# First-Class Electronic Institutions in the Object Event Calculus

Charalampos Tampitsikas[1], 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 can evolve over time, whose rules, in terms of powers, permissions and obligations, are perceivable as first class entities by agents belonging to the institution. To represent the institution we utilise the Object Event Calculus (OEC) formalism that deals with the evolution of complex structures in time. We extend the OEC to deal with the mediation of the events and with the perception of complex structures and events within the institution. As a consequence, our framework presents 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. We use an eHealth 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

During the last decade there have been many efforts towards the reduction of health costs. The public health care system rapidly is absorbing an ever-increasing share of the gross domestic product. According to the latest available data, hospital costs account for approximately 35% physicians, 25% drugs, 15% medical equipment and supporting IT tools while the rest 25% concerns various secondary health costs. Although there have been numerous research projects trying to reduce the hospital costs without losing the quality of offered services, the cost of drug and medical equipment supplies is continuing to increase.

An automatic negotiation mechanism would enhance the cooperation of medical units as well as the optimal use of their medical supplies, leading to a reduction of medical supplies costs. Such negotiation platform for hospitals and medical units could benefit if designed as an *open multi-agent system*. A MAS can be considered open [14] when it satisfies a set of properties such as:

– Agents are free to join and leave at any time;
– Agents represent different stakeholders with different objectives.

The above definition can directly apply to a negotiation platform between hospitals. Each hospital unit is free to enter or leave the negotiation platform according to its will, while at the same time different hospitals have different goals and different strategies in order to achieve them.

As Barber and Kim specify in [4], we can say that as a consequence of the nature of open MAS, to handle them it is necessary to take into consideration a set of issues. In particular 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 due to different ownership, there is no central authority in a MAS. Normative systems, or as Ägotnes et al. specify in [1], systems were social rules apply, try to tackle the issues presented by open MAS by defining rules to coordinate heterogeneous agents. In particular, as noted by Modgil et al. in [13], there are two approaches to define normative systems:

– regimentation based normative systems, in which a set of rules and protocols are defined to coordinate the behaviour of the agent;
– 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.

On one hand, Modgil et al. argue that the enforcement based normative systems are more flexible than regimentation based normative systems as for the latter 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. On the other hand, 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. Proper coordination of agents is crucial at a negotiation platform between hospitals. At the same time, it is important to ensure the autonomy of the hospital agents. Thus, we argue that enforcement based normative systems is the appropriate approach for such a platform, since it offers the agents the possibility not to follow some of the rules of the system but simultaneously it applies a set of sanctions as a response to undesirable actions to secure the system stability.

In this paper we present a meta-model for enforcement based normative systems where we try to apply a flexible approach to the issue of defining rules for the normative systems. We envisage electronic institutions (EC) as an approach to normative systems that can be created at runtime, but where their rules are first class citizens [16], meaning that the agents can observe the rules and reason about them at runtime.

The contribution and significance of the resulting system is to provide a model of enforcement based electronic institutions that allows to present the norms, commitments, sanctions and electronic institution themselves as first class abstractions that the agent can observe, reason about and manipulate. We illustrate this at an eHealth marketplace example.

The reminder of this paper is structured as follows: Section 2 presents a motivating case study; Section 3 is a description of the EI life-cycle and of its features within an agent environment; Section 4 shows how we apply our model within a eHealth market place based on Dutch auctions; Section 5 puts our work in comparison with existing EI frameworks; finally Section 6 concludes this paper and shows some possible future work directions.

## 2 Motivating Case Study

To illustrate our meta-model, we take as a motivating case study a network of hospitals which need to trade about several different medical items. The general medical challenges that we want to address with our negotiation system are the following ones:

- How do we exploit blood and medicament overplus?
- How do we exploit medicaments whose expiration date is approaching? How is it possible to supply them at another hospital unit?
- How do we enhance the cooperation and the coordination of hospital units during urgent incidents?
- How do we reduce the costs for drug supplies at the hospitals?
- How do we ensure the cost and time efficient accomplishment of inter-hospital requests?

