Engineering Multi-agent Systems as Electronic Institutions

Carles Sierra, Juan A. Rodríguez-Aguilar, Pablo Noriega,Josep Ll. Arcos
Artificial Intelligence Research Institute, IIIA
Spanish Council for Scientific Research, CSIC
08193 Bellaterra, Barcelona, Spain.
{sierra,jar,pablo,arcos}@iiia.csic.es
Marc Esteva
Graduate School of Library and Information Science
University of Illinois at Urbana-Champaign
501 E. Daniel Street, Champaign, IL 61820
esteva@uiuc.edu

Abstract

As the complexity of real-world applications increases, particularly with the advent of the Internet, there is a need to incorporate organisational abstractions into computing systems that ease their design, development, and maintenance. Electronic institutions are at the heart of this approach. Electronic institutions provide a computational analogue of human organisations in which human and intelligent agents playing different organisational roles interact to accomplish individual and organisational goals. In this paper we introduce an integrated development environment that supports the engineering of a particular type of distributed systems, namely multi-agent systems, as electronic institutions.

keywords: multi-agent systems, electronic institutions, software engineering, auctions

1 Introduction

As the complexity of real-world applications increases, particularly with the advent of the Internet, there is a need to incorporate organisational abstractions into computing systems that ease their design, development, and maintenance. Electronic institutions are at the heart of this approach. Electronic institutions provide a computational analogue of human organisations in which intelligent agents[8] playing different organisational roles interact to accomplish individual and organisational goals. In this scenario, agent technology helps enterprises reduce their operational costs and speed-up time to market by helping distributed business parties, represented by agents, run smoother and in a better coordinated fashion. Electronic institutions appear as the glue that puts together self-interested business parties, coordinating, regulating, and auditing their collaborations.

But why electronic institutions? Research and development in agent-based systems has traditionally bargained for well-behaved agents immersed in reliable infrastructures in relatively simple domains. Such assumptions are not valid any longer when considering *open systems*[6] whose components are unknown beforehand, can change over time, and can be self-interested human and software agents developed by different parties. Thus, open multi-agent systems[8] (MAS) can be regarded as distributed systems where (possibly) large, varying populations of agents exhibiting different (possibly deviating, or even fraudulent) behaviours interact. Notice that unlike distributed systems, cooperation among these agents is not fixed at design time but may emerge at real time.

Openness and self-interest without some control may lead to unexpected, chaotic behaviours in a MAS. The challenging issue is to avoid dreadful execution dynamics, particularly in critical applications (e.g. open marketplaces, collaborative project management, virtual organisations, supply network management). Therefore, the design and development of open MAS appears as a highly complex task. Hence, it seems apparent the need for introducing regulatory structures establishing what agents are permitted and forbidden to do. Notice that human societies have successfully dealt with regulation by deploying institutions. Thus, we advocate for the introduction of their electronic counterpart, namely *electronic institutions* (EIs)[9], to shape the environment wherein agents interact by introducing sets of artificial constraints that articulate and thus help coordinate their interactions. Our actual experiences in the deployment of actual-world MAS as EIs [12, 2] allows us to defend the validity of this approach. Notice though, that as noted in [4, 11] we believe that engineers need to be supported by well-founded tools. Hence the purpose of this paper is to introduce an integrated development environment for EIs that supports engineers in the principled design and development of MAS. But first things first. Thus, a characterisation of EIs is in place.

2 Electronic Institutions

According to [10], human interactions are guided by *institutions*, which represent the rules of the game in a society, including any (formal or informal) form of constraint that human beings devise to shape human interaction.

Thus, institutions are the framework within which human interaction takes place, defining what individuals are forbidden and permitted and under what conditions. Human organisations and individuals conform to the rules of institutions in order to receive legitimacy and support. Establishing a stable structure to human interactions appears as the *raison d'être* of institutions.

We defend the adoption of a mimetic strategy in order to cope with the complexity of engineering open MAS. And then, if we uphold that open MAS can be effectively designed and implemented as EI, what is an EI? In what follows we identify the core notions on which we found our current conception of electronic institution:

- *Agents and Roles.* Agents are the players in an electronic institution, interacting by the exchange of speech acts[1] or illocutions, whereas roles are defined as standardised patterns of behaviour. Any agent within an electronic institution is required to adopt some role(s). Recently, the concept of role is becoming increasingly considered by researchers in the agents' community [15]. We differentiate between institutional and non-institutional (external) roles as well as institutional and non-institutional (external) agents. Whereas institutional roles are those enacted to achieve and guarantee institutional rules, non-institutional roles are those requested to conform to institutional rules.

