrastructures. The core notions of an electronic institution include:

- *Agents and Roles.* Agents are the players in an electronic institution, interacting by the exchange of illocutions, whereas roles are defined as standardised patterns of behaviour. The identification and regulation of roles is considered as part of the formalisation process of any organisation [13]. Any agent within an electronic institution is required to adopt some role(s). As dialogic actions are associated to roles, an agent adopting a given role is allowed to perform the actions associated to that role. A major advantage of using roles is that they can be updated without having to update the actions for every agent on an individual basis.
- *Dialogic framework.* The context or framework of interaction amongst agents of an institution, such as the objects of the world and

the language employed for communicating, are fixed. In a dialogic institution, agents interact through illocutions. Institutions establish the acceptable illocutions by defining the ontology (vocabulary) —the common language to represent the "world"— and the common language for communication and knowledge representation. All of these contextual features are bundled together in what we call dialogic framework. By sharing a dialogic framework, we enable heterogeneous agents to exchange knowledge with other agents.

- *Scene.* Interactions between agents are articulated through agent group meetings, which we call *scenes*, with a well-defined communication protocol. We consider the protocol of a scene to be the possible dialogues agents may have.
- *Performative structure.* Scenes can be connected, composing a network of scenes (the so-called performative structure) which captures the existing relationships among scenes. The specification of a peformative structure contains a description of how the different roles can legally move from scene to scene. A performative structure is to contain the multiple, simultaneous ongoing activities, represented by scenes. Agents within a performative structure may participate in different scenes at the same time with different roles.
- *Normative Rules.* Agent actions in the context of an institution may have consequences that either limit or enlarge its subsequent acting possibilities. Such consequences will impose obligations to the agents and affect its possible paths within the performative structure.

Next, in the next section, we offer a detailed analysis of the notions introduced above that helps construct a formal specification of an electronic institution infrastructure.

4 Electronic Institutions. Structure

Let $Agents = \{a_1, \ldots, a_n\}$ and $Roles = \{r_1, \ldots, r_m\}$ represent a finite set of agent and role identifiers, respectively. At the specification level we regard agents and roles simply as symbols and type names. Then by $a_i : r_j$ we mean that agent a_i is of type r_j. Moreover, we define the sets $V_A = \{x_1, \ldots, x_A\}$ and $V_R = \{\rho_1, \ldots, \rho_R\}$ as a finite set of agent and role variables respectively.

4.1 Roles

We have already identified the notion of role as central in the specification of electronic institutions. Roles allow us to abstract from the individuals, in the activities of an institution. In order to take part in an electronic institution, an agent is obliged to adopt some role(s). Thereafter, an agent playing a given role must conform to the pattern of behaviour corresponding to that particular role. More precisely, we define a role as a finite set of dialogic actions.

In human institutions roles played are hierarchically organised. There are many ways of constructing a role hierarchy to express different types of relationships among roles. Here we consider that roles are organised as a partially ordered set (poset), represented as a pair $\mathcal{R} = \langle Roles, \succeq \rangle$, reflecting a role hierarchy. The relation $\succeq \subseteq Roles \times Roles$ is reflexive, antisymmetric, and transitive. Then if $r \succeq r'$ holds we say that r subsumes r'. We interpret this relationship as: an agent playing role r is also enabled to play role r'.

We also realise that there are conflicting roles within an institution. For instance, in the fish market the *boss* and *buyer* roles are mutually exclusive in the sense that no one can *ever* act as the boss of the market and as a buyer. We define a policy of static separation of duty (ssd) to mean that roles specified as mutually exclusive cannot be both authorised to an agent. The ssd protects the institution against agent's malicious behaviour.

We represent the static separation of duties as the relation $ssd \subseteq Roles \times Roles$. A pair $(r, r') \in ssd$ denotes that r, r' cannot be authorised to the very same agent.

Finally, we explicitly state the requirements identified so far as necessary for managing the role/role relationships:

(i) The static separation of duties relation is symmetric. Formally, for all $r_i, r_j \in Roles \quad (r_i, r_j) \in ssd \Rightarrow (r_j, r_i) \in ssd$.

(ii) If a role subsumes another role, and that role is in static separation of duties with a third role, then the first role is in static separation of duties with the third one. Formally, for all $r, r_i, r_j \in Roles, r \succeq r_i, (r_i, r_j) \in ssd \Rightarrow (r, r_j) \in ssd$.