In particular, our concrete objective is to create a negotiation platform that supports the trading of three different categories of products such as (a) medicaments, (b) blood, (c) medical equipment. This negotiation platform is considered as a marketplace where multiple dutch auctions can occur simultaneously. Each hospital can start an auction in order to trade a product or it can join an already running auction in order to express interest for buying a product. A number of legal restrictions can apply at a marketplace between different hospital units. Some of them are related to the rights of hospitals to re-sell medical supplies. How we could overcome these legal obstacles is still a point for further study. For the purpose of this paper we propose a solution based on the replacement of real monetary units with virtual ones. Each product belonging to one of the three categories mentioned above, will be negotiated with a starting price expressed in terms of credits. This implies that each hospital unit will have an account balance with the other hospitals on credits. The introduction of the notion of credits instead of real monetary units is crucial in order to avoid legal restrictions on trading of medical supplies.

Given this setting, we do not make any assumption on the kind of entities performing the negotiation: they can be either human entities or software agents interacting with the negotiation platform, although it is important to say that in an emergency scenario one would expect that the trade is handled by human agents rather than from software agents. Within the next sections of this paper we will utilise the motivating scenario as a driving example to present our system architecture and behaviour.

## 3 Unifying Normative Systems and Agent Environments

The agent environment can be used as an entity that mediates the interaction between agents taking part to a distributed MAS. According to Weyns [20] the agent environment coordinates and constrains the actions that the agents can perform at a given time.

At the same time the agent environment provides to the agents those interfaces that are necessary to perceive the environment and the other agents situated in it.

Institutions [19] have many properties of agent environments. First of all, an institution defines a set of rules that mediate the interaction taking place between the agents. Secondly, they have a description in terms of a declarative ontology that defines the concepts that can be expressed in the institution. Thirdly, institutions are complex structures that evolve over time.

From our perspective an institution defines three main concepts (adapted from [3]) for mediation:

- Power: an agent can perform a designated action in a context, which creates or changes an institutional fact.
- Obligation: it expresses the idea that at a given time the agent should produce an action as specified by the rules of the normative system. Obligations also include the concept of prohibition which refers to a negative obligation; that means an action that is forbidden by the rules of the system.
- Permission: the concept of permission is both related to the state of the institution and to the concept of power. When an agent tries to perform an action for which it has the power, this action will have success only if the action is also permitted in the institution at that point in time. In other words, an agent can exercise the power he has obtained entering the institution, if and only if the institutional conditions permit it. In general, in the design of institution it is useful to specify conditional powers [9], that is, powers that start to hold if certain conditions start to hold or if certain events happen in the system.

On one hand, despite the fact that normative systems provide a level of abstraction in terms of social rules amongst agent societies, it is not always clear how these rules mediate the interaction in a MAS. On the other hand, the governor agents suggested by some enforcement based normative aproaches [7] have a number of disadvantages: they put extra computational load to the MAS by introducing a new entity which adds complexity to the whole system, while at the same time mixing the concepts of agent and the concept of infrastructure.

Another difference between the concept of agent environment and the current approaches in normative models is the lack of mechanisms in normative systems that allow the perception of institutional entities and events. The existence of a way to notify the agents that a new institution has been created and also to allow agents to perceive it, is crucial. The perception of an institution could allow the agents to decide whether participating in the institution would benefit the accomplishment of their goals.

Finally, 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 furthermore they are not always the appropriate means for the description of the evolution of the system.

### 3.1 Modelling Institutions in the Object Event Calculus

In our formalisation we assume that institutions can be composed of three basic components: agents, objects and spaces.

