MANET: A Model for First-Class Electronic Institutions

Charalampos Tampitsikas[1,2], Stefano Bromuri[2], and Michael Ignaz Schumacher[2]

[1] Università della Svizzera italiana,
Communication Sciences Department,
via G. Buffi 13, 6900 Lugano, Switzerland,
charalampos.tampitsikas@usi.ch
[2] University of Applied Sciences Western Switzerland,
Institute of Business Information Systems, 3960 Sierre, Switzerland
{stefano.bromuri, michael.schumacher}@hevs.ch

Abstract. In this paper we present a new approach to model electronic institutions that are situated in agent environments where heterogeneous agents reside. An electronic institution is seen here as an entity that is deployed within an environment infrastructure that directly mediates the agents interaction. The environment allows the rules of the institution, in terms of powers, permissions and obligations to be perceivable as first class entities by the agents belonging to the institution. We express institutions as first class abstractions that can be inspected, manipulated and modified, created and destroyed by the agents populating the agent environment where the institution resides. To represent the institution we utilize the Object Event Calculus (OEC) formalism that deals with the evolution of complex structures in time and we extend it to deal with the mediation of the events and with the perception of complex structures and events within institutions. We use an e-Health marketplace based on dutch auctions to illustrate the properties of our model.

Keywords: normative systems, multi-agent systems, electronic institutions, logic programming, agent environments

1 Introduction

In the Web 3.0 [14], human beings and software applications freely interact to carry out complex activities, inclusive of (but not limited to) e-business and e-government applications. People and organizations delegate many of their tasks to software applications, called agents. An agent is considered an autonomous entity which observes and acts upon an environment and directs its activity towards achieving its goals [25]. These agents behave as representatives acting both reactively and proactively in their principal's interest while they are also empowered to carry out tasks that have legal effects, like signing contracts and performing business transactions.

Many Web 3.0 applications can be defined as complex *open multi-agent systems* (MASs) [13]. A MAS can be considered open [18] when it satisfies the following properties: i) agents are free to join and leave at any time and ii) agents are designed by and represent different stakeholders with different objectives.

Due to the properties of open MASs, a set of issues must be considered [4]: an open MAS is by definition dynamic as the agents may join or leave at any time; it is insecure as an agent may be programmed to be malicious; it is not deterministic as no agent can have a global knowledge of the system; and finally it has not a central authority. Normative systems, or as Ägotnes et al. specify in [1], systems were social rules apply, try to tackle these issues by defining rules to coordinate heterogeneous agents.

Electronic institutions (EI) [12] are an approach to normative systems, containing a constitutive and regulative part [5]. We can regard an EI as a means for imposing a well-defined structure to the social reality within which agents interact [20], based on a set of rules that mediate the interaction taking place between the agents.

However, EIs suffer from a number of drawbacks. First of all, despite the fact that normative systems provide a level of abstraction in terms of social rules amongst agent societies, it is not clear how these rules mediate the interaction in a MAS in terms of concrete mechanisms. Moreover, the governor agent approach suggested by some enforcement based normative systems [10] has the disadvantage of mixing the concept of infrastructure with the concept of agent, implying that everything in the system is represented as a communicating agent, even when encapsulating low level reactive resources, resulting in quite computationally expensive applications. An example of a model where everything inside the MAS is considered as a communicating agent is the framework proposed by Campos et al. [7] as an extension to electronic institutions.

Secondly, EIs lack of mechanisms that allow the perception of institutional entities and events. The perception of an EI could allow the agents to decide whether participating in the institution would benefit the accomplishment of their goals. Thirdly, current research on normative systems is mainly focused on the communication events inside the system. While communication events are of great importance, they are not sufficient to describe all the possible interactions of the institutional entities and they cannot fully describe the evolution of the system.

To avoid the three drawbacks described above, we present a meta-model which considers the notion of EI as the social constitutive element of an agent environment. In particular, our contribution is to provide a model that proposes the following solutions to the current drawbacks of EIs: i) we introduce the concept of institutional space as the mediator of the social interaction between agents in the agent environment; ii) we propose a perception model to observe institutional spaces and their norms; iii) we present an event system to handle the evolution of institutional spaces and of MASs related to these institutional spaces. We illustrate the perception properties of these concepts by means of an e-Health marketplace example. Although the perception of norms implies that agents can interpret them, for the purposes of this paper we focus only on the social interactions of the agents.

The remainder of this paper is structured as follows: Section 2 is a description of the main properties of agent environment that we have to take into consideration to define our EI meta-model; Section 3 presents our meta-model of first-class Electronic Institutions; Section 4 shows how we apply our model within an e-Health market place based on Dutch auctions; Section 5 shows sketches of our implementation in Prolog; Section 6 puts our work in comparison with existing EI frameworks; finally Section 7 concludes this paper and shows some possible future work directions.

2 Normative Systems and First-Class Agent Environments

Although there is not a clear definition of the agent environment in the traditional normative system models, Esteva in [9] was the first to mention that EIs can be considered as an effort to shape the environment where the agents are situated, offering the agents the conditions to exist and interact. Usually, in the literature, the environment is considered as a domain-specific infrastructure for agents while its main responsibility is the objective coordination of the agents [24].

The agent environment can be used as a first-class abstraction that mediates the interaction between agents taking part in a distributed MAS. First-class abstraction means that the environment is an independent component inside the MAS structure that has its own responsibilities irrelevant to the goals of the agents. According to Weyns [24], the agent environment as a first-class entity can offer four different levels of support:

Basic level: at this level the environment enables agents to access to the *deployment context*. By deployment context, it is meant the external resources with which the MAS interacts (e.g. printers, databases and Web services).

