Programming Electronic Institutions with Utopia

Pierre Schmitt, Cédric Bonhomme, Jocelyn Aubert, and Benjamin Gâteau

Centre de Recherche Public Henri Tudor
Service Science and Innovation Dpt.
Luxembourg, G.D. of LUXEMBOURG
{firstname.lastname}@tudor.lu
http://www.tudor.lu

Abstract. In Multi-Agent Systems, Organizations are means to structure cooperation and collaboration between agents. MoiseInst is a normative Organization model giving the possibility to constraint agents behaviour according to four dimensions (structural, functional, contextual and normative). Mabeli as Electronic Institution model allows the supervision of MoiseInst Organizations compliance through an arbitration system. The difficulty is to easily instantiate such Organizations to obtain a dynamic entity in which agents can evolve. In this paper we introduce Utopia, our Institution-oriented and Institution-based programming framework. Utopia permits to easily and automatically set up a MAS thanks to a XML MoiseInst Specification file. The framework convert this file into an innovative mathematical structure namely a recursive graph, and solve several optimization problems in order to compute the most efficient role distribution. We show a concrete application of the prototype through RED, an EUREKA/CELTIC European project use-case.

1 Introduction

In human societies, Institutions define rules [1] that enclose all kinds of formal or informal constraints used by human beings to interact. In Multi-Agent System domain, Electronic Institutions have been introduced to model rules with normative systems [2]. That is why we define Electronic Institutions as a set of agents which behave according to Norms and by taking into account their possible violation (and sanction).

These last years Electronic Institution platforms have been improved thanks to new services making them able to express cooperation schemes defined by the user with an Organization Modelling Language such as for instance $\mathcal{M}$OISE$^+$ [3], ISLANDER [4], OMNI [5]. The aim of these services is to constraint and supervise agent's actions and interactions in order for them to achieve some global Goals. We call those explicit cooperation schemes *Orgazination Specification* (OS).

The model used to specify the organization of an Electronic Institution is $\mathcal{M}$OISE^{Inst} [6]. In this context, the functioning of the agents is supervised and

controled with a set of *Institution services* regrouped in a specific "normative middleware" called $\mathcal{S}$YNAI on which the agents execute themselves.

This paper aims at presenting how it is possible to easily implement an Electronic Institution specified with $\mathcal{M}$OISE^{Inst}, supervised with $\mathcal{S}$YNAI and in which standard agents provided with the platform evolve and achieve their Goals. For that, three steps have been needed:

1. Define the structure of data in which the OS will be stored.
2. Develop a set of agents working in and able to supervise an Organisation Entity (OE) instantiating the OS defined by an user.
3. Develop a template of JADE based agents able to evolve in the OE (i.e. able to play Roles and achieve Goals) by loading specific behaviours provided by the user in order to execute actions achieving the Goals defined in the OS.

The paper is built as follows: in Section 2 we present rapidly $\mathcal{M}$OISE^{Inst} and $\mathcal{S}$YNAI composing the foundations of our work. Section 3 deals with the implementation of the framework (named Utopia) allowing the implementation of such Electronic Institution. At last, before conclude, the Section 4 illustrates the use of Utopia through an application of security policies deployment developped in the context of European RED project.

2 Normative Organization Modelling

$\mathcal{M}$OISE^{Inst} [6] is founded on the $\mathcal{M}$OISE^{+} organizational model [3]. It is composed of the following components that are used to specify an Organisation of agents in terms of structure, functioning, evolution and Norms (see Figure 1):

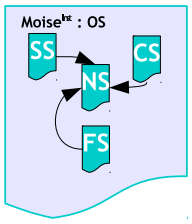

Fig. 1. $\mathcal{M}$OISE^{Inst}, a normative Organization Specification model

- A *Structural Specification* (SS) defines: (i) the Roles that agents will play in the Organization, (ii) the *relations* between these Roles in terms of authority, communication or accointance, (iii) the *Groups*, additional structural primitives used to define and organize sets of Roles;
- A *Functional Specification* (FS) defines global *business processes* that can be executed by the different agents participating to the Organization according to their Roles and Groups;

– A *Contextual Specification* (CS) specifies, *a priori*, the possible evolution of the Organization in terms of a *state/transition graph*;
– A *Normative Specification* (NS) defines the deontic relations gluing the three independant Specification (SS, FS, CS). This NS clearly states rights and duties of each Roles/Groups of SS on sets of Goals (Missions) of FS, within specific states of CS.