In [5], the notion of agent is used to describe active entities with a separation between a cognitive mind and a physical body with sensors and effectors while the notion of object is used to describe virtual entities or virtualizations of external resources. From the standpoint of institutions, we utilise the same definition for agents as in [5], but we extend the object model by allowing the creation of *institutional objects*, which are objects existing only in common agreement amongst the agents of an institution, and can be categorised as: a) objects that can exists within the communication amongst the agents, such as the good to trade in a market; b) objects that represent commitments between one or more parties; c) objects that represent sanctions for the incorrect behaviour of the agent in the institution; d) objects that represent norms of an institution e) objects that represent institutional spaces, f) and finally roles of agents within an institution. Moreover, physical objects can be considered as institutional ones when they obtain institutional attributes during the evolution of the agent environment. Fig. 1 shows the relationships between the entities of our model.

For our model an environment is a set of spaces. Such spaces include objects and agents, and contain information about their topology and configuration. Moreover, we do not restrict our definition only to physical environment and physical spaces. Exactly as in human world, besides the physical spaces there also exist social spaces [17] in our model we propose:

- Physical spaces, describing the physical topology of the system.
- Institutional spaces, describing the institutions in a normative system.

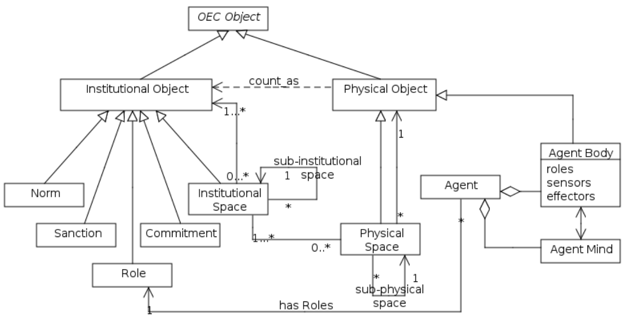


**Fig. 1.** Entity Categories

It is important to stress that in our framework norms, commitments and sanctions are expressed as complex structures themselves, meaning that they can be deployed as objects in an institutional space. A space is also a boundary for the events perfomed by the agents, meaning that the effects of an event produced inside one space hold only for that space. Since spaces are the natural boundaries and containers of the events, they can detect 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 retreive the information of the appropriate sanction objects and will apply them to the agent that did not comply with the rules of the system.

Spaces can be folded inside other spaces or can be distributed across more than one spaces creating complex topologies. In the normative systems, the agents' interactions can create new institutional realities. In this paper, we argue that each time a new institution is born, a new institutional space is being created, which includes the norms, the objects and the agents of the institution.

To represent the state of the entities at a given time, we will use the C-logic formalism [6] as it is a convenient formalism to represent complex structures and it has a direct translation to first-order logic. According to the above definition of spaces, to describe the state of an institutional space at a given time we utilise the following C-logic structure:

```
institutional_space:is1[
    agents ⇒ { agent:a1[ roles ⇒ {role:r1, role:r2} ]
    agent:a2[ roles ⇒ {role:r1, role:r3} ] }
    agent:a3[ 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, a3, a set of institutional objects that the agents can manipulate in the institution and a set of sub institutional spaces. We will explain the meaning of all these concepts in the remainder of the paper. The term described above refers to the following first order logic schema:

```
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,instituational_objects, o1,start(e1)).
objects(is1,instituational_objects,o2,start(e1)).
object(is1,instituational_spaces, s3,start(e1)).
objects(is1,instituational_objects,s4,start(e1)).
```

that defines the structure of an OEC object of type institutional_space. Similarly the following C-logic structures:

```
commitment:c1[
        action ⇒ create_auction:Ev[auction ⇒ IDA, item ⇒ IT],
        deadline ⇒ 3000,
        contractor ⇒ IDContractor,
        debtor ⇒ IDDebtor]
```