- *Dialogical framework.* Some aspects of an institution such as the objects of the world and the language employed for communicating are

[1] In the sense proposed by Searle[13], who postulates that utterances are not simply propositions that are true or false,but attempts on the part of the speaker at achieving some goal or intention.

fixed, constituting the context or framework of interaction amongst agents. EIs establish the acceptable speech acts by defining the ontology[2] and the common language for communication and knowledge representation which are bundled in what we call dialogical framework. By sharing a dialogical 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 specification of the possible dialogues agents may have. Notice however that the communication protocol defining the possible interactions within a scene is role-based instead of agent-based. In other words, a scene defines a role-based framework of interaction for agents.

- *Performative structure.* Scenes can be connected, composing a workflow, the so-called performative structure. The specification of a peformative structure contains a description of how agents can legally move from scene to scene by defining both the pre-conditions to join in and leave scenes. Satisfying such conditions will fundamentally depend on the roles allowed to be played by each agent and his acquired commitments[3] through former utterances.

- *Normative Rules.* Agent actions in the context of an institution have

[2]Here we adhere to the definition in [1]: *An ontology for a body of knowledge concerning a particular task or domain describes a taxonomy of concepts for that task or domain that define the semantic interpretation of the knowledge.*

[3]We understand commitments as obligations to do somehting or to bring about some state of affairs.

consequences, usually in the shape of compromises which impose obligations or restrictions on dialogic actions of agents in the scenes wherein they are acting or will be acting in the future. The purpose of normative rules is to affect the behaviour of agents by imposing obligations or prohibitions. Notice that institutional agents are committed to undertake the required actions so that to ensure that non-institutional agents abide by institutional rules.

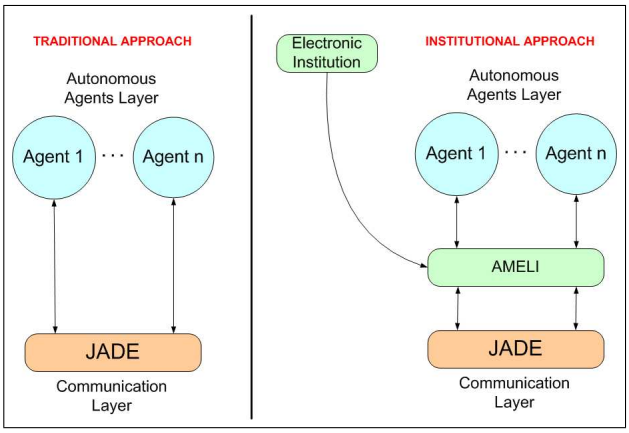

Figure 1: Agent mediation via electronic institutions

The notions above picture the regulatory structure of an EI as a "workflow" (performative structure) of multi-agent protocols (scenes) along with a collection of (normative) rules that can be triggered off by agents' actions (speech acts). At run time, agents enter and leave an EI, joining and leaving scenes, which are dynamically created and destroyed, moving among scenes, and acting by uttering speech acts. The main goal of an EI is to enforce the specified norms to participating agents at run time. At this aim, all participating agents have their interactions mediated by the EI as shown in

figure 1. Unlike traditional approaches that allow agents to openly interact with their peers via a communication layer, our computational realisation of an EI must be regarded as a *social* middleware that sits between the external, participating agents, and the chosen communication layer validating (filtering in) or rejecting (filtering out) their actions[4].

3 An IDE for Electronic Institutions

IDE-eli, the Integrated Development Environment for Electronic Institutions, is a set of tools aimed at supporting the engineering of MAS as electronic institutions. Software agents appear as the key enabler technology behind the electronic institutions vision. Thus, electronic institutions encapsulate the coordination mechanisms that mediate the interactions among software agents representing different parties as depicted in figure 1. IDE-eli allows for engineering both electronic institutions and their participating software agents. Notably, IDE-eli moves away from machine-oriented views of programming toward organisational-inspired concepts that more closely reflect the way in which we may understand distributed applications such as MAS. It supports a top-down engineering approach: firstly the organisation, secondly the individuals. IDE-eli is composed of:

ISLANDER A graphical tool that supports the specification of the rules and protocols in an electronic institution.

AMELI Software platform to run electronic institutions. Electronic institutions specified with *ISLANDER* are run by *AMELI*.

[4]In figure 1, Jade[7] is employed as the communication layer.

aBUILDER Agent development tool.

SIMDEI Simulation tool to animate and analyse *ISLANDER* specifications prior to the deployment stage.

