GOLDSMITHS Research Online
Conference or Workshop Item

d'Inverno, Mark, Luck, Michael, Noriega, Pablo, A., Juan and Sierra, Carles

Communicating Open Systems: Extended Abstract

You may cite this version as: d'Inverno, Mark, Luck, Michael, Noriega, Pablo, A., Juan and Sierra, Carles. 2013. 'Communicating Open Systems: Extended Abstract'. In: IJCAI 2013. Beijing, China. [Conference or Workshop Item] : Goldsmiths Research Online.

Available at: http://research.gold.ac.uk/9752/

COPYRIGHT

All material supplied via Goldsmiths Library and Goldsmiths Research Online (GRO) is protected by copyright and other intellectual property rights. You may use this copy for personal study or research, or for educational purposes, as defined by UK copyright law. Other specific conditions may apply to individual items.

This copy has been supplied on the understanding that it is copyright material. Duplication or sale of all or part of any of the GRO Data Collections is not permitted, and no quotation or excerpt from the work may be published without the prior written consent of the copyright holder/s.

http://eprints-gro.goldsmiths.ac.uk
Contact Goldsmiths Research Online at: lib-eprints@gold.ac.uk
Communicating Open Systems (Extended Abstract)*

Mark d’Inverno1, Michael Luck2, Pablo Noriega3, Juan A. Rodriguez-Aguilar3 and Carles Sierra3

1Department of Computing, Goldsmiths, University of London, United Kingdom
2Department of Informatics, King’s College London, United Kingdom
3III, Artificial Intelligence Research Institute, CSIC, Spanish National Research Council, Spain

Abstract

Just as conventional institutions are organisational structures for coordinating the activities of multiple interacting individuals, electronic institutions provide a computational analogue for coordinating the activities of multiple interacting software agents. In this paper, we argue that open multi-agent systems can be effectively designed and implemented as electronic institutions, for which we provide a comprehensive computational model. More specifically, the paper provides an operational semantics for electronic institutions, specifying the essential data structures, the state representation and the key operations necessary to implement them.

1 Introduction

Open systems—whose constituent components are not known in advance and may change over time—are becoming a de facto model for computing [Hewitt, 1986]. They are characterised by decentralised control, concurrency and loose coupling. At the same time, multi-agent systems (MAS) have emerged as a promising approach for their development [Jennings et al., 1998]. Thus, a class of open MAS populated by heterogenous, self-motivated, non-reliable agents whose internal state is not known by the system) are being used for several applications. Since they are highly complex, costly and may sustain critical applications, there is a need for principled methodologies for their specification, analysis and validation [Jennings et al., 1998; Bergenti et al., 2004]. In addition, research in MAS has shown an increasing interest in incorporating organisational concepts into MAS as well as in shifting from agent-centered to organisation-centered designs [Lopes Cardoso, 2010; Dignum, 2004; Esteva, 2003; Dellarocas and Klein, 2000; Ferber and Gutknecht, 1998; pa So and Durfee, 1998] that treat the organisation as a first-class entity, similar to the views articulated in pioneering work by Gasser [Gasser et al., 1987] and Pattison et al. [Pattison et al., 1987]. In this view, a shared organisational structure provides agents with descriptions of their roles and responsibilities in the multi-agent context and contains guidelines for their intelligent cooperation and communication.

One way of providing such organisational structure for open MAS is through electronic institutions (EI), which are a computational analogue of conventional institutions for coordinating the activities of multiple interacting (human or software) agents [Noriega, 1999; Sierra and Noriega, 1998; Rodriguez-Aguilar, 2003; Esteva, 2003]. In our original paper [d’Inverno et al., 2012] we argue that point and provide the first formal account of the EI concept in a rigorous and unambiguous way. That account includes an operational semantics for EI, specifying the essential data structures, the state representation and the key operations necessary to implement them. In this extended abstract we limit the presentation to a very broad outline of electronic institutions and a brief discussion of only one of its components, the concept of scene, to illustrate the nature of the specification.