(iii) If a role subsumes another role, then the two roles cannot be in static separation of duties. Formally, $r_i \succeq r_j \Rightarrow (r_i, r_j) \notin ssd$.

4.2 Dialogic Framework

In the most general case, each agent immersed in a multi-agent environment is endowed with its own inner language and ontology. In order to allow agents to successfully interact with other agents we must address the fundamental issue of putting their languages and ontologies in relation. For this purpose, we propose that agents share what we call the *dialogic framework* [10], composed of a communication language, a representation language for domain content and an ontology. By sharing a dialogic framework, we enable heterogeneous agents to exchange knowledge with one another.

Definition 4.1 *We define a* dialogic framework *as a tuple $DF = \langle O, L, I, CL, Time \rangle$ where O stands for an ontology (vocabulary); L stands for a representation language for domain content; I is the set of illocutionary particles; CL is the (agent) communication language; and $Time$ is a discrete and partially ordered set of instants.*

Within a dialogic framework the representation language allows for the encoding of the knowledge using the vocabulary offered by the O ontology. The propositions built with the aid of L, the "inner" language, are embedded into an "outer language", CL, which expresses the intentions of the utterance by means of the illocutionary particles in I. We take this approach in accordance to speech act theory [14], which postulates that utterances are not simply propositions that are true or false, but attempts on the part of the speaker that succeed or fail.

We consider that CL *expressions* are constructed as formula of the type $\iota(\alpha_i : \rho_i, \alpha_j : \rho_j, \varphi, \tau)$ where $\iota \in I, \alpha_i$ and α_j are terms which can be either agent variables or agent identifiers, ρ_i and ρ_j are terms which can be either role variables or role identifiers, $\varphi \in L$ and τ is a term which can be either a time variable or a value in $Time$. We say that a CL expression is an *illocution* when α_i, α_j are agent identifiers, ρ_i and ρ_j are role identifiers and τ is a value in $Time$ time-stamping the illocution. Furthermore, a CL expression is an *illocution scheme* when some of the terms corresponds to a variable. This distinction will be valuable when specifying scenes in the following section.

Consider the following examples corresponding to a CL expressions based on the case study in Section 2:

- $request(?auc : auctioneer, ?b_i : buyer, bid(?price), ?t)$ is an illocution scheme.
- $refuse(buyer\#KQLAT, auctioneer, bid(1\$), 5)$ is an illocution, sent by buyer $buyer\#KQLAT$ to the auctioneer agent *auctioneer* refusing the proposal to bid at 1$.

Finally, we would like to stress the importance of the dialogic framework as the component containing the ontologic elements on the basis of which any agent interaction can be specified. Thus, a dialogic framework must be regarded as a necessary ingredient to specify scenes.

4.3 Scene

Activities within an electronic institution was described above as a composition of multiple, well-separated, and possibly concurrent, dialogic activities, each one involving different groups of agents playing different roles. For each activity, interactions between agents are articulated through agent group meetings, which we call *scenes*, that follow well-defined protocols. In fact, no agent interaction within an institution takes place out of the context of a scene. We consider the protocol of each scene to model the possible dialogic interactions between roles instead of agents. In other words, scene protocols are patterns of multi-role conversation.

A distinguishing feature of scenes is that they allow agents either enter or leave a scene at some particular moments(states) of an ongoing conversation. For this purpose, a set of acces and exit states is defined for each role.

Definition 4.2 *Formally, a scene is a tuple $S = \langle \mathcal{R}_s, DF, W, w_0, W_f, (WA_r)_{r \in R}, (WE_r)_{r \in R}, \Theta, \lambda, min, Max \rangle$ where $\mathcal{R}_s$ is a role poset; DF is a dialogic framework; W is the non-empty set of scene states; $w_0 \in W$ is the initial state; $W_f \subseteq W$ stands for the set of final states; $(WA_r)_{r \in R} \subseteq W$ is a family of non-empty sets such that WA_r stands for the set of access states for the role $r \in R$; $(WE_r)_{r \in R} \subseteq W$ is a family of non-empty sets such that WE_r stands for the set of exit states for*