Figure 2 depicts the role of the IDE-eli tools in an electronic institution's development cycle. Notice that such cycle is regarded as an iterative, refining process fully supported by the IDE-eli tools. In what follows we detail the different steps of such development cycle along with the roles played by the IDE-eli tools.

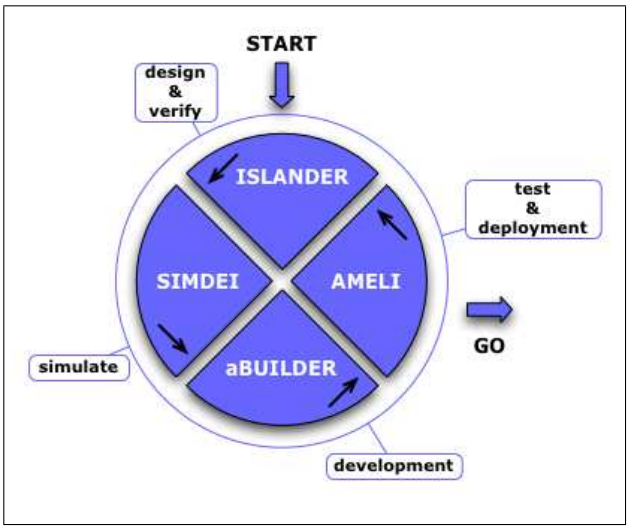

Figure 2: Electronic Institution Development Cycle

- *Design.* Electronic institutions can be graphically specified with the aid of *ISLANDER* [3][5]. It allows for the definition of a common ontology, all the interactions that agents may have, and the consequences

[5]Best prototype paper award at the First Joint International Conference on Autonomous Agents and Multiagent Systems AAMAS 2002.

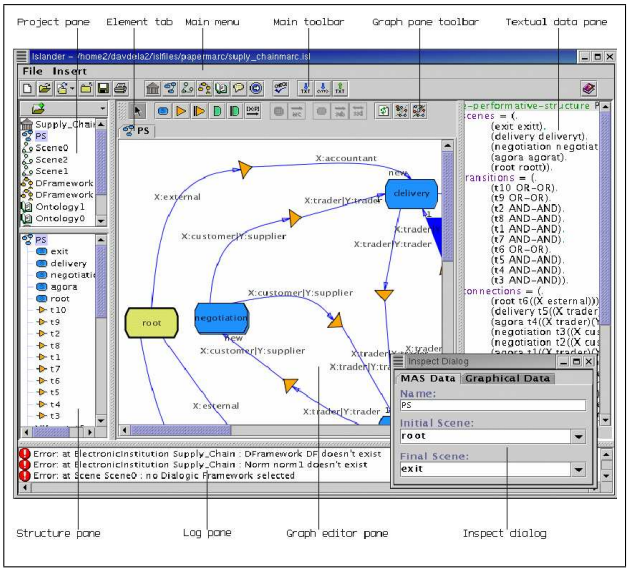

Figure 3: Islander GUI

of such interactions. Figure 3 shows the graphical user interface of *ISLANDER*. The result is a precise description of the kinds and order of messages that the agents in the MAS can exchange, along with the collection of norms that regulate their actions. *ISLANDER* eases the job of EI designers by supporting graphical specifications based on the formalisation of EIs offered in [4]. Notice that the specification of an EI focuses on macro-level (societal) aspects, instead of on micro-level (internal) aspects of agents.

- *Verification.* Once specified an institution, it should be verified before opening it to external, participating agents. This step is twofold. While the first verification stage focuses on *static*, structural properties of a specification, the second stage is concerned with the expected,

dynamic behaviour of the EI.

The *static verification* of EIs amounts to checking the structural correctness of specifications. For instance, to check that protocols are correctly specified. This process is fully supported by the *ISLANDER* editor.

On the other hand, the dynamic verification of EIs is carried out via simulation. The process starts out with the definition of populations of agents of varying features capable of acting in the specified institution. In order to facilitate this task we have developed an *agent builder* capable of generating, from a specification, an agent skeleton depending on the roles and interactions in which the agent may participate. In order to completely define an agent, the generated skeleton must be filled out by agent designers with decision making mechanisms. Once agents have been implemented, simulations of the EI can be run using the *SIMDEI* simulation tool developed over REPAST [14]. *SIMDEI* supports simulations of EIs with varying populations of agents to conduct *what-if analysis*. The institution designer is in charge of analysing the results of the simulations and return to the design stage if they differ from the expected ones.