2 An Overview of Electronic Institutions

According to [North, 1991], (conventional) institutions are artificial constraints—or “rules of the game”—that humans devise to articulate their interactions. Electronic institutions serve that same purpose in open MAS. Electronic institutions are a way of specifying and implementing a given set of conventions in order to create spaces where human and software agents may interact according to those particular conventions. Therefore, our goal in proposing electronic institutions is to specify open systems in which agent interactions are meaningful, contextual, consequential and regulated.

Take for example a traditional market where fish is auctioned, and visualise an electronic market that fulfils the same purpose. In abstract terms, the traditional market institution consists of different activities that buyers, sellers and market staff perform according to some well-known conventions. Thus, an EI that implements the fish market would involve agents that, performing different roles, come together to buy and sell fish on-line according to the corresponding conventions. In the real market, for instance, auctioning may follow a Dutch bidding protocol where prices are called in descending order and the first bidder that cries “mine” is awarded the box of fish. Hence, the electronic fish market ought to implement the same bidding protocol and thus recognise certain messages as bids and acknowledge them as valid bids if and when these messages are uttered in the appropriate context.

In fact, the conventions that define any traditional market are quite complex. Such conventions, for instance, reg-

* This paper is an extended abstract of the AI Journal publication [d’Inverno et al., 2012].
ulate not only bidding but several interconnected activities—registering goods to be sold, setting up credit lines, several auctions or negotiations taking place simultaneously, settling accounts—and in each activity conventions regulate what an individual playing a certain role may or may not do and under which circumstances. Market conventions also govern when an activity starts or ends, when an individual may become involved in—or needs to finish—an activity. Moreover, market conventions also establish the meaning and consequences of certain actions, for instance that prices are in euros, that rising a tablet is the way to place a bid, that only certain types of fish are sold, or that some quality control takes place. In order to mirror in a significant and useful manner traditional institutions, a model for electronic institutions ought to include means to account for these types of features. Alongside, the conceptual model needed to specify an EI and prescribe the way it should operate, we need to provide the computational means to implement such EI and enact it.

We propose a model for electronic institutions that represents an institution as a network of scenes (each scene corresponds to an activity), connected by transitions. Each scene, as we shall detail below, contains a protocol that regulates agent interactions which are specifically utterances (in the form of illocutions) and pauses. The EI governs those interaction by checking that certain preconditions hold for an interaction to be deemed admissible and each admissible interaction has some effects on the state of the scene, which is shared by all agents that are in that scene, and ultimately in the state of the institution (so that propagation of effects are dully taken into account in other scenes and in the future).

To support those functionalities, our model for electronic institutions involves, on one hand a set of languages and a set of data structures, and on the other, a set of operations on those data structures. Languages are needed to specify the content of utterances and movements between scenes, to define protocols, to constrain utterances and to reflect their effects in the state of the institution. Data structures are needed to represent roles and relationships, utterances, state, scenes, transitions and the network of scenes, as sketched in Figure 1.

On the other hand, to provide an operational semantics for electronic institutions, we include the set of operations of Table 1. These operations implement the basic action of “speaking” and all the other actions that either the EI itself should be able to perform (create or eliminate scenes or admit an agent in the EI, for example) and those that individual agents need to perform in order to interact (e.g., join the institution, move from a transition to a scene).

<table>
<thead>
<tr>
<th>Operation</th>
<th>Called by</th>
</tr>
</thead>
<tbody>
<tr>
<td>StartElectronicInstitution</td>
<td>Infrastructure</td>
</tr>
<tr>
<td>CreateSceneInstance</td>
<td>Infrastructure</td>
</tr>
<tr>
<td>RequestAccess</td>
<td>Agent</td>
</tr>
<tr>
<td>JoinInstitution</td>
<td>Agent</td>
</tr>
<tr>
<td>Speak</td>
<td>Agent</td>
</tr>
<tr>
<td>Timeout</td>
<td>Infrastructure</td>
</tr>
<tr>
<td>CloseSceneInstance</td>
<td>Infrastructure</td>
</tr>
<tr>
<td>SelectNewDesires</td>
<td>Agent</td>
</tr>
<tr>
<td>RemoveOldDesires</td>
<td>Agent</td>
</tr>
<tr>
<td>EnableAgentsToLeaveOrTransition</td>
<td>Infrastructure</td>
</tr>
<tr>
<td>EnableAgentsToLeaveAndTransition</td>
<td>Infrastructure</td>
</tr>
<tr>
<td>MovingFromSceneToTransition</td>
<td>Infrastructure</td>
</tr>
<tr>
<td>MoveAgentFromTransitionToScene</td>
<td>Infrastructure</td>
</tr>
<tr>
<td>LeaveInstitution</td>
<td>Agent</td>
</tr>
<tr>
<td>RemoveClosedInstances</td>
<td>Infrastructure</td>
</tr>
</tbody>
</table>