```
power:p1(open_auction:Ev[actor ⇒ IDActor]@T)[
  mediates ⇒ open_auction,
  check_role ⇒ {IDActor, employee}]

sanction:s1[
    agent ⇒ ag1
    credits ⇒ 200]
```

decribe the commitment between two agents to open auction for a specific product after a period of time, the power of an agent that enters the institution as an employee to open an auction and the contents of a sanction. In particular we extend the semantic of the C-logic formalism with parametrised C-logic structures, where the labels of the attributes, in the case of norms, have also the meaning of predicates. We will show later in this paper how such norms are applied when the agents execute an action.

### 3.2 Evolution of Institutional Spaces

To unify the two concepts of agent environment and institution, we will need to define how to represent the state of an institution, how to perceive its state and the state of the agents taking part in the interaction, and how to represent the events. In the reminder of this Section we will illustrate our approach in defining institutions by means of the Object Event Calculus [11], following the motivating example presented in Section 2. In particular the Object Event Calculus is a dialect of the Event Calculus (EC) [12] that is suitable to represent the evolution in time of complex structures by means of events.

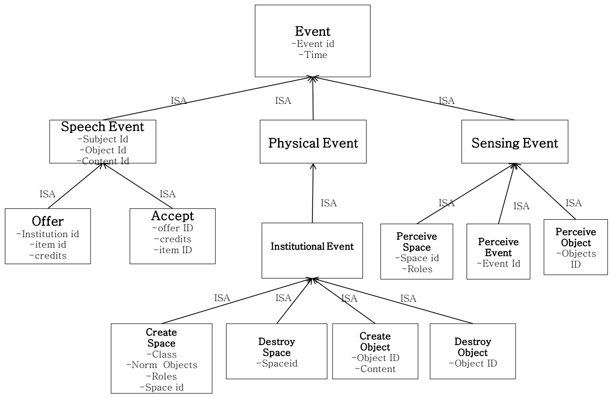


**Fig. 2.** Events Hierarchy

The event schema that we take into consideration in our system are shown in Fig. 2. We distinguish between three kind of events that are speech events, physical events and sensing events. This distinction is not new and it was already presented in [5], in this paper we further extends the hierarchy of events introducing *institutional events* that we will discuss in details later. To present how the institution state evolves in time due to the production of events in the institution we use the Object Event Calculus predicates, as 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 [11]. 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 superclasses. 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.

Event descriptions themselves are specified as complex terms. For example, the event description below:

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

represents an instutional action of agent ag1 who attempts to open an auction about an item item1 of class blood_batch. 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 next how the affordances 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) ←              initiates(E, auction, Au, item, I) ←
        open_auction:E,                    open_auction:E [item ⇒ I].
        auction_of(E,Obj).
```

In this way the new position of the agent has been initiated as a result of the move. The complete description of the event's effects also requires to terminate the attribute holding the old position of the agent; this is handled in OEC by the general rule described in clause C9.

### 3.3 Acting and Perceiving inside Institutions

The representation in terms of C-logic structures of the institutions allow us to have multiple institutions recursively embedded within each others. In order to act within an institution, the agents have to be aware of which is the institution where they want to perform an action. Moreover, the agents' actions are going to be mediated by the rules of the institution. As a consequence we say that in order to be performed within an institution, an action has to be attempted in that institution first. We specify the modification of the institution 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 institution 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.
```

The consequence of defining the happens/2 relation in this way is that every event in the system has to be directed to a normative system of which the agent must be aware of. 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 systems. 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", after the time T* at which the obligation should have been satisfied, 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 institutions, 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) perceive(E, S, T) ←
        happens(E, T),
        perceive_institutional_space(E),
        E[sensor_of⇒ S, focus⇒ Focus,institutional_space ⇒ IS],
        solve_at(IS,Focus, T).
```

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 commitments, sanctions and norms, in the institutional space.