- *Development.* Once the institution specification is validated, it can be deployed and open for agent participation. Thus, it is time for agent programmers to implement their participating agents. Notice that we do not impose restrictions on the type of agents that can participate in an EI. Agent designers can choose their own language and architecture.

Nonetheless, we believe that it is important to support this intricate development process via the *aBUILDER* tool. *aBUILDER* facilitates the development of agents in JAVA in a pre-defined architecture. In the future it is planned to support further languages and architectures.

- *Deployment.* An electronic institution defines a normative environment that shapes agents' interactions. Since agents may be heterogeneous and self-interested we can not expect that they behave according to the institutional rules. Therefore, any EI is executed via *AMELI* [5], an infrastructure that mediates and facilitates agents' interactions while enforcing the institutional rules. The implemented infrastructure is of general purpose, as it can interpret any *ISLANDER* specification. Therefore, it must be regarded as *domain independent*, and it can be used in the deployment of any specified institution without any extra coding. *AMELI* keeps the execution state and uses it, along with the EI specification, to validate the actions that agents attempt. Hence, the execution of an EI starts out by running *AMELI* after loading the specification. Thereafter, external agents may enter the institution to interact with other agents through *AMELI*. An EI execution can be monitored thanks to monitoring tool that depicts all the events occurring at run time, as shown in figure 4.

In the next section we sketch an actual-world EI whose development has been supported by IDE-eli.

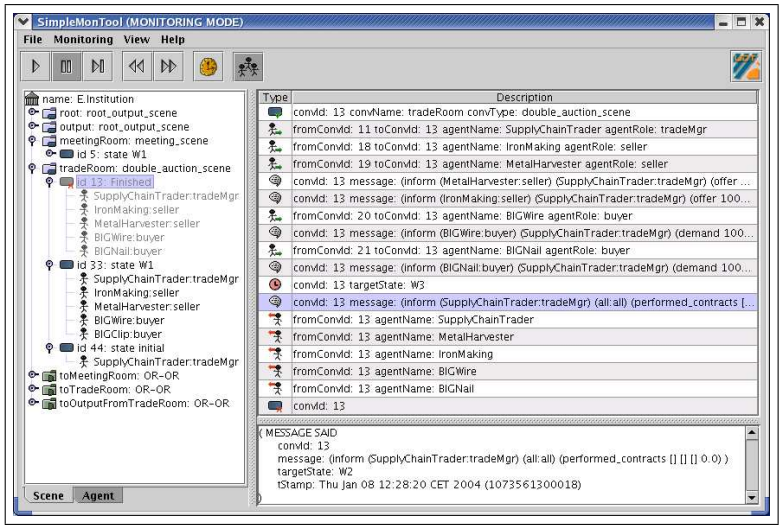

Figure 4: Electronic institution monitoring

4 Case study

In the Mediterranean fresh fish has been traditionally sold through downward bidding auctions operating in auction houses close to harbours. Fish is grouped into sets of boxes, called *lots*, and put at auction following the Dutch protocol: price is progressively and quickly lowered -4 quotes per second- until a buyer submits a bid or the price descent reaches the withdraw price. The buyer submitting the bid can decide to buy the complete lot or just some boxes. In the later case, the remaining boxes are put back at auction the next round. When the last box is sold, the auction is over.

Some fishmarkets are adapting their selling methods to new technologies and most auctions are nowadays automated by some *auction system*. Nonetheless, the presence of human buyers at the auction houses is still necessary. This imposes two main barriers. First, it restricts the potential

buyers to those present in the auction house. Second, it makes the participation in several auctions simultaneously costly, as companies have to send a representative to each one. The elimination of such limitations would be very profitable for both buyers and sellers. Increasing the number of buyers makes the market more competitive, and thus increases the buying price to the benefit of sellers. It also permits the participation of buyers without intermediaries saving costs to the buyers.

Agent technologies may be used to eliminate these limitations. The Multi-Agent System for FIsh Trading (MASFIT)[2] [6] allows buyers to remotely and simultaneously participate in several wholesale fish auctions with the help of software agents, while maintaining the traditional auctions.The participation of buyer agents in auctions is mediated by an EI. The MASFIT's institution controls buyers' access to the auctions, provides them with information, and collects their bids during the auctions. At this aim, the auction system running at the auction houses has been connected to the developed institution. MASFIT interconnects multiple auction houses, and therefore it gives structure to a federation of auction houses. Importantly, the MASFIT system guarantees that the buyer agents have access to the same information, and have the same bidding opportunities as human buyers physically present at the auction house. Furthermore, the system does not alter the current operation of the auction houses.