Table 1: Electronic institution operations

With those elements we provide a “general-purpose” framework allowing the specification of individual EIs for any domain, generating the corresponding institutional infrastructure that allows human or software agents to interact according to that specification. This framework may be implemented in different architectures [Esteva et al., 2008] and it has been used in several types of applications. For example, modelling of electronic markets, social opinion gathering, simulation of archaeological sites or enabling participatory simulation [Arcos et al., 2007; Osman et al., 2010; Bogdanovych and Simoff, 2011; Brito et al., 2010].

3 Scenes and Transitions

Agents interact by exchanging utterances within a group. Utterances can only occur in scenes which contain a group of participating agents, the roles they are playing, and a particular shared set of variables modelling the attributes and historical context of the scene. Without scenes, which include the history of utterances, it is not possible to interpret utterances, since previous utterances provide agents with the necessary context for interpretation. In fact, we argue that the only way to understand an utterance in a group is to understand the role it is playing in that scene, the roles the others play, and the history of what has been said up to that point.

3.1 Scenes

An electronic institution thus comprises several scenes through which agents pass, where a scene is a directed graph of places, in which the links between places correspond to actions. In this context, an action is either a line (adopting a theatre metaphor), which is an utterance with preconditions and postconditions, or a pause, which is a specific amount of time in which nothing is uttered by any agent. We choose to use preconditions because the electronic institution must verify that the agent actions (utterances) are performed in the right context, and postconditions because electronic institutions must guarantee that the consequences of agent actions
are enacted. We use time-outs as a way of protecting the institution from the inactivity of agents. Both lines and pauses move a scene from one conversation place to another and are thus links between them. For instance, an auctioneer with an opening line to start an auction moves the scene to a place at which bids can be made by buyers, while a pause of five seconds without a bid being heard moves the scene to a place at which the auctioneer can close the auction round.

To illustrate our formal model in the context of scenes, we include a snippet of the Z specification here. Note that we cannot provide details of types and data structures, but provide this merely as illustration, to whet the appetite for the full specification in [d’Inverno et al., 2012] in the language Z which has been useful specifying other multi-agent work for nearly 20 years [Luck and d’Inverno, 1995; d’Inverno et al., 2004; d’Inverno and Luck, 2012]. More specifically, the Scene schema contains: a name for the scene that enables us to identify it (line 1), such as DutchAuction; the set of role identifiers in the scene (line 2); the limits on the number of agents allowed to instantiate each role (line 3); a set of places in the conversation graph (line 4) and the moves between them (line 5) that are traversed when an action is made; the actions labelling each link that arises as a result of such moves (line 6); the initial place (line 7) and the set of possible final places (line 8); for each scene role, the set of access and leaving conversation places (line 9); and a flag to indicate whether scenes can be multiply instantiated to repeat the activity for different groups of agents (line 10).