Abstraction level: at this level the environment shields low-level details of external resources defining a standard interface that the agents can access from the environment.

Interaction-mediation level: the interaction mediation level offers support to regulate the access to resources and to mediate the interaction between agents.

Reflection level: The environment supports the modification of its composition and function during runtime. The agents can perceive the properties of the environment and interact in order to modify its state.

A distributed implementation of the model of agent environment proposed by Weyns is represented by the GOLEM agent platform [6]. One of the drawbacks of GOLEM is that it does not model the social interaction amongst the agents, handling it in an ad-hoc manner according to the application, limiting the reusability of the agents and of the infrastructure. Based on the basic level of support proposed by GOLEM, we want to use the abstraction and mediation level of support provided by the agent environment to offer a solution to the main drawbacks of electronic institutions as presented in the previous section and to provide a reusable social interaction model for agent environments. For the purposes of this paper, we study only the environment perception provided by the reflection level and we do not consider run-time modification of its laws.

In order to embed these levels of support into electronic institutions, we first need to specify the appropriate type of normative systems we will use for the mediation of agent interactions. There are two approaches to define normative systems [17]: a) *regimentation based normative systems*, in which a set of rules and protocols are defined to coordinate the behavior of the agent; b) *enforcement based normative systems*, in which some of the agents in the open MAS have the role of regulator agents enforcing the rules when they discover they have been violated.

Regimentation based normative systems are less flexible as it is necessary to specify the rules at design time and the agents are not free to perform actions outside the rules defined by the normative system. The enforcement based approach allows agents to take actions outside the rules of the normative system, but it has the drawback that

sometimes the agents can behave maliciously and not being caught by the enforcer of the law. While proper coordination of agents is crucial, at the same time, it is important to ensure the autonomy of agents. Thus, we envisage electronic institutions as enforcement based normative systems where the agents are free to perform actions outside the rules of the system but for every forbidden action, the system is tracking the violation and applies a corresponding sanction to punish the agent and to preserve its stability. We use enforcement based normative systems that can be created at runtime, but where their rules are first class citizens [21], meaning that the agents can observe the rules.

Moreover, EIs include two basic types of norms: i) constitutive norms and ii) regulative norms. Constitutive norms are based on the notion that "X count-as Y in context C" and are used to support regulative norms by introducing institutional facts in the representation of legal reality [5]. Regulative norms are the main mediation drivers in EIs and are realized by using three main concepts (adapted from [3]) for mediation: power, obligation and permission. Power specifies that an agent can perform a designated action in a context, which creates or changes an institutional fact. Obligation expresses the idea that at a given time the agent should produce an action as specified by the rules of the normative system. Obligation implies also the concept of prohibition or negative obligation as an action that is forbidden by the rules of the system at a certain time. The concept of permission is both related to the state of the EI and to the concept of power. An agent could either exercise its power, if and only if the institutional conditions permit it (conditional power [11]) or exercise its power even if it does not have the permission to do it. On the second case the agent will be sanctioned by the system. Which of the two previous approaches will be followed depends only on the choice of the EI designer.

3 Modeling First-Class Electronic Institutions

3.1 The MANET meta-model

The MANET (Multi-agent Normative EnvironmenTs) meta-model is based on the assumption that the agent environment is composed by two fundamental building blocks; the physical environment, concerned with agent interaction with physical resources and with the MAS infrastructure and the social environment, concerned with the social interactions of the agents and coinciding with the notion of electronic institutions.

In the MANET meta-model we assume that EIs can be composed of three structural components inspired by Stratulat et al. [22]: agents, objects and spaces.

The notion of agent describes the proactive entities within the normative system. For the agents, we assume a separation between a cognitive mind and a physical body with sensors and effectors as described in [6]. The cognitive mind analyzes and reasons about the data received by the sensors as well as reasoning about the agent strategy. The agent uses its effectors to act inside the environment.

The notion of object describes first-class entities that represent virtual entities, virtualizations of external resources or web services, offering an abstraction that hide the low level details from the agents. From the standpoint of EIs, these virtual entities can depict either physical objects either institutional objects.

On one hand physical objects are considered the physical entities of the application (web services, databases, external files etc.) that are present inside an EI. On the other hand, institutional objects, are objects existing only in common agreement amongst the agents of an EI. Institutional objects can be further categorized as: a) objects that can exist within the communication amongst the agents, such as the goods to trade in a market; b) objects that represent agreements between one or more parties; c) objects that represent sanctions for the incorrect behavior of the agent in the EI; d) objects that represent norms of an EI e) objects that represent institutional spaces, f) and finally objects that represent roles of agents within an EI. Moreover, physical objects can be considered as institutional ones when they obtain institutional attributes during the evolution of the agent environment.

Finally the third structural component of our model are spaces. By default in our model, there always exists a root space which contains all the physical laws (derived from the infrastructure of the MAS) of the system (as first-class objects) and where all the other spaces of the system are been created. But in direct analogy to the human reality where we can distinguish between the physical world and the social world, in MASs we can consider institutional spaces [22] describing the EIs in a normative system. All the institutional spaces of a MASs are situated inside the root physical space.

Institutional spaces constitute a first-class representation of the boundaries and the structure of legal entities like EIs. These spaces include the objects and the agents participating in an EI, and contain information about institutions' topology and configuration. The term boundary here implies that spaces specify the limits of the effects of the events performed by the agents. In our model we suppose that the effects of an event produced inside one space hold only for that space. Since spaces are the boundaries and containers of events, they manage norm violations and fulfillments. The content of each event and the combination of role/power of the agents that produced the event are always checked by the space against the corresponding norms. In case of a norm violation, a space will retrieve the information of the appropriate sanction objects and will apply them to the agent that did not comply with the rules of the system. In other words, we see institutional spaces as structures whose state exist in the physical environment, that is perceivable and modifiable through production of events.