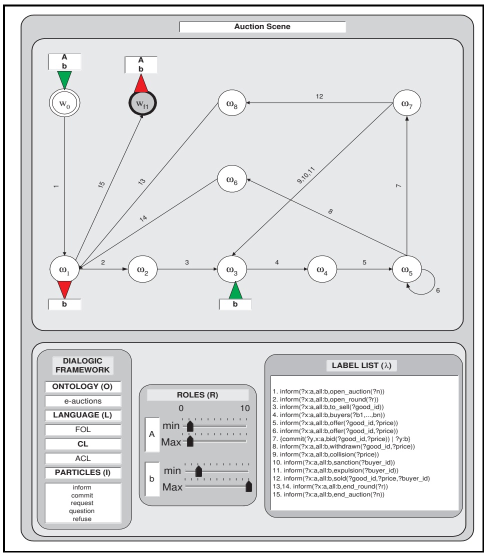

Figure 1: Graphical Specification of an Auction Scene

the role $r \in R$; $\Theta \subseteq W \times W$ is a set of directed edges; $\lambda : \Theta \longrightarrow CL$ is a labelling function; and $min, Max : R_s \longrightarrow I\!N$ $min(r)$ and $Max(r)$ return respectively the minimum and maximum number of agents that must and can play the role $r \in R$;

Notice that not every illocution scheme is valid to label an arc. In general, a CL expression $\iota(\alpha_i : \rho_i, \alpha_j : \rho_j, \varphi, \tau)$ can label an arc if it satisfies that α_i and α_j are agent variables; τ is a time variable; and ρ_i and ρ_j are either role variables or role identifiers in R.

Next, we describe an example of a scene extracted from the case study presented in Section 2.

4.3.1 Graphical Representation

The purpose of the formal definition above is to give a a mathematically sound definition of scene. However in practice scenes will be graphically specified. For instance, figure 1 depicts the graphical specifications of the scene presented above.

For the *auction* scene (see Figure 1), the specification of the role set requires the participation of exactly one auctioneer(A) and, at least, two buyers(b)(min), although up to ten buyers might be allowed to participate(Max). The graph depicts the states of the scene, along with the edges representing the legal transitions between scene states, and labelled with illocution schemes of the communication language of the dialogic framework. The information contents of such schemes is expressed in first-order logic (FOL) making use of the concepts in the *e-auctions* ontology. The type of performative is specified in the set I listed in the *Particles* area.

Notice that some states are identified as access and exit states. Apart from the initial and final states, the w_1 state is labelled as an access and exit state for buyers —meaning that between rounds buyers can leave and new buyers might be admitted into the scene— while ω_3 is uniquely an access state.

4.4 Performative Structure

The notion of performative structure is the most complex and interesting of this formalism, since it models the relationships among scenes. Although conversations (scenes) are currently admitted as the unit of communication between agents, limited work has been done concerning the modelling of the relationships among different scenes. This issue arises when conversations are embedded in a broader context of organisations and institutions making it necessary to capture the relationships among scenes. Thus, while a scene models a particular multi-agent dialogic activity, more complex activities can be specified by establishing relationships among scenes that allow us to:

- capture *causal dependencies* among scenes (f.i. a patient cannot undergo an operation without being previously diagnosed by a doctor);
- define *synchronisation* mechanisms involving scenes (f.i. within an exchange house, we might synchronise traders before allowing them to comment a negotiation scene);
- establish *parallelism* mechanisms involving scenes (f.i. in an auction house, several auctions might be run at the same time);
- define *choice points* that allow roles leaving a scene to choose their destination (f.i. an agent attending a conference is expected to opt for one of various, simultaneous talks);
- establish the *role flow policy* among scenes, i.e. which paths can be followed by the roles leaving a scene and which target scenes they can reach. For example, in a conference center, a speaker, after finishing off his talk, is permitted to leave the conference room and make it

for another conference room to attend an ongoing talk. Notice that, for this particular case, the migration of this speaker also involves the adoption of another role.

In general, the activity represented by a performative structure can be depicted as a collection of multiple, concurrent scenes. At execution time, a performative structure becomes populated by agents that make it evolve whenever agents comply with the rules encoded by the specification. Thus, an agent participating in the execution of a performative structure devotes his time to jointly start new scene executions, to enter active scenes where the agent interacts with other agents, to leave active scenes to possibly enter other scenes, and finally to abandon the performative structure.

Moreover, the very same agent can be possibly participating in multiple scenes at the same time. Thus, given this complexity of activities, there is a need for a formal specification of performative structures that is expressive enough to facilitate the specification of such rules.