![Figure 2: The mechanics of the Dutch auction scene](Image 343x667 to 377x699)

```
scene : SceneName
sceneroles : P RoleConst
limits : RoleConst ⇒ P1 (N)
places : P ConvPlace
moves : ConvPlace ↔ ConvPlace
link : (ConvPlace × ConvPlace) → Action
start : ConvPlace
closing : P ConvPlace
access, leaving : RoleConst ⇒ (P ConvPlace → [P ConvPlace])
multiple : Bool

∀ cs : ConvPlace | cs ≠ start ⋄
  ex ∈ (rn([start] ∪ ([movess])))
  ([start] ∪ closing) ⊆ places
  (rn access ∪ rn leaving) ⊆ places
  (dom access ∪ dom leaving) ⊆ sceneroles
  ∀ r : RoleConst; c1 : ConvPlace | c1 ∈ (access r) ⋄
  ∃ c2 : closing ⋄ (c1, c2) ∈ moves*
  dom link = moves
  (dom moves ∪ dom moves) = places
  start ∉ (rn moves)
  closing ∩ (dom moves) = {}
  ∀ s1, s2 : places ⋄ (s1, s2) ∈ (moves ∪ moves*) ⋄
  ∀ r : RoleConst; c1 : ConvPlace | c1 ∈ (access r) ⋄
  (3 c2 ∈ (leaving r) ⋄ (c1, c2) ∈ moves*)
  dom limits = sceneroles
```

Schema 1: Z specification of scenes in electronic institutions.

The predicates of the schema are as follows. First, we specify that all states are reachable from the starting place, by stating that any non-starting place is in the reflexive transitive closure of moves domain-restricted to the start place (line 11). Then, two integrity constraints require the starting place and the closing places (line 12), and access and leaving places (line 13), to be legitimate places within the scene. Clearly, only roles from the scene can be allowed to join or leave (line 14) and, for any agent accessing the scene, there must always be a path leading to a closing place (line 15). An action labels every link between places (line 16), links are defined only over places (line 17) and must not lead to the initial conversation place (line 18) nor lead out of any closing place (line 19). In all this, the directed graph of conversation places and utterances must be connected (line 20). Moreover, for any place where an agent can join (access) there must exist at least one reachable place where the agent can leave (line 21). Finally, the limits function is defined for all scene roles (line 22).

Table 2 details how the Dutch auction is instantiated in this data structure, while Figure 2 offers a graphical representation of the scene. More precisely, the scene limits participation in the auction to a single auctioneer and 100 bidders. The choice of places within the scene structure is related to the states through which the auction evolves: start auction, bidding (where bidders are allowed to bid), call price (from where the auctioneer calls a new price), winner declaration (from where the auctioneer can declare the winner), end auction (from where the auctioneer can close the auction).

The scene starts at the start auction place and finishes at the end auction place. Moreover, these are the only access and leaving states, respectively, for both the auctioneer and the buyers (so that, in this simplified example, buyers thus cannot leave until an auction has finished). The moves in the specification detail the legal moves from scene place to scene place. For instance, the auction starts by moving the scene from place start auction to place bidding. This occurs after the auctioneer utters an illocution linking these two places, namely the utterance template specified by Line1, putting an item on sale at some starting price. Note that here, and below, we do not specify the content of these lines and pauses, due to space constraints, but the meaning should be clear.

Thereafter, the scene is designed to proceed as follows. Once at the bidding place, three distinct events may occur: either no bids are received in a pre-defined time limit, a potential buyer bids but does not have enough credit, or a potential buyer with enough credit makes a bid. These three cases cause transitions to the call price, bidding, and winner declaration places respectively, and are captured by the moves con-
Finally, if the time specified by the auctioneer to close a scene. Here, the theatrical metaphor breaks down: it is our view, it is the ability of agents to operate in environments that they can understand socially, in which they can participate meaningfully, and where they can freely create and modify social processes (such as the formation of groups) that provides the foundation for social interaction and coordination. In this paper we have proposed electronic institutions as a mechanism that provides the computational means to coordinate the activities of interacting autonomous software agents immersed in open MAS. Our work details a framework that includes those specific aspects of electronic institutions that we claim are necessary to support the effective design of socially functioning open autonomous agent systems. In particular, we make explicit: (i) requirements for meaningful communication among agents; (ii) requirements to set up new social coordination or interaction processes; (iii) required operations that the institutional environment needs in order to be able to support social interactions; and (iv) operations that agents must perform in order to interoperate meaningfully and knowingly. By doing so, our model is a first approximation of the building blocks required by agreement computation in general and artificial social intelligence in particular. We have provided a concrete formal and computational model, supported by, and in support of, tools and applications that enables sophisticated social behaviour of autonomous agent systems to take place in open systems.