## 4 Modelling an e-Health Marketplace

To model the eHealth 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 institution representing a Dutch auction in order to sell medicaments in a eHealth marketplace. In particular, we adapted the Dutch auction as presented by Viganò et al. in [19] to our formalism based on the OEC and we introduce norms expressed in terms of objects of the Object Event Calculus formalism. The Dutch auction 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:

- 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.
- Participant: it is an agent that can express interest for an auction, becoming buyers within the auction.
- Buyers: it is an agent that is participating to an auction in the auction house agent environment trying to buy an item of interest.
- Auctioneer: it is an agent that coordinates an auction on behalf of a seller agent.
- Seller: it is an agent that delegates an auctioneer to sell an item in the Dutch auction.

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

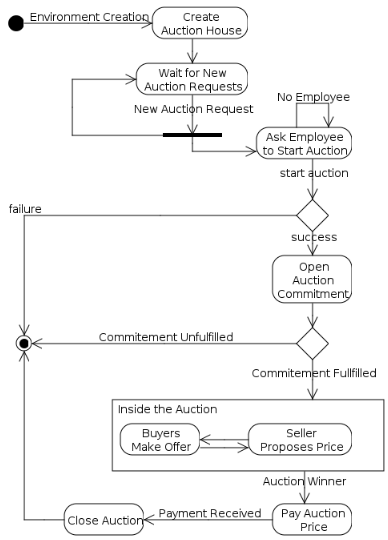


**Fig. 3.** Auction House Environment State Chart

In Fig. 3, 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 a commitment institutional object in the agent environment.

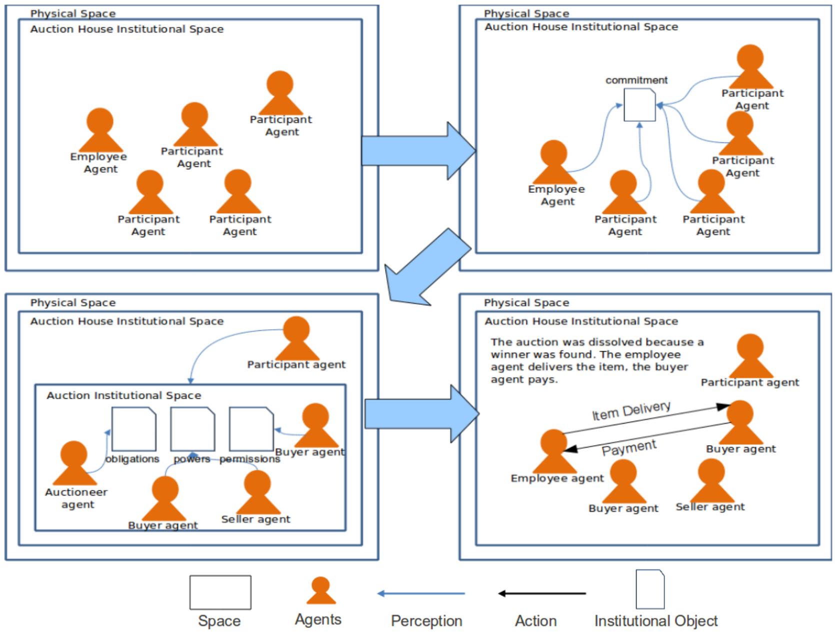


**Fig. 4.** Interaction in the Auction House Agent Environment

This commitment is perceivable by all the agents inside the auction house, which can modify its attributes only with institutional actions. In C-logic terms this commitment object is described as follows:

```
commitment:c1[
        object ⇒ blood_batch: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 blood batch item b1. When a commitment 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 commitment object with an institutional action. In particular, when the deadline of the commitment expires, we utilise a count_as/3 as follows:

```
count_as(participate:Ev,assign_role:Ev*, T)←
                actor_of(Ev,AID),
                instance_of(C,commitment,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 institution for the auction.