It is important to stress that in our model, norms, agreements and sanctions are expressed as complex structures, meaning that they can be deployed as objects in an institutional space. Institutional spaces can be folded inside other spaces or can be distributed across more than one space creating complex topologies. In this paper we show how a space can be created inside another but we do not elaborate the details of possible dependencies between different institutional spaces. We assume that norms of a father space are not propagated to a child space inside it. Each institutional space is discrete and distinct.

In general, in normative systems, agents' interactions can create new institutional realities (e.g. new EIs). In our model, each time a new EI is to be born, a new institutional space is being created, which includes all the norms, the objects and the agents of the institution, which combined together constitute a first-class representation of an EI.

3.2 Modeling First-Class Electronic Institutions with the Object Event Calculus and C-logic

In order to describe the dynamics of our meta-model we use the Object Event Calculus (OEC) formalization. The Object Event Calculus is a dialect of the Event Calculus (EC) [16] that is suitable to represent the evolution in time of complex structures by means of events. The main advantage of OEC is that it determines the state of an object by assigning values to its attribute. Based on this property, it deals with the evolution of an object over time, parameterizing its attributes with times at which these attributes hold various values.

The Object Event Calculus predicates we use for the purposes of this paper are shown below:

(C1) holds_at(Id, Class, Attr, Val, T)← happens(E, Ti), Ti ≤ T, initiates(E, Id, Class, Attr, Val),
not broken(Id, Class, Attr, Val, Ti, T).
(C2) broken(Id, Class, Attr, Val, Ti, Tn)← happens(E, Tj), Ti < Tj ≤Tn, terminates(E, Id, Class, Attr, Val).
(C3) holds_at(Id, Class, Attr, Val, T)← method(Class, Id, Attr, Val, Body), solve_at(Body, T).
(C4) attribute_of(Class, X, Type)← attribute(Class, X, Type).
(C5) attribute_of(Sub, X, Type)← is_a(Sub, Class), attribute_of(Class, X, Type).
(C6) instance_of(Id, Class, T)← happens(E, Ti), Ti ≤ T, assigns(E, Id, Class), not removed(Id, Class, Ti, T).
(C7) removed(Id, Class, Ti, Tn)← happens(E, Tj), Ti < Tj ≤ Tn, destroys(E, Id).
(C8) assigns(E, Id, Class)← is_a(Sub, Class), assigns(E, Id, Sub).
(C9) terminates(E, Id, Class, Attr, _)← attribute_of(Class, Attr, single), initiates(E, Id, Class, Attr, _).
(C10) terminates(E, Id, _, Attr, _)← destroys(E, Id).
(C11) terminates(E, Id, _, Attr, IdVal)← destroys(E, IdVal).

Clauses C1-C2 provide the basic formulation of OEC deriving how the value of an attribute for a complex term holds at a specific time. Clause C3 describes how to represent derived attributes of objects treated as method calls computed by means of a solve_at/2 meta-interpreter as specified in [15]. C4-C5 support a monotonic inheritance of attributes names for a class limited to the subset relation. As C1-C2 describe what holds at a specific time, C6-C7 determine how to derive the instance of a class at a specific time. The effects of an event on a class is given by assignment assertions; the clause C8 states how any new instance of a class becomes a new instance of the super-classes. Finally, deletion of objects is catered for by clauses C9-C11. C9 deletes single valued attributes that have been updated,while C10-C11 delete objects and dangling references.

All the structural entities of our meta-model are considered OEC objects and the relationships between them are depicted in Fig. 1

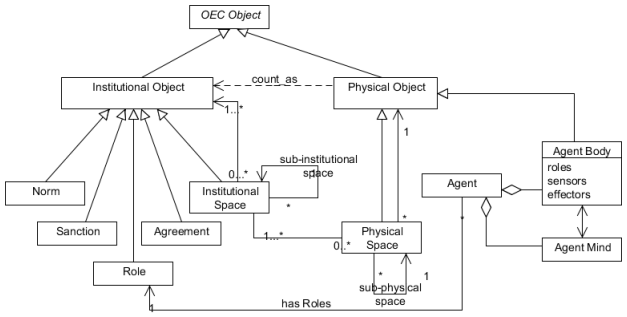

Fig. 1. Entity Categories

To represent the state of the entities at a given time, we will use the C-logic formalism [8] as it is a convenient formalism to represent complex structures and it has a direct translation to the OEC. Complex object descriptions are considered as collections of atomic properties. An object with several attribute labels can be considered as a collection of several atomic formulas. According to the definition of spaces we have introduced, to describe the state of an institutional space at a given time we utilize the following C-logic structure:

```
institutional_space:is1[
    agents ⇒ { agent:a1[ roles ⇒ {role:r1, role:r2} ], agent:a2[ roles ⇒ {role:r1, role:r3} ]},
    institutional_objects ⇒ {norm_object:11, inst_object:o2, inst_object:o3},
    institutional_spaces ⇒ { institutional_space:s2, institutional_space:s3,institutional_space:s4}]
```

that means that is1 is an institutional_space, which has a set of agents a1, a2, a set of institutional objects that the agents can manipulate in the EI and a set of sub institutional spaces. We can translate the C-logic term above to the following first order logic clauses that we can query utilizing the predicates of the OEC:

```
is_a(is1, institutional_space). attribute(institutional_space, agents, multi).
attribute(institutional_space, institutional_objects, multi). attribute(institutional_space, institutional_spaces, multi).
time(e1,1). instance(is1,institutional_space, start(e1)). object(is1,agents, a1,start(e1)).
objects(is1,agents,a2,start(e1)). object(is1,institutional_objects, o1,start(e1)).
objects(is1,institutional_objects,o2,start(e1)). object(is1,institutional_spaces, s3,start(e1)).
objects(is1,institutional_objects,s4,start(e1)).
```