Acknowledgments

We thank the projects funding this work (PRAISE, ACE, COR, AT, and Generalitat of Catalunya grant 2009-SGR-1434), and Carme Roig for her tolerance, patience, and support. We dedicate this work to the memory of Marc Esteva.

3.3 Scene Network

Given the discussion of scenes, transitions and arcs, we have the basic components necessary to characterise and represent the entire institutional framework. In an auction house, for example, we need to connect together the scenes for registration, admission, auction, payment, and so on. In particular, the scene network just outlined enables us to capture the causal dependencies between scenes indicating order (which scenes must follow others), synchronisation of scenes (which scenes must finish before new ones start), parallelism (which scenes start after others finish), or choice points (which particular scene to move to when there are several options). This relates not just to the scenes themselves, but also to the transitions that enable agents playing particular roles to move between scenes. In this way, each scene may be connected to multiple transitions, and each transition to multiple scenes.

3.2 Transitions and Arcs

Electronic institutions are neutral with respect to the architecture of agents, with no understanding of their goals and motives within the institution. However, the institution needs to connect all the possible suites of activities of agents. This is often referred to as choreographing the activities of agents in workflow terminology. In order to do this we need to introduce locations where agents can wait, regroup and synchronise themselves so that the future activity of the institution can be properly choreographed. We call these locations transitions and, along with arcs that connect transitions with scenes, we can build the scene network of an institution. The resulting scene network allows groups of agents to jointly decide whether to start a new scene, join a scene, leave a scene, or close a scene. Here, the theatrical metaphor breaks down: in contrast to traditional plays in which scenes are sequential, our model permits a network of interconnected scenes in which agents can play multiple roles, even concurrently.

As indicated, arcs link scenes to transitions and transitions to scenes, with each arc associated with a set of actions from the action language and constraints from the constraint language, corresponding to preconditions that govern the ability of an agent, playing a particular role, to traverse an arc. Networking scenes in this way is necessary to capture the causal dependencies between them, including order, synchronisation, parallelism, choice points, creation, change of roles between scenes, and so on.

### Table 2: The Dutch auction scene in terms of Schema 1.

<table>
<thead>
<tr>
<th>name</th>
<th>Dutch Auction</th>
</tr>
</thead>
<tbody>
<tr>
<td>sceneroles</td>
<td>auctioneer, buyer</td>
</tr>
<tr>
<td>limits</td>
<td>start auction, bidding, call price, winner declaration, end auction</td>
</tr>
<tr>
<td>places</td>
<td>(start auction, bidding, call price, winner declaration, end auction)</td>
</tr>
<tr>
<td>moves</td>
<td>(start auction, bidding, call price, winner declaration, end auction)</td>
</tr>
<tr>
<td>start</td>
<td>start auction</td>
</tr>
<tr>
<td>closing</td>
<td>end auction</td>
</tr>
<tr>
<td>access</td>
<td>auctioneer → start auction, buyer → start auction</td>
</tr>
<tr>
<td>leaving</td>
<td>auctioneer → end auction, buyer → end auction</td>
</tr>
</tbody>
</table>

Linking these places along with their links, namely Pause1, Line4, and Line5: Pause1 updates the auction price, Line4 verifies that a buyer has insufficient credit, and Line5 verifies that a buyer does have enough credit. If a buyer does have enough credit, the scene specifies a move to place winner declaration, otherwise there is a move that returns the scene to the same place, discarding the invalid bid. From winner declaration there is a further move specified by Line6, which occurs when the auctioneer declares the winner of the auction and charges the cost of the purchase to the winning buyer. Finally, if the time specified by Pause1 elapses, there is a move that takes the scene from place bidding to call price. From here, the auctioneer may call a new price, according to Line2, as long as the new offer price is strictly higher than the reservation price. If the new offer price is equal to or less than the reservation price, the scene moves to place end auction from where the auctioneer can withdraw the item (Line3).