Preliminary Proceedings

of

COIN 2007 @ DURHAM

Coordination, Organization, Institutions and Norms

Editors: Pablo Noriega, Julian Padget

Acknowledgements

We gratefully acknowledge the invaluable assistance of the program committee in evaluting the submissions to COIN2007@DURHAM.

Guido Boella *(Universitá di Torino, IT)*
Olivier Boissier *(EMSE Saint-Etienne, France)*
Cristiano Castelfranchi *(ISTC, Italy)*
Stephen Cranefield, *(University of Otago, NZ)*
Marina De Vos *(University of Bath, UK)*
Virginina Dignum *(Utrecht University, The Netherlands)*
Marc Esteva *(University of Technology Sydney, AU)*
Nicoletta Fornara *(Universitá della Svizzera-Italiana Lugano, CH)*
Carl Hewitt, *(MIT, USA)*
Christian Lemaitre *(Universidad Autonónoma México, MX)*
Victor Lesser *(University of Massachussets, USA)*
Gabriela Lindemann, *(Universität Humboldt, DE)*
Fabiola López *(BUAP, MX)*
Michael Luck *(King's College, University of London, UK)*
Eric Matson *(Wright State, USA)*
Tim Norman *(University of Aberdeen, UK)*
Eugénio Oliveira *(Universidade do Porto, PT)*
Andrea Omicini *(Universitá di Bologna, IT)*
Sascha Ossowski *(Universidad Rey Juan Carlos, Madrid, Spain)*
Juan Antonio Rodríguez *(IIA CSCI, Spain)*
Marek Sergot *(Imperial College, UK)*
Carles Sierra *(IIIA CSCI, Spain)*
Mario Verdicchio *(Universitá di Bergamo, IT)*
Wamberto Vasconcelos *(University of Aberdeen, UK)*
Javier Vázquez-Salceda *(Universitat Politècnia de Catalunya, ES)*

Additional Reviewers

Holger Bilhards, Andrés García-Camino, Jelle Herbrandy, Henrique Lopes Cardoso, Maite López-Sánchez, Luca Tummolini.

COIN2007@DURHAM Organization

Program Chairs:	Pablo Noriega *(IIA CSCI, Spain)*
	Julian Padget *(University of Bath, UK)*
Local Organization:	Rafael Bordini *(University of Durham, UK)*
	Berndt Farwer *University of Durham, UK*
	Tricia Shaw *University of Durham, UK*

Table of Contents

Using Case-Based Reasoning in Autonomic Electronic Institutions

Eva Bou[1] and Maite López-Sánchez[2] and J. A. Rodríguez-Aguilar[1]

[1] IIIA, Artificial Intelligence Research Institute
CSIC, Spanish National Research Council, Campus UAB 08193 Bellaterra, Spain,
email: {ebm, jar}@iiia.csic.es
[2] WAI, Volume Visualization and Artificial Intelligence, MAiA Dept., Universitat de Barcelona, email: maite@maia.ub.es

Abstract. Electronic institutions (EIs) define the rules of the game in agent societies by fixing what agents are permitted and forbidden to do and under what circumstances. Autonomic Electronic Institutions (AEIs) adapt their regulations to comply with their goals despite coping with varying populations of self-interested external agents. This paper presents a self-adaptation model based on Case-Based Reasoning (CBR) that allows an AEI to yield a dynamical answer to changing circumstances.

1 Introduction

The growing complexity of advanced information systems in the recent years, characterized by being distributed, open and dynamical, has given rise to interest in the development of systems capable of self-management. Such systems are known as self-* systems [1] , where the * sign indicates a variety of properties: self-organization, self-configuration, self-diagnosis, self-repair, etc. A particular approximation to the construction of self-* systems is represented by the vision of autonomic computing [2], which constitutes an approximation to computing systems with a minimal human interference. Some of the many characteristics of an autonomic system are: it must configure and reconfigure itself automatically under changing (and unpredictable) conditions; it must aim at optimizing its inner workings, monitoring its components and adjusting its processing in order to achieve its goals; it must be able to diagnose the causes of its eventual malfunctions and repair itself; and it must act in accordance to and operate into a heterogeneous and open environment.

Electronic Institutions (EIs) [3] have been proved to be valuable to regulate open agent systems. EIs define the rules of the game by fixing what agents are permitted and forbidden to do and under what circumstances. We have defined Autonomic Electronic Institutions (AEIs) as an EI with autonomic capabilities that allows it to adapt its regulations to comply with institutional goals despite varying agent's behaviours [4]. Thus, an AEI has to self-configure its regulations to accomplish its institutional goals. In previous work [4] we have learned those regulations that best accomplished the institutional goals for a collection of simulated agent populations. This paper extends that work with a Case-Based Reasoning (CBR) approach that allows an AEI to self-configure its regulations for any agent population. Since our hypothesis is that populations that behave similarly can be regulated in a similar manner, the CBR approach helps us identify populations that behave similarly and subsequently retrieve the "control" parameters for an AEI to regulate it.

The paper is organized as follows. In section 2 we describe the notion of autonomic electronic institutions. Section 3 details the learning model that we propose and how an AEI uses CBR. Section 4 describes the case study employed as a scenario wherein we have tested our model. Section 5 provides some empirical results. Finally, section 6 summarizes some conclusions and related work and outlines paths to future research.

2 Autonomic Electronic Institutions

In general, an EI [3] involves different groups of agents playing different roles within scenes in a performative structure. Each scene is composed of a coordination protocol along with the specification of the roles that can take part in the scene.

According to [3] an EI is solely composed of: a dialogic framework (DF) establishing the common language and ontology to be employed by participating agents; a performative structure (PS) defining its activities along with their relationships; and a set of norms (N) defining the consequences of agents' actions. We have extended the notion of EI to support self-configuration, in the sense of regulation adaptation. In this manner in [4] we incorporate notions of institutional goals and regulation configuration to define an *autonomic electronic institution* (AEI) as a tuple: $\langle PS, N, DF, G, P_i, P_e, P_a, V, \delta, \gamma \rangle$. Next, we only provide an intuitive idea about the elements of an AEI (further details can be found in [4]).

We assume that the main objective of an AEI is to accomplish its institutional goals (G). For this purpose