Notice that the way agents move from scene to scene depends on the type of relationship holding among the source and target scenes. As mentioned above, sometimes we might be interested in forcing agents to synchronise before enter either new or existing scene executions, or offer choice points so that an agent can decide which target scene to enter, and so on. Thus, in order to capture the type of relationships listed above we consider that any performative structure contains a special type of scene, the so-called *transition* scenes[1], designed to mediate different types of connections among scenes. Scenes and transitions are connected by means of *arcs*. Each scene may be connected to multiple transitions, and in turn each transition may be connected to multiple scenes. In both cases, the connection between a scene and a transition is made by means of a directed arc. Notice that there is no direct connection between two scenes, or, in other words, all connections between scenes are mediated by transitions.

Agents move from a scene instance (execution) to another by traversing the transition connecting the scenes and following the arcs that connect transitions and scenes. The difference between transition scenes and the rest of the scenes is that transitions must be regarded as a kind of routers that contain local information about the scene instances which they connect. Therefore, instead of modelling some activity, they function to route agents towards their destinations in different ways, depending on the type of transition. Thus, within a transition agents will be informed about current active scenes that they can reach and request for their destination (i.e. either some active scene to join or some scene to be started). Then the ontology of these scenes will be composed by the elements of the performative structure.

The arcs connecting transitions to scenes also play a fundamental role. Notice that as there might be multiple (or perhaps none) scene executions of a target scene, there is a need to specify whether the agents following the arcs are allowed to start a new scene execution, whether they can choose a single or a subset of scenes to incorporate into, or whether they must enter all the available scene executions.

We define a set of different types of transitions and arcs whose semantics will be constrain the mobility of agents among the scene instances (the ongoing activities) of a performative structure. The differences between the diverse types of transitions that we consider are based on how they allow the agents to progress towards other scenes. Transition are divided into two parts: the *input*, through which it receives agents from the incoming arcs, and the *output*, through which agents leave following the outgoing arcs towards other scenes. Then, the following classification of transitions is based on the behaviour that they exhibit on their input and output sides:

- **And/And**: They establish synchronisation and parallelism points since agents are forced to synchronise at their input to subsequently follow the outgoing arcs in parallel.
- **Or/Or**: They behave in an asynchronous way at the input (agents are not required to wait for others), and as choice points at the output (agents are permitted to select which outgoing arc to follow when leaving).
- **And/Or**: A hybrid of the two types of transitions above: on the one hand, they force agents to synchronise on the input side, while on the other hand, they permit agents to individually make the choice of which path to follow when leaving.

[1] Henceforth *transitions* for shorter. Although transitions are a particular class of scenes, hereafter we will use the terms separately to distinguish non-transition scenes from transition scenes.

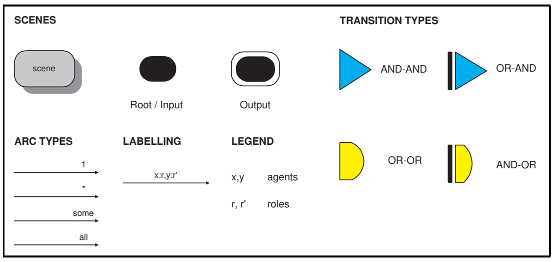

Figure 2: Graphical Elements of a Performative Structure

- **Or/And**: Also a hybrid of *and/and* and *or/or* transitions. Agents are not required to wait for others on the input side, but they are forced to follow all the possible outgoing arcs.

According to this classification, we define $\mathcal{T} = \{and/and, or/or, and/or, or/and\}$ as a set of transition types.

Depending on the path followed by the agents when traversing a transition, they may either start scenes or enter one or more ongoing scenes. Thus, there are also different types of arcs, for reaching scenes after traversing transitions. We define $\mathcal{E} = \{1, some, all, *\}$ as the set of arc types. Following a *1-arc* constrains agents to enter a single scene instance of the target scene, whereas a *some-arc* is less restrictive and allows the agents to choose a subset of scene instances to enter, and an *all arc* forces the agents to enter all the scene instances to which the paths lead. Finally, a **- arc* instantiates the creation of a new scene instance of the target scene. Furthermore, each arc is labelled with the collection of roles that are allowed to follow the arc.

Figure 2 depicts the graphical representations that we will employ to represent the performative structures components introduced so far. From the point of view of the modeller, such graphical components are the pieces that serve to construct graphical specifications of performative structures.