These four Specifications form the Organizational Specification (OS). The Organizational Entity (OE) is then built by instantiating the OS through the Agent playing roles, achieving goals and respecting active norms in valid contexts. The $\mathcal{S}$YNAI [7] middleware manages and controls the functioning of this OE . As depicted on Figure 2, $\mathcal{S}$YNAI is composed by a set of manager agents supervising the actions of agents "Agt" on the OE.

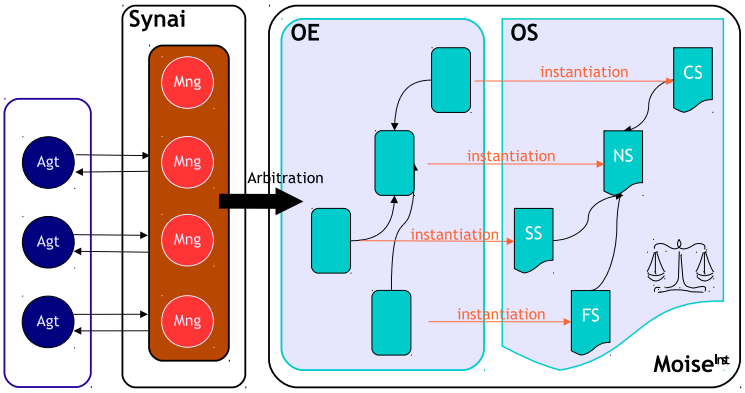

Fig. 2. Supervision by $\mathcal{S}$YNAI of an OE

This layer is in charge of: (i) managing the life cycle of SS as entering/exiting of agents within the Organization, or requesting/leaving of Roles or Groups by the agents, (ii) coordination of the concurrent execution of FS as commitment to Missions or achievement of Goals, etc, (iii) dynamic and evolution of the Organization state through the CS, (iv) the monitoring and supervision of Norms of NS activated/deactivated by the evolution of the Organization.

While agents evolve inside the organization, agents of $\mathcal{S}$YNAI have to interpret and "understand" the OS (in order to respect it or to control it). For that, we need to structure the data of the organization and to this end, we chose recursive graph.

3 Implementation of Utopia

Recursive graphs are innovative mathematical structures [8] widely used to have a very generic representation of data. In our case, a recursive graph particularly

meets the underlying needs of $\mathcal{M}$OISE^{Inst} which is mostly recursive : Groups can include others Groups, Missions can include others Missions, etc... Moreover, the sub recursive graph extraction makes the data sharing more easier.

Utopia and its architecture using an Electronic Institution paradigm make the essential problematics of setting up a Multi-Agent System easier. Indeed two steps are needed:

1. Define the OS in a XML file (an authoring tool to specify the OS will be developped later).
2. Develop specific behaviours (in java classes) that the generic agents will load in order to execute actions achieving the goals defined in the OS.

4 Demonstration scenario

Our use-case is part of a demonstrator set up in the context of the RED project [9] which defines and designs solutions to enhance the detection/reaction process, improves the overall resilience of IP networks to attacks by embedding means to enrich the alert with better characterized information, and additional information about the origin and the impact of the security incident.

To provide the detection and reaction functionalities, RED proposes an architecture containing a set of elements, depicted in Figure 3:

- ACE (Alert Correlation Engine): this element is in charge to receive alerts from network nodes, and enhances the detection of attacks by combining several diagnosis combinations.
- PIE (Policy Instantiation Engine): this element receives the information about attacks from the ACE and instantiates new security policies to react to the attack in a high level reaction loop. This paper is focused on this element.
- PDP (Policy Decision Point): this element receives the new security policies defined by the PIE and deploies them in the enforcement points.
- RDP (Reaction Decision Point): this element receives the information about attacks from the ACE and decides of how to act in a mid level reaction loop.
- PEP/REP (Policy Enforcement Point/Reaction Enforcement Point): This component, outside the RED node, enforces the security policies provided by the PDP and the reaction provided by the RDP. It also performs an immediate low level reaction.