The powers of the agents are constrained by the permissions norms in the institution: for example an auctioneer is authorized to open an auction only if its starting time has 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 eHealth 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 utilise the following norms.

```
N1) power:n1(start_action:Ev[auctioneer⇒ agent:A,
        item ⇒ O,
        starting_price ⇒ P
        institutional_space ⇒ IS]@T)[
            check_role ⇒ {A, employee, T}
]

N2) power:n2(change_price:Ev[auctioneer⇒ agent:A,
        item ⇒ O,
        new_price ⇒ Price,
        institutional_space ⇒ IS]@T)[
            check_role ⇒ {IS, A,auctioneer, T},
]
N3) permission:n3(change_price:Ev[auctioneer⇒ agent:A,
        item ⇒ O,
        new_price ⇒ Price]@T)[
            check_role ⇒ {IS,A,auctioneer, T},
            hasItem ⇒ {IS, O ,T},
            hasPrice ⇒ {IS, O, CurrentPrice, T},
            lessThan ⇒ {Price, CurrentPrice},
]

N4) obligation:n4(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(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 environment only if it is already 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 institutional space representing an auction 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. Norms N4 and N5 express the obligations of the agents participating in the auction. More precisely, 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.

Finally, we introduce the following 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 institution, will be sanctioned of 200 credits. A further count_as/3 clause has been defined to handle those exceptions when the auctioneer does not deliver the good after the auction, but we omit it as it is similar to the clause above.

## 5 Related Work

Colombetti and Fornara have developed OCeAN [10] , a meta-model for the specification of artificial institutions, and a complete Agent Communication Language (ACL) that can be used to model open interaction systems where heterogeneous software and human agents may interact. At its present stage of development, the OCeAN metamodel of artificial institutions consists mainly of the following components: (i) the constructs necessary to define the core ontology of an institution, (ii) a treatment of roles and of events, two fundamental concepts that are common to all artificial institution and that are used in the specification of powers and norms; (iii) a treatment of the counts-as relation, which is necessary for the concrete performance of institutional actions; (iv) and a treatment of norms, used to impose obligations and prohibitions to perform certain actions on agents interacting with the system. Our work is related to the OCeAN meta-model. One difference between our approach and the OCeAN meta-model is that we consider institutions as first-class abstractions which allow the perception of their components (e.g. norms, objects and sanctions) which 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 strongly depended on permissions, obligations and sanctions. However, in our work we consider institutional rules as first-class entitites which can be observed by the agents, allowing them to reason about the normative constraints of the open MAS.

In [18] Urovi and Stathis define the MAGE framework. In such framework they make use of the game metaphore to represent protocols. In particular they use the OEC formalism to represent games as first class abstraction 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. Urovi and Stathis also define how the agents can perceive the legal actions in a game at a certain time. With respect to Urovi and Stathis, we can compare our concept of institutional space with the concept of game. Also institutional spaces are defined in terms of the OEC and their state evolves due to the production of events. The main difference we have with the work of Urovi and Stathis is that for the moment we do not consider atomic or composite electronic institutions. Another difference is that we include the possibility of defining institutional objects, such as norms and commitments, that can be observed as part of an institution. An advantage of our formalisation with respect to the one proposed by Urovi and Stathis, is that we allow for norms to be observed, meaning that the agents can reason about whether complying with a norm is to their best interest.