Before stating a concrete definition of performative structure, there is a last element to be considered. Notice that although two scenes may be connected by a transition, the eventual migration of agents from a source scene instance to a target scene instance not only depends on the role of the agents but also on the results achieved by agents in the previous scenes. Thus, in the fish market for instance, although a registration scene is connected to an auction scene, the access of a buying agent to the execution of the auction scene is forbidden if it has not successfully completed the registration process. Hence arcs connecting scenes and transitions require constraints, which agents are required to satisfy when attempts to reach a destination scene.

Definition 4.3 *Formally, given a meta-language ML, a performative structure is a tuple $PS = \langle S, T, s_0, s_\Omega, E, f_L, f_T, f_S, f_E, C, \mu \rangle$ where*

- *S is a finite, non-empty set of scenes;*
- *T is a finite and non-empty set of transitions;*
- *$s_0 \in S$ is the* root *scene;*
- *$s_\Omega \in S$ is the* output *scene;*
- *$E = E^I \bigcup E^O$ is a set of arc identifiers where $E^I \subseteq (WE_S) \times T$ is a set of edges from exit states of scenes to transitions where $WE_S = \bigcup_{s \in S} \bigcup_{r \in R} WE_r^s$, and $E^O \subseteq T \times S$ is a set of edges from transitions to scenes;*
- *$f_L : E \longrightarrow 2^{V_A \times R}$ is the labelling function;*
- *$f_T : T \longrightarrow \mathcal{T}$ maps each transition to its type;*
- *$f_E : E \longrightarrow \mathcal{E}$ maps each arc to its type;*
- *$C : E \longrightarrow ML$ maps each arc to a meta-language expression of type boolean, i.e. a predicate, representing the arc's constraints;*
- *$\mu : S \longrightarrow I\!\!N$ sets the upper bound on the number of allowed simultaneous scenes for the scene scheme represented by each scene node.*

From the definition above, a performative structure can be viewed as a network of scenes interconnected by different types of transitions. The specification of a performative structure amounts to select a collection of scenes, creating sound connections among them, and defining the limits ,μ, on the number of concurrent executions of each scene. Figure 3 shows how the graphical elements in Figure 2 are combined in order to produce the graphical specification of the performative structure of the fish market. Note that the scene depicted in Figure 1 appears also in the resulting specification as a particular node.

Any performative structure is required to contain a root and an output scene. The output scene does not model any activity, and so it must be regarded as the exit point of the performative structure. On

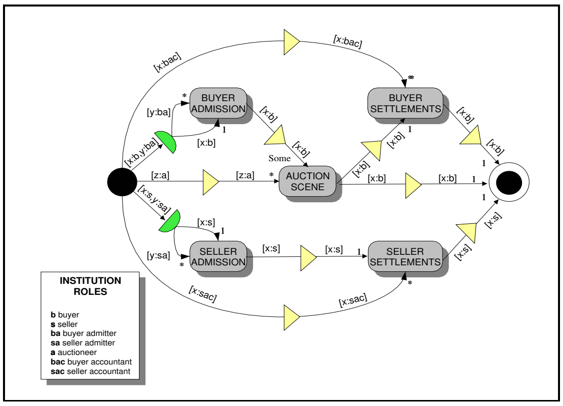

Figure 3: Graphical Specification of the Fishmarket Performative Structure.

the other hand root scene is regarded as the starting point of any agent accessing the performative structure. Departing from the root scene, agents will make for other scenes within the performative structure.

If we look at the example in figure 3 we can see that there are five scenes apart from the root and the output scene. We can observe the different paths that agents can follow depending on their role. Each scene is created by an institutional agent: buyer admitter, seller admitter, auctioneer, buyer accountant and seller accountant. On one hand, buyers after enter in the root scene have to be admitted at the buyer admission scene. If they are admitted they can go to the auction scene to participate in the auctions. Leaving the auction scene they can go to the buyer settlements scene in order to pay, if they have won any round or leave directly to the output scene. On the other hand, sellers after enter in the root scene may go to the seller admission scene where they can deliver the goods to be auctioned. From this scene the can go to the seller settlements where they receive the money for their good once sold in the auction room. They can leave the institution from this scene reaching the output scene.

4.5 Normative Rules

A peformative structure constrains the behaviour of any participating agents at two levels:

(i) *intra-scene:* Scene protocols dictate for each agent role within a scene what can be said, by whom, to whom, and when.