As reported in [2], the IDE-eli tools played a key role in the design and development of the MASFIT system. On the one hand, the MASFIT electronic institution was specified using *ISLANDER*. On the other hand,

[6] http://www.masfit.net.

agents in the institution have their interactions mediated by *AMELI*. It facilitates agent participation and communication, within the institution, while enforcing the institutional rules encoded in the specification. At this aim, it loads institution specifications as generated by *ISLANDER*. *AMELI* permits, among many others features, the distributed execution of agents. This is important as agents must be running at different places.

Finally, notice that the market scenario created by MASFIT makes the participation on simultaneous auctions a complex decision-making task. Agents have to manage huge amounts of information —even uncertain information— and their reasoning and processing time must be short enough to react to changes. To support this complex design, MASFIT also includes tools to create, customise, manage and train software buying agents.

5 Summary

In this paper we have introduced an integrated development environment for the engineering of multiagent systems as electronic institutions. Major benefits derive from employing the IDE-eli tools. Firstly, they help shorten the development cycle. The engineering of an electronic institution requires a low-cost implementation since only its participating agents must be programmed. The inherent flexibility of *ISLANDER* in the design of coordination mechanisms favours an easy, ready maintenance: once changes are accommodated in a new specification, this is ready to be run by *AMELI*, and agents are ready to plug and play. Secondly, the IDE-eli simulation tools support what-if analysis of electronic institutions' designs

prior to their deployment, facilitating the location of unexpected behaviours that may jeopardise critical applications. Furthermore, the development of both external (non-institutional) and internal (institutional) agents for the specified institutions is supported by *aBUILDER*.

IDE-eli has proven to be highly valuable in the development of e-commerce applications such as the MASFIT system presented in section 4. However, a wider range of application areas may be tackled with the aid of the IDE-eli tools. In general terms, the electronic institutions approach is deemed as appropriate in complex domains where multiple partners are involved, and a high degree of coordination and collaboration is required. Thus, electronic institutions to support workflow management, the monitoring and management of shop-floor automation, or supply network management issues look promising in the near future.

For more information and software downloads, the interested reader should refer to http://e-institutions.iiia.csic.es.

Acknowledgements

Marc Esteva enjoys the Fulbright/MECD postdoctoral scholarship FU2003-0569. This paper has been partially supported by project Web-i(2) (TIC-2003-08763-C02-01). The authors would like to thank the IIIA Technological Development Unit's programmers for their valuable contribution to the development of the IDE-eli.

References

[1] L. K. Alberts. *YMIR: An Ontology for Engineering Design*. PhD thesis, University of Twente, 1993.

[2] Guifré Cuní, Marc Esteva, Pere Garcia, Eloi Puertas, Carles Sierra, and Teresa Solchaga. Masfit: Multi-agent systems for fish trading. In *16th European Conference on Artificial Intelligence (ECAI 2004)*, Valencia, Spain, August 2004.

[3] M. Esteva, D. de la Cruz, and C. Sierra. Islander: an electronic institutions editor. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-agent Systems (AAMAS 2002)*, pages 1045–1052, Bologna, Italy, July 15-19 2002.

[4] Marc Esteva. *Electronic Institutions: from specification to development*. IIIA PhD Monography. Vol. 19, 2003.

[5] Marc Esteva, Juan A. Rodríguez-Aguilar, Bruno Rosell, and Josep L. Arcos. Ameli: An agent-based middleware for electronic institutions. In *Third International Joint Conference on Autonomous Agents and Multi-agent Systems (AAMAS'04)*, New York, USA, July 19-23 2004.

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

[7] Java Agent Development Framework. http://sharon.cselt.it/projects/jade.

[8] Nicholas R. Jennings, Katia Sycara, and Michael Wooldridge. A roadmap of agent research and development. *Autonomous Agents and Multi-agent Systems*, 1:275–306, 1998.

[9] Pablo Noriega. *Agent-Mediated Auctions: The Fishmarket Metaphor.* IIIA Phd Monography. Vol. 8, 1997.

[10] D. North. *Institutions, Institutional Change and Economics Perfomance.* Cambridge U. P., 1990.

[11] Juan A. Rodríguez-Aguilar. *On the Design and Construction of Agent-mediated Electronic Institutions.* IIIA Phd Monography. Vol. 14, 2001.

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

[14] REPAST URL. http://repast.sourceforge.net.

[15] F. Zambonelli, N. Jennings, and M. Wooldridge. Developing multiagent systems: The gaia methodology. *ACM Transactions on Software Engineering and Methodology*, 12(3):317–370, 2003.