RED proposes three different types of reaction based on level of diagnosis required to apply them:

- Immediate reaction, which is an automatic response with a diagnosis based on the capabilities embedded in the device and decided by the PEP/REP,
- Short term reaction, where the diagnosis is done with a limited and local vision of the monitored information system, decided by the RDP based on the information provided by the ACE and which does not instantiate new security policies,

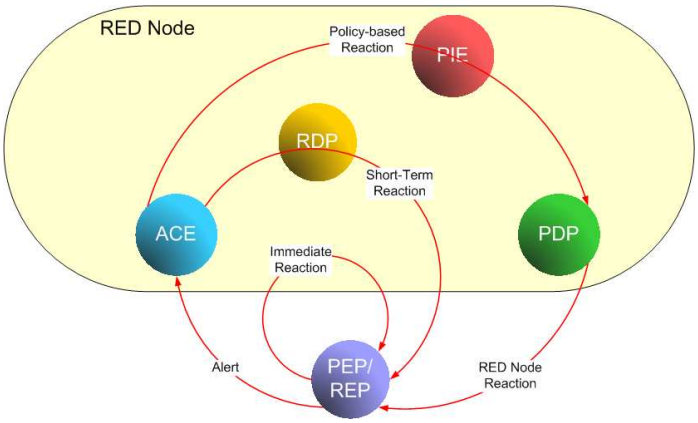

Fig. 3. RED architecture

– Long term reaction, where the diagnosis is done with a global vision of the monitored information system, decided by the PIE and which generate new security policies based on the ACE alerts which are sent to the PDP to deploy them in PEP.

A multi-agent system is used to represent RED nodes. Each component is represented by an agent playing a Role (ACE, RDP, PIE, PDP, REP, PEP) of the node which is represented as a $\mathcal{M}$OISE^{Inst} Organization. In the following, we will describe the Goals that agents have to achieve in a context of a black-hole attack.

4.1 Black-hole attack and countermeasures

In our scenario Alice and Bob are communicating with help of a VoIP service provided by a SIP server. A Malicious node executes an attack structured in two successive steps. First, the Malicious node changes the ARP tables of Alice, Bob and the SIP Server (ARP poisoning) in order to have all the trafic routed by itself. Then, it carries out a black-hole attack by dropping (not retransmitting) the packets. As a result, the conversation between Alice and Bob cannot progress.

Once the attack succeeded, an intrusions detection tool detects the attack and sends alerts to the PIE and the RDP through the ACE. The agent playing the Role of RDP have to apply a short term reaction by asking PEP to delete their ARP entries corresponding to the MAC address of the malicious node. The agent playing the Role of PIE aims at implementing new policies forbidding the input and the forward of trafic coming from the malicious node (via its MAC address) and adding static ARP entries binding the real IP addresses and MAC addresses. Then the PIE agent sends these new policies to PDP which transform them into script and/or executable command regarding to PEP's specifications (type, host, OS, etc.). At last, agents playing PEP Role have to execute command

and/or scripts on the device they interface. We will see more precisely in the next section how an Organization is implemented with Utopia in order to represent a RED node as an Electronic Institution.

4.2 Implementation with Utopia

Utopia make possible to easily deploy a MAS where agents play the appropriate Roles, namely ACE, PIE, PDP and PEP from a simple Structural Specification. Thanks to cardinalities, the MAS composition can respect the RED architecture : ACE, PIE and PDP are played by only one agent and PEP are distributed over the network devices.

We can handle the agent behaviour after an attack with a simple Functional Specification : four Missions (one for each agent) composed by two Goals run in parallel, one dedicated to messages reception, the other to message sending. The following shows Domain Knowledge Specification of the goals binding them to their corresponding java classes that the user have to provide, and the FS coming from the OS XML file. There is no grouping of goals in missions, that's why the FS is so simple.