(ii) *inter-scene:* The connections among the scenes of a performative structure define the possible paths that agents may follow depending on their roles. Furthermore, the constraints over output arcs impose additional limitations to the agents when attempting to reach a target scene.

In addition to this, agent's actions within a scene may have consequences that either limit or enlarge its subsequent acting possibilities out of the scope of the scene. On the one hand, some actions will introduce subsequent acting commitments that have to be interpreted as acting obligations. On the other hand, other consequences occurring locally within a scene may vary the paths that an agent can follow in the performative structure because they affect the satisfaction and dissatisfaction of the constraints labelling paths. Both types of consequences will be required to be kept by an institution for each agent on an individual basis. In general, for a given agent we shall refer to such consequences as the agent *context* within the institution.

For instance, a trading agent winning a bidding round within an auction house is obliged to subsequently pay for the acquired good. Considering the performative structure in Figure 3 this implies that the trading agent has to move at some time to the buyers' settlements scene to pay for the acquired good. Notice that although the auction scene is connected to the output scene, the path is disallowed to agents unless they fulfil their pending payments.

In order to represent the deontic notion of obligation, we set out the predicate *obliged* as follows:

- $obliged(x, \psi, s)$ = agent x is obliged to do ψ in scene s.

where ψ is taken to be an illocution scheme.

Syntactically, normative rules will be composed by illocutions and meta-language predicates:

$$(s_1, \gamma_1) \wedge \ldots \wedge (s_m \gamma_m) \wedge \phi_1 \ldots \phi_n \Rightarrow \phi_{n+1} \wedge \ldots \wedge \phi_r$$

where $(s_1, \gamma_1), \ldots, (s_m, \gamma_m)$ are pairs of illocution scenes and illocution schemes, and $\phi_1 \wedge \ldots \wedge \phi_r$ are meta-language predicates. Notice that some of these rules will be devoted to the triggering of obligations, while others will be used for inferring facts that will be subsequently employed to determine the access of an agent to other scenes.

Consider the following example that will help illustrate how normative rules are specified[2]:

(auction, commit(?x:a,?y:b, sell(?good,?price))) $\Rightarrow$
commit(?y:b,?z:c, pay(?good,?price))

If an auctioneer commits to sell the good at an auction to a given buyer, this is obliged to commit to a buyers' accountant in a *settlements* scene to pay for the good.

Overall the deployment of normative rules is motivated by the need of an institution to infer agents' obligations as well as the consequences of agents' local actions (within scenes) that have effect out of the scope of a scene.

4.6 Electronic Institution

Finally, we can define an electronic institution choosing a performative structure and defining the rest of its components. These are the institutional roles, the hierarchy between roles, the policy of duties and the normative rules. The institutional roles define a set of roles that can not be played by the external agents. They are like the workers in a human institution. In the case of the hierarchy of roles and the ssd we can to take into account that both are applied to the set of roles of the institution which are all the roles appearing in the different scenes of the performative structure.

Definition 4.4 *An electronic institution is defined as a tuple* $\langle \mathcal{PS}, IR, \succeq, ssd, N_{PS} \rangle$ *where* PS *stands for a performative structure;* IR *is a subset of roles representing the institutional roles;* $\succeq$ *stands for the hierarchy partial order over the roles;* ssd *is the set of static separation of duties between roles; and* N_{PS} *stands for a set of normative rules.*

Notice that the specification of an electronic institution must be regarded as a compositional activity to be undertaken by the institution designer. We can consider that specifications of dialogic frameworks, scenes and peformative structures are to be naturally organised into specification libraries.

Once completed a specification, it must go through a validation process that checks its well-formedness. Ultimately, if such a specification proves to be correct, its equivalent textual representation must be generated in order to be manipulated by the agents intending to participate in the institution. Moreover, the generated textual representations can be also employed for brokering purposes.

In the light of the complexity of the whole process, it is apparent the need of tools that assist the institution designer through the specification, validation, and translation of an electronic institution into a machine-readable format so that it can be easily parsed by agents.

5 Conclusions and Future Work

In this paper we have argued that open multi-agent systems can be effectively designed and constructed as electronic institutions using a formal specification language.

Furthermore, we have introduced a graphical language that allows for creating graphical specifications instead of cumbersome textual specifications. Graphical specifications are extremely easy to understand. They are similar to the informal diagrams employed by engineers and designers while designing, constructing and analysing a system.