Similarly, the following C-logic structures:

```
power:p1[ mediates ⇒ open_auction:Ev[actor ⇒ IDActor]@T, check_role ⇒ {IDActor, employee}]
sanction:s1[agent ⇒ ag1, credits ⇒ 200]
```

describe respectively a power rule p1, that mediates events of class open_auction, by checking the power of an agent IDActor that enters the EI as an employee to open an auction at time T, and a sanction s1 of 200 credits, applied to agent ag1. We will show later in this paper how such norms and sanctions are applied when the agents execute an action.

3.3 Evolution of Institutional Spaces

To represent our EIs as first-class abstractions, we will need to define how to represent the state of an EI, how to perceive its state and the state of the agents taking part in the interaction, and how to represent the events. In this Section we will illustrate our approach in defining EIs by means of the OEC.

The event schema that we take into consideration in our system is shown in Fig. 2. We distinguish between three kinds of events that are speech events, physical events and sensing events. This distinction is not new and it was already presented in [6], in this paper we further extend the hierarchy of events introducing *institutional events*. Institutional events are considered physical events. This does not go against Searle's definition of social events [20], as, despite the fact that the institutional events modify institutional entitites, they actually change the state of the agent environment acting as regular physical events. Institutional events include the creation/deletion of institutional spaces and objects and are necessary for the construction of every new first-class electronic institution that happens during the evolution of the MAS. Event descriptions are specified as complex terms and are perceivable by any agent inside the institutional space. This property of the events allows the agents to understand every action that happens inside their institutional context. For example, the event description below:

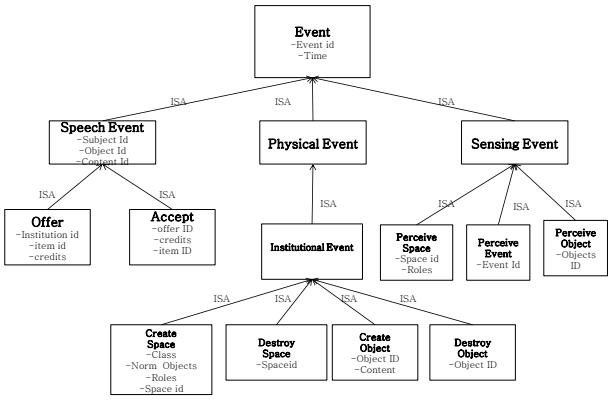

Fig. 2. Events Hierarchy

open_auction:e14[actor ⇒ ag1, auction ⇒ auction:au1[item⇒ medical_item:item1]].

represents an institutional action of agent ag1 who attempts to open an auction about an item item1 of class medical_item. We will see later, how such an action is executed by the agent that causes the event to happen. For the time being, we will assume that the event has happened and we will show how the entities state in the agent environment will evolve as a result of the happening of this event. To do this we need to define domain specific initiates, terminates, assigns and destroys clauses, as shown below:

assigns(E,Obj, auction) ← open_auction:E, auction_of(E,Obj).
initiates(E, Au, auction, item, I) ← open_auction:E [item ⇒ I].

in this way the assigns/3 domain dependent clause above deals with the creation of an auction while the initiates/5 clause assigns an attribute item to the newly created institution. The specification would need also the definition of destroys/3 and terminates/5 clauses to deal with the destruction of an object and termination of an attribute; this is handled in the OEC by the clause C9 shown in Section 3.

3.4 Acting and Perceiving inside Institutions

The representation in terms of C-logic structures of the EIs allows us to have multiple institutions recursively embedded within each others. In order to act within an institutional space, the agents have to be aware of the space where they want to perform an action. For the purposes of this paper, we do not consider dependencies between institutional spaces.

Moreover, the agents' actions are going to be mediated by the regulative rules of the institution as we have already mentioned. As a consequence we say that in order to be performed within an EI, an action has to be attempted in that EI first. We specify how the EI evolves in time by means of assertion of events, where we keep the events description separated from the attempt.

attempt(e14, 120).
do:e14[actor ⇒ ag1, act ⇒ open_auction:m1[institutional_space⇒ IS1]].

In particular, through the rule H1a below we say that in order to happen within the EI the event has to be attempted, the agent producing the event has to have the power to produce the event and the event must be permitted.

H1a)happens(Event, T)← attempt(Event[institutional_space ⇒ IS], T), power(Event,T), permitted(Event, T).
H1b)happens(Event, T)← happens(Event*,T), counts_as(Event*[institutional_space ⇒ IS],Event,T).
H1c)happens(sanction:Event, T)← obligation(Event*@T*,T), T* =T.

As a consequence of defining the happens/2 relation in this way, agents must be aware of the normative systems where they produce events. The rule H1b handles those cases when an event produced outside the normative system, like a physical event in the agent environment, has an effect on a normative system. To achieve this we make use of the counts_as/3 predicate, which states that if an event Event* happens at time T, then also another event Event in relation to an institution identified by IS happens too. The rule H1c specifies that if an obligation has not been satisfied until T, where @T means "at time T", a sanction event happens. We specify further the predicates to enforce the norms of the institution as follows:

H2)obligation(Ev[institutional_space ⇒ IS],T)← instance_of(IS,institutional_space,T), holds_at(IS,norm_object,Oid,T),
instance_of(Oid,obligation,T), apply_norm(Oid,Ev,T).
H3)permission(Ev[institutional_space ⇒ IS],T)← instance_of(IS, space,T), holds_at(IS,institutional_space,norm_object,Oid,T),
instance_of(Oid,permission,T), apply_norm(Oid,Ev,T).
H4)power(Ev[institutional_space ⇒ IS],T)← instance_of(Sid, institutional_space,T),
holds_at(Sid,institutional_space, norm_object, Oid,T), instance_of(Oid,power,T), apply_norm(Oid,Ev,T).

The clauses H2), H3), H4) specify the concepts of power, permission and obligation, that define three distinct kind of norms. The predicate apply_norm/3 is a meta-interpreter that takes the norms in form of objects and check them against the events produced. To express how perception takes place in the EIs, we define the H5) and H6) clauses.

H5)notify(Class:E, Sensor, T) ← happens(E, T), E[institutional_space ⇒ IS], holds_at(IS,agent,Ag,T),
holds_at(IS,owns,Sensor,T), holds_at(Sensor,senses,Class,T).
H6)perceiveE, S, T) ← happens(E, T), perceive_institutional_space(E), E[sensor_of⇒ S, focus⇒ Focus,
institutional_space ⇒ IS].

H5) specifies that whenever an event happens within an institutional space, such event is notified to the agents that are part of such space if they have a sensor that is capable to perceive such events. H6) specifies how an agent can focus to a particular institutional space and perceive its properties, where the solve_at/3 predicate returns a variable substitution of the variables in Focus, if any. The implications of rule H6) is that the agents deployed in the agent environment and taking place in an institutional space can perceive the institutional entities, such as agreements, sanctions and norms, in the institutional space.

4 Applying MANET to an e-Health Marketplace

In this section we illustrate our model with an application from the e-Health domain. We consider an e-Health Marketplace where different hospitals can trade about medicaments, medical equipment and blood. This marketplace can work as a secondary market to the contracts between hospitals and medical providers. The whole marketplace is based on multiple Dutch auctions which can run simultaneously.

To model the e-Health Market place we make use of the general purpose rules presented in Section 3 and we add a set of domain dependent axioms to deal with the evolution of an EI representing a Dutch auction in order to sell medicaments in a e-Health marketplace. In particular, we adapted the Dutch auction as presented in [11] to our formalism based on the OEC and we introduce norms expressed in terms of objects

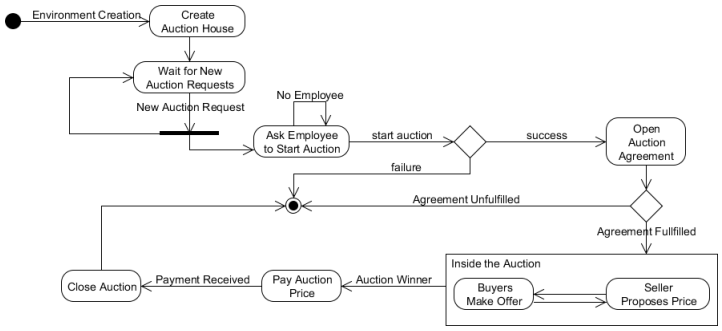

Fig. 3. Auction House Environment State Chart

of the Object Event Calculus formalism. Fig. 3 shows the life-cycle of a Dutch auction within the auction house agent environment.

The Dutch auction electronic institution defines a set of roles for the performance of institutional actions. The roles define the powers that the agents have in the institution. These roles, for the purposes of the e-Health marketplace example, are: a) *Employee*: it is an agent representing the auction house agent environment and that is entitled to open auctions. The agent having this role can also run an auction assuming the role of auctioneer for that auction; b) *Participant*: it is an agent that can express interest for an auction, becoming buyers within the auction; c) *Buyers*: it is an agent that is participating to an auction in the auction house agent environment trying to buy an item of interest; d) *Auctioneer*: it is an agent that coordinates an auction on behalf of a seller agent; e) *Seller*: it is an agent that delegates an auctioneer to sell an item in the Dutch auction. In Fig. 4, once the auction house is created, the environment waits for the opening of an auction. Once a seller contacts an employee agent to open an auction, the employee agent creates an agreement institutional object in the agent environment. This agreement is perceivable by all the agents inside the auction house, which can modify its attributes only with institutional actions. In C-logic terms this agreement object is described as follows:

agreement:c1[
object ⇒ medic_item:b1, debtor:a1[roles ⇒ {role:employee, role:seller}],
creditor:a3[roles ⇒ {role:seller, role:auctioneer}], minimum_price ⇒ 200, deadline ⇒ 2000,
participants ⇒ {agent:aid1, agent:aid2 ... agent:aidn}]

The term above specifies that an agent a1 is going to open an auction before time 2000 for a medical item b1. When an agreement is created, it can be observed from the agents populating the environment, that can express their interest to participate in the auction by modifying the participants attribute of the agreement object with an institutional action. In particular, when the deadline of the agreement expires, we utilize a count_as/3 as follows:

count_as(participate:Ev, assign_role:Ev^*, T)← actor_of(Ev,AID),instance_of(C,agreement,T),
instance_of(C,institution,IS,T),holds_at(C,participant, AID,T), role_of(Ev^*,buyer), institution_of(Ev^*, IS).

to specify that the participate event Ev counts as an assign role event Ev^* in the newly created EI for the auction.