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="xml/os.xsl" type="text/xsl" ?>
<!DOCTYPE OrganizationalSpecification SYSTEM "../xml/os.dtd">
<OrganizationalSpecification id="Red">
    <DomainKnowledgeSpecification>
      <Goal id="gPIESend" class="red.pie.GPIESend"></Goal>
      <Goal id="gACESend" class="red.ace.GACESend"></Goal>
      <Goal id="gPDPListen" class="red.pdp.GPDPListen"></Goal>
      <Goal id="gPEPListen" class="red.pep.GPEPListen"></Goal>
      <Goal id="gPEPIPListen" class="red.pep.GPEPIPListen"></Goal>
[...]
    </DomainKnowledgeSpecification>
[...]
    <FunctionalSpecification>
      <GoalId>gPIESend</GoalId>
      <GoalId>gACESend</GoalId>
      <GoalId>gPDPListen</GoalId>
      <GoalId>gPEPListen</GoalId>
      <GoalId>gPEPIPListen</GoalId>
    </FunctionalSpecification>
</OrganizationalSpecification>
```

The Normative Specification only force the four agents playing the Roles of ACE, PIE, PDP and PEP to do their associated Missions, that is to say, to run two Java Goal implementations. Obviously, each Goal implementation allow the specialization of the agents, and thanks to Utopia's primitive functions, it is very easy to send or receive messages and XML alerts.

5 Conclusion

In this paper we described an Electronic Institution programming framework named Utopia based on $\mathcal{M}$OISE^{Inst} for the Organization Specification and on recursive graph for the Organization representation. Thanks to a recursive graph,

all the homogeneous data are stored in an unique recursive structure, allowing us to easily distribute the shared information between agents of Utopia using concepts such as sub-recursive graphs.

With the RED use-case we showed how easily the essential problematics of setting up a Multi-Agent System could be solved with Utopia and its powerful architecture using an Electronic Institution paradigm. Actually Utopia allows to simply deploy a MAS without any need of network programming (as Socket coding or thread management). Furthermore, with this kind of network abstraction, the implementation of RED is completely reusable: we can run the system on many different networks. Moreover, it is far easier to brings into the MAS development many security specialists, as Electronic Institution permits to clearly separate the different system Goals and thus, the different security problematics.

Despite the easiness of implementing a working Electronic Institution that Utopia brings, as demonstrated in a real use-case, some improvements can be considered. Actually, the way of managers and supervisor to control the functionning of the organization is basically a centralized arbitration system. However the multi-agent system principles advocate decentralization. As a consequence, a first evolution could be done in order to obtain an Electronic Institution allowing the distribution of the OE and $\mathcal{S}$YNAI without putting the optimization of the role distribution aside. Moreover, the agents' decision taking mechanisms could be improved to exhibit a smarter behaviour in order to choose the right Goals to achieve at the right time more efficiently.

Acknowledgment This work has been funded by Luxembourger FNR-CORE project TITAN (C08/IS/21).

References

1. North, D.C.: Institutions, Institutional Change and Economic Performance. 1st edition edn. Political Economy of Institutions and Decisions. Cambridge University Press (October 26 1990)
2. Jones, A., Carmo, J.: Deontic logic and contrary-to-duties. In: Handbook of Philosophical Logic. Kluwer (2001) 203–279
3. Hübner, J.F., Sichman, J.S., Boissier, O.: A model for the structural, functional, and deontic specification of organizations in multiagent systems. In: SBIA'02. Number 2507 in LNAI, Springer (2002) 118–128
4. Esteva, M., Rosell, B., Rodriguez-Aguilar, J.A., Arcos, J.L.: Ameli: An agent-based middleware for electronic institutions. In: AAMAS'2004, New York City, USA, ACM Press (19-23 July 2004) 236–243
5. Dignum, V., Vazquez-Salceda, J., Dignum, F.: Omni: Introducing social structure, norms and ontologies into agent organizations. In: ProMAS International Workshop 2004, New York, USA (2004)
6. Gâteau, B., Boissier, O., Khadraoui, D., Dubois, E.: Controlling an interactive game with a multi-agent based normative organizational model. In: COIN II. Volume 4386/2007 of LNCS., Springer (2007) 86–100
7. Gâteau, B.: Modélisation et supervision d'institution multi-agent. PhD thesis, ENS Mines Saint-Etienne (2007)

8. Harel, D.: Towards a theory of recursive structures. In Springer, ed.: 23rd International Symposium on Mathematical Foundations of Computer Science. Volume 1450 of LNCS. (1998) 36–53
9. Feltus, C., Khadraoui, D., de Remont, B., Rifaut, A.: Business governance based policy regulation for security incident response. In: Crisis'07, Marrakech, Morocco (2-5 July 2007)