We believe that the advantages in a formal specification (models) of electronic institutions include:

- a model allows the designer to investigate a new institution before actually constructing it;
- an explicit description of both states and actions, in contrast to most description languages which describe either the states or the actions;
- the analysis of the execution of an electronic institution model, either by means of simulation or by means of more formal analysis methods; and
- the process of creating the description and performing the analysis allows the modeller to gain a dramatically improved understanding of the modelled institution.

Once completed a specification, it must go through a validation process that checks its well-formedness. Ultimately, if such a specification proves to be correct, its equivalent textual representation must be generated in order to be manipulated by the agents intending to participate in the institution. In the light of the complexity of the whole process, it is apparent the need of tools that

[2] a, b, c stand respectively for the auctioneer, buyer and accountant roles.

assist the institution designer through the specification, validation, and translation of an electronic institution into a machine-readable format.

6 Acknowledgements

Marc Esteva enjoys the CIRIT doctoral scholarship 1999FI-00012. The research reported in this paper is supported by the ESPRIT LTR 25500-COMRIS *Co-Habited Mixed-Reality Information Spaces* project, the SLIE project IST-1999-10948 and the Spanish CICYT project SMASH, TIC96-1038-C04001.

References

[1] Mihai Barbuceanu, Tom Gray, and Serge Mankovski. Coordinating with obligations. In *Proceedings of the Third International Conference on Autonomous Agents (AGENTS'99)*, pages 62–69, 1998.

[2] P. R. Cohen and H. J. Levesque. Communicative actions for artificial agents. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 65–72, Menlo Park, CA., jun 1995. AAAI Press.

[3] Daniel David Corkill and Victor Lesser. The use of meta-level control for coordination in a distributed problem solving network. In Alan H. Bond and Les Gasser, editors, *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, pages 748–756. Karlsruhe, Federal Republic of Germany, Morgan Kaufmann Publishers, August 1983.

[4] Chrysanthos Dellarocas and Mark Klein. Civil agent societies: Tools for inventing open agent-mediated electronic marketplaces. In *Proceedings ACM Conference on Electronic Commerce (EC-99)*, 1999.

[5] Antoni Diller. *Z An Introduction to Formal Methods*. John Wiley & Sons, Inc, 1990.

[6] L. Gasser, C. Braganza, and N. Herman. *Distributed Artificial Intelligence*, chapter MACE: A flexible test-bed for distributed AI research, pages 119–152. Pitman Publishers, 1987.

[7] C. Hewitt. Offices are open systems. *ACM Transactions of Office Automation Systems*, 4(3):271–287, 1986.

[8] S. Moss and B. Edmonds. A formal preference-state model with qualitative market judgements. *Omega – the International Journal of Management Science*, 25(2):155–169, 1997.

[9] Pablo Noriega. *Agent-Mediated Auctions: The Fishmarket Metaphor*. Number 8 in IIIA Monograph Series. Institut d'Investigació en Intel.ligència Artificial (IIIA), 1997. PhD Thesis.

[10] Pablo Noriega and Carles Sierra. Towards layered dialogical agents. In *Third International Workshop on Agent Theories, Architectures, and Languages, ATAL-96*, 1996.

[11] H. Edward Pattison, Daniel D. Corkill, and Victor R. Lesser. *Distributed Artificial Intelligence*, chapter Instantiating Descriptions of Organizational Structures, pages 59–96. Pitman Publishers, 1987.

[12] Juan A. Rodríguez-Aguilar, Pablo Noriega, Carles Sierra, and Julian Padget. Fm96.5 a java-based electronic auction house. In *Second International Conference on The Practical Application of Intelligent Agents and Multi-Agent Technology(PAAM'97)*, pages 207–224, 1997.

[13] W. R. Scott. *Organizations: Rational, Natural, and Open Systems*. Englewood Cliffs, NJ, Prentice Hall, 1992.

[14] J. R. Searle. *Speech acts*. Cambridge U.P., 1969.

[15] Eric Werner. *Distributed Artificial Intelligence*, chapter Cooperating Agents: A Unified Theory of Communication and Social Structure, pages 3–36. Pitman Publishers, 1987.

[16] Michael Wooldridge, Nicholas R. Jennings, and David Kinny. A methodology for agent-oriented analysis and design. In *Proceedings of the Third International Conference on Autonomous Agents (AGENTS'99)*, May 1999.