The powers of the agents are constrained by the permission norms in the EI: for example an auctioneer is authorized to open an auction only if its starting time has

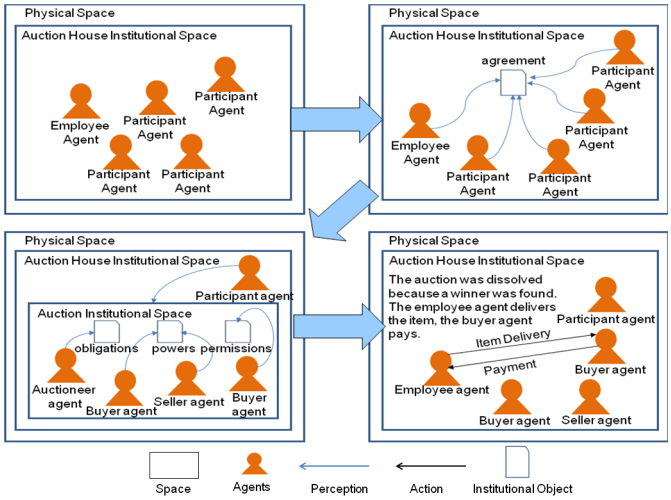

Fig. 4. Interaction in the Auction House Agent Environment

elapsed and if there are at least two agents registered as participants. Fig. 4 represents the interaction taking place in the agent environment represented by the auction house where the auctions are created and dissolved. In particular, as defined in the auction life-cycle in Fig. 3, the e-Health auction is dissolved when an agent wins the auction offering a price that matches the current offer of the seller. To handle the evolution of the auction within the agent environment represented by the auction house, we utilize the following norms:

N1)power:n1[mediates ⇒ start_action:Ev[auctioneer⇒ agent:A, item ⇒ O, starting_price ⇒ P, institutional_space ⇒ IS]@T, check_role ⇒ {A, employee, T}]
N2)power:n2[mediates⇒change_price:Ev[auctioneer⇒agent:A, item ⇒ O, new_price ⇒ Price,institutional_space ⇒ IS]@T, check_role ⇒ {IS, A,auctioneer, T}]
N3)permission:n3[mediates⇒ change_price:Ev[auctioneer⇒agent:A,item ⇒O,new_price ⇒ Price]@T,hasItem⇒{IS,O T} check_role ⇒ {IS,A,auctioneer, T}, hasPrice ⇒ {IS, O, CurrentPrice, T}, lessThan ⇒ {Price, CurrentPrice}]
N4)obligation:n4[mediates ⇒ assign_item:Ev[auctioneer ⇒ Auc, item ⇒ O, buyer ⇒ Buyer, institutional_space ⇒ IS]@T, lastOffer ⇒ {IS,Buyer,O, LastOffer}, currentPrice ⇒ {IS, O,CurrentPrice}, equal ⇒ { LastOffer,CurrentPrice }]
N5)obligation:n5[mediates ⇒ pay:Ev[buyer ⇒ Buyer, amount ⇒ LastOffer, item ⇒ O, institutional_space ⇒ IS]@T, isAssigned ⇒ { IS, Buyer,O,T }, currentPrice ⇒ { IS,O,Price,T }, equal ⇒ LastOffer,Price]

Norm N1 specifies that an agent has the power to start an auction in the auction house space when it is an employee for the auction house, while norm N2 and norm N3 express the power of an agent to change the price of an item within an auction space in which the agent is taking part with the role of auctioneer, and the permission to change the price from the point of view of the auction if the auction has that item and the new price is less than the previous one. Norm N4 expresses the obligation of the auctioneer to assign an item to the winner of the auction, while norm N5 expresses the obligation of a buyer agent to pay for the item assigned by the auctioneer. In the case that the events produced by the agents respect the norms of the institutional space, then the evolution of the Dutch auction institutional space is handled in terms of initiates/4 and terminates/4 clauses that modify the attributes of the Dutch auction whenever an event takes place. For example, the following clauses:

initiates(change_price:Ev, AuID, lastoffer, NewOff)← time(Ev,T), offer(Ev, NewOff), auction(Ev,AuID), value(NewOff, NVal), holds_at(AuID, lastoffer, OldOff, T), value(OldOff, OVal), NVal < OVal.

terminates(makeoffer:Ev, AuID, lastoffer, _)← initiates(makeoffer:Ev, AuID, lastoffer, NewOff).

state that a new offer, is considered the last offer, only if the value of the offer is less than the previous offer. Finally, we introduce a domain dependent count_as/3 to deal with the case of a buyer agent leaving the institutional space of an auction before having paid:

count_as(leave_auction:Ev[institutional_space ⇒ IS],sanction:Ev*, T)← actor_of(Ev,AID),
obligation(pay:Ev**[institutional_space ⇒ IS],T), actor_of(Ev**,AID), institution_of(Ev*, IS), credit_of(Ev*,200).

The count_as/3 above specifies that an agent leaving while an obligation of paying holds in the EI, will be sanctioned of 200 credits. A further count_as/3 clause has been defined to handle the exception of an auctioneer not delivering the good after the auction, but we omit it as it is similar to the clause above.