Also related to our work is the work of Ricci with the A&A (Agents and Artifacts) meta-model [15]. This approach 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. Unlike Ricci we do not assume in our model only one type of computational entities. The spaces in our approach are not just the containers of agents and objects but, contrariwise, they are first-class entities which are also perceivable by the agents. These spaces are the boundaries of the events that mediate the interaction of the agents.

## 6 Conclusion and Future Works

We presented a metamodel to unify the concept of agent environment [20] with the concept of institution [8] by extending upon the OEC formalism [11] and on the concept of agent environment as presented in [5]. 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 commitments, sanctions and norms that the agents can perceive as part of an institution. We presented this model utilising an eHealth 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 recognised by Artikis in [2], learning the rules of an institution is recognised 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. Thomas Ägotnes, Wiebe van der Hoek, and Michael 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. Alexander Artikis, Georgios Paliouras, François Portet, and Anastasios Skarlatidis. Logic-based representation, reasoning and machine learning for event recognition. In Jean Bacon, Peter R. Pietzuch, Joe Sventek, and Ugur Çetintemel, editors, *DEBS*, pages 282–293. ACM, 2010.
3. Alexander Artikis and Marek Sergot. Executable Specification of Open Multi-Agent Systems. *Logic Journal of the IGPL*, 18(1):31–65, 2010.

4. K. Suzanne Barber and Joonoo Kim. Soft security: Isolating unreliable agents from society. In *Trust, Reputation, and Security*, pages 224–233, 2002.
5. S. Bromuri and K. Stathis. Situating Cognitive Agents in GOLEM. In *Engineering Environment-Mediated Multiagent Systems (EEMMAS'07)*. Springer, Oct 2007.
6. 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.
7. Marc Esteva, Bruno Rosell, Juan A. Rodríguez-Aguilar, and Josep Lluis Arcos. Ameli: An agent-based middleware for electronic institutions. volume I, pages 236–243. ACM, ACM, 2004.
8. Nicoletta Fornara and Marco Colombetti. Specifying and enforcing norms in artificial institutions. In Lin Padgham, David C. Parkes, Jörg P. Müller, and Simon Parsons, editors, *AAMAS (3)*, pages 1481–1484. IFAAMAS, 2008.
9. Nicoletta Fornara and Marco 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.
10. Nicoletta Fornara, Francesco Viganò, Mario Verdicchio, and Marco Colombetti. Artificial institutions: a model of institutional reality for open multiagent systems. *Artif. Intell. Law*, 16(1):89–105, 2008.
11. Nihan Kesim. *Temporal Objects in Deductive Databases*. PhD thesis, Imperial College, 1993.
12. R Kowalski and M Sergot. A logic-based calculus of events. *New Gen. Comput.*, 4(1):67–95, 1986.
13. Sanjay Modgil, Noura Faci, Felipe Meneguzzi, Nir Oren, Simon Miles, and Michael 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, Richland, SC, 2009. International Foundation for Autonomous Agents and Multiagent Systems.
14. Jeremy Pitt, Abe Mamdani, and Patricia Charlton. The open agent society and its enemies: a position statement and research programme. *Telematics and Informatics*, 18(1):67–87, 2001.
15. Alessandro Ricci, Michele Piunti, and Mirko Viroli. Environment programming in multiagent systems – an artifact-based perspective. *Autonomous Agents and Multi-Agent Systems*, 2010.
16. Christopher Strachey. Fundamental concepts in programming languages. *Higher Order Symbol. Comput.*, 13(1-2):11–49, 2000.
17. Tiberiu Stratulat, Jacques Ferber, and John Tranier. Masq: towards an integral approach to interaction. In *AAMAS (2)*, pages 813–820, 2009.
18. Visara Urovi and Kostas Stathis. Playing with agent coordination patterns in mage. In *COIN@AAMAS&IJCAI&MALLOW*, pages 86–101, 2009.
19. Francesco Viganò, Nicoletta Fornara, and Marco Colombetti. An Event Driven Approach to Norms in Artificial Institutions. In Olivier Boissier, Julian Padget, Virginia Dignum, Gabriela Lindemann, Eric Matson, Sascha Ossowski, Jaime Sichman, and Javier Vzquez-Salceda, editors, *Coordination, Organizations, Institutions, and Norms in Multi-Agent Systems*, volume 3913, pages 142–154. Springer, 2006.
20. Danny Weyns, Andrea Omicini, and James Odell. Environment as a first class abstraction in multiagent systems. *Autonomous Agents and Multi-Agent Systems*, 14(1):5–30, 2007.