5 Implementation Issues

For the implementation of the normative systems we adopted a logic programming approach due to the formal and declarative semantics of our model and we implemented it as a Prolog theory that we include in the GOLEM framework [6]. In particular, we utilized a version of the OEC described in [15], which is based on caching the periods of time in which an attribute of an object holds. The top-level implementation for the holds_at/4 in OEC is shown below:

```
holds_at(Obj,Attr,Val,T):- object(Obj,Attr,Val,start(E)), time(E,T1), T1 =< T,
    not (object(Obj,Attr,Val,end(E*)), time(E*,T2), T2>T1, T2<T).
```

where the object/4 assertions store when an attribute have been initiated/terminated at a certain time. A similar definition for the instance_of/3 predicate exists, using instance/3 assertions. This representation brings the advantage that indexing can be performed on both the time interval and the object identifier, meaning that the time to retrieve the attribute of an object is O(1), once the identifier and the interval are known as in our specification. An example of the state of a norm of the normative system to create an auction in the auction house agent environment can be expressed as follows:

```
instance(r1:[IDS,ID,auctioneer, T], power, start(e1)). object(r1:[IDS,ID,auctioneer, T], mediates, open_auction, start(e1)).
object(r1:[IDS,ID,auctioneer, T],template, do(ID,open_auction, [property(institution, IDS)]):T,start(e1)).
object(r1:[IDS,ID,auctioneer, T],check_role,[IDS,ID, auctioneer,T],start(e1)). time(e1,1).
```

The state above specifies that there is an instance of a norm that mediates an event of class open_auction, where the template attribute defines the mediated event. The norm also specifies that it calls the check_role/4 predicate to check if the role of the agent performing the event is the one of auctioneer. Notice that we append the variables that will be called in the check_role/4 predicate in the identifier of the rule, so that we can instantiate their value in the apply_norm/2 meta-predicate as specified below:

```
attempt(E,T):- power(E,T), permitted(E,T), add(E,T).
power(E,T):- E = do(Actor,EventClass, Elements), member(property(institution, IDS), Elements),
        instance_of(IDS, institution,T), holds_at(IDS, rules, ID:Vars, T), instance_of(ID:Vars,power,T),
        holds_at(ID:Vars,template, E:T,T), not(not(apply_norm(ID:Vars,T))).
apply_norm(ID:Vars,T):- holds_at(ID:Vars, Attr, Vars, T), append([Attr],Vars, Var2), Pred =.. Var2, Pred.
```

The attempt/2 predicate implemented above checks if the agent has the power and the permission to produce an event in the environment. If this is the case the add/2 predicates add object/4 or instance/3 assertions to the Prolog database, according to the effects of the event. The power/2 predicate, checks if there is a norm that specifies if the agent has the power to produce a certain event with respect to a certain institution.

To do so, the power/2 predicate utilizes the apply_norm/2 meta-predicate to check if the norm specifies any constraint that prevents the agent from performing the action. For example, we implement the check_role/4 constraint as follows:

```
check_role(IDS, ID,Role,T):- instance_of(ID,agent,T), instance_of(IDS,institution,T),
                             holds_at(IDS, roles,RID,T), instance_of(RID,Role,T), holds_at(RID, agent, ID,T).
```

The check_role/4 predicate implemented above checks if an agent identified with the variable ID, has a certain role Role in an EI IDS at a certain time T.

6 Related work

Fornara et al. have developed OCeAN [12], a meta-model for the specification of artificial institutions, and an Agent Communication Language (ACL) to model open interaction systems where heterogeneous software and human agents interact. The OCeAN meta-model consists of the following components: (i) constructs to define the core ontology of an institution, (ii) roles and of events; (iii) the counts-as relation, which is necessary for the concrete performance of institutional actions; (iv) and norms. One difference between MANET and the OCeAN meta-model is that we consider institutions as first-class entities, which allow the perception of their components (e.g. norms, objects and sanctions) that are also described as first-class entities. Another difference is related to the types of events that are possible inside an institution. In the OCeAN meta-model only communication events are considered, whereas in our approach we define a more detailed schema of events in order to describe all the possible situations during the evolution of an open MAS.

Artikis and Sergot in [3] present a model of executable specifications of open MAS where open MAS are considered instances of normative systems. The authors represent the social constraints (laws) of the system in terms of physical capabilities, institutional power, permission and prohibition as well as sanctions and enforcement policies. In our model we adopt a very similar model of institutional rules based on powers which are dependent on permissions, obligations and sanctions. However, in our work we consider institutional rules as first-class entities which can be observed by the agents, allowing them to reason about the normative constraints of the open MAS.

In [23] Urovi and Stathis define the MAGE framework. MAGE uses the OEC formalism to represent games as first-class entities that evolve in time. Such games are interconnected between each others in a hierarchy composed of atomic games and composite games. The state of the composite games is defined by the relationships between the atomic games and their transitions and the agents can perceive the legal actions in a game at a certain time. MANET institutional spaces correspond to MAGE games which also evolve due to the production of events. The main difference between MANET and MAGE is that we include the possibility of defining institutional objects, such as norms and agreements, allowing for norms to have a structure and be perceivable, meaning that the agents can reason about whether or not complying with a norm is to their best interest.

Also related to our work is the work of Piunti et al. to unify at one model the concepts of agents, organizations and environments [19]. This model allows for designing

and programming an environment in terms of a dynamic set of first-class computational entities called artifacts, collected in workspaces. Artifacts represent resources and tools that agents can dynamically instantiate, share and use to support their individual and collective activities. The notions of artifacts and workspaces have similarities with these of objects and spaces. But unlike Piunti et al. spaces in our approach are not just the containers of agents and objects modeling the locality of the application domain but contrariwise they are the main enforcers of the law and regulators of the MAS evolution.

7 Conclusion and Future Works

We presented a meta-model that describes institutions as social agent environments [24] by extending upon the OEC formalism [15], based on the concept of agent environment as presented in [6]. In particular we presented a model that can handle the life-cycle of multiple institutions at runtime, where the institutions are represented as first-class objects that the agents can perceive. Moreover, our model supports the definition of institutional objects, such as agreements, sanctions and norms that the agents can perceive as part of an electronic institution. We presented this model utilizing an e-Health marketplace based on Dutch auctions as a motivating example.

There are several directions that is worth exploring for future works. First of all, we plan to extend our framework to handle dynamic norm change within the institution when an agent society requires it due to an external exception. Secondly, we plan to extend our framework to manage institutions in distributed settings. As recognized by Artikis et al. in [2], learning the rule of an institution is recognized as a problem, as a consequence we plan to investigate machine learning approaches that can make use of our model to create cognitive agents capable to learn how to interact within multiple heterogeneous institutions.

ACKNOWLEDGEMENTS

Our work is partially supported by the SER project Open Interaction Frameworks: Towards a Governing Environment under the Agreement Technologies COST Action IC0801.

References

1. T. Ägotnes, W. van der Hoek, and M. Wooldridge. Normative system games. In *AAMAS '07: Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, pages 1–8, New York, NY, USA, 2007. ACM.
2. A. Artikis, G. Paliouras, F. Portet, and A. Skarlatidis. Logic-based representation, reasoning and machine learning for event recognition. In *DEBS*, pages 282–293, 2010.
3. A. Artikis and M. Sergot. Executable Specification of Open Multi-Agent Systems. *Logic Journal of the IGPL*, 18(1):31–65, 2010.
4. K. S. Barber and J. Kim. Soft security: Isolating unreliable agents from society. In *Trust, Reputation, and Security*, pages 224–233, 2002.

5. G. Boella, G. Pigozzi, and L. van der Torre. Normative systems in computer science - ten guidelines for normative multiagent systems. In G. Boella, P. Noriega, G. Pigozzi, and H. Verhagen, editors, *Normative Multi-Agent Systems*, number 09121 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2009.
6. S. Bromuri and K. Stathis. Distributed Agent Environments in the Ambient Event Calculus. In *DEBS '09: Proceedings of the third international conference on Distributed event-based systems*, New York, NY, USA, 2009. ACM.
7. J. Campos, M. Lpez-Snchez, J. Rodrguez-Aguilar, and M. Esteva. Formalising situatedness and adaptation in electronic institutions. In J. Hbner, E. Matson, O. Boissier, and V. Dignum, editors, *Coordination, Organizations, Institutions and Norms in Agent Systems IV*, volume 5428 of *Lecture Notes in Computer Science*, pages 126–139. Springer, 2009.
8. W. Chen and D. S. Warren. C-logic of Complex Objects. In *PODS '89: Proceedings of the eighth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, pages 369–378, New York, NY, USA, 1989. ACM Press.
9. M. Esteva, J. A. Rodríguez-Aguilar, C. Sierra, P. Garcia, and J. L. Arcos. On the formal specifications of electronic institutions. In *Agent Mediated Electronic Commerce, The European AgentLink Perspective.*, pages 126–147, London, UK, 2001. Springer-Verlag.
10. M. Esteva, B. Rosell, J. A. Rodríguez-Aguilar, and J. L. Arcos. Ameli: An agent-based middleware for electronic institutions. volume I, pages 236–243. ACM, ACM, 2004.
11. N. Fornara and M. Colombetti. Specifying artificial institutions in the event calculus. In *Handbook of Research on Multi-Agent Systems: Semantics and Dynamics of Organizational Models*, pages 335–366, 2009.
12. N. Fornara, F. Viganò, M. Verdicchio, and M. Colombetti. Artificial institutions: a model of institutional reality for open multiagent systems. *Artif. Intell. Law*, 16(1):89–105, 2008.
13. T. Gruber. Collective knowledge systems: Where the Social Web meets the Semantic Web. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6(1):4–13, 2007.
14. J. Hendler and T. Berners-Lee. From the semantic web to social machines: A research challenge for ai on the world wide web. *Artif. Intell.*, 174(2):156–161, 2010.
15. F. N. Kesim and M. Sergot. A Logic Programming Framework for Modeling Temporal Objects. *IEEE Transactions on Knowledge and Data Engineering*, 8(5):724–741, 1996.
16. R. Kowalski and M. Sergot. A logic-based calculus of events. *New Gen. Comput.*, 4(1):67–95, 1986.
17. S. Modgil, N. Faci, F. Meneguzzi, N. Oren, S. Miles, and M. Luck. A framework for monitoring agent-based normative systems. In *AAMAS '09: Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems*, pages 153–160.
18. J. Pitt, A. Mamdani, and P. Charlton. The open agent society and its enemies: a position statement and research programme. *Telematics and Informatics*, 18(1):67–87, 2001.
19. M. Piunti, O. Boissier, J. F. Hubner, and A. Ricci. Embodied organizations: a unifying perspective in programming agents, organizations and environments. In *11th Workshop on Coordination, Organization, Institutions and Norms in Multi-Agent Systems*, 2010.
20. J. R. Searle. *The construction of social reality.* Free Press, New York, 1995.
21. C. Strachey. Fundamental concepts in programming languages. *Higher Order Symbol. Comput.*, 13(1-2):11–49, 2000.
22. T. Stratulat, J. Ferber, and J. Tranier. Masq: towards an integral approach to interaction. In *AAMAS (2)*, pages 813–820, 2009.
23. V. Urovi and K. Stathis. Playing with agent coordination patterns in MAGE. In *Coordination, Organization, Institutions and Norms in Agent Systems (COIN@AAMAS09)*, Budapest, Hungary, May 2009.
24. D. Weyns, A. Omicini, and J. Odell. Environment as a first class abstraction in multiagent systems. *Autonomous Agents and Multi-Agent Systems*, 14(1):5–30, 2007.
25. M. Wooldridge. *MultiAgent Systems*. John Wiley and Sons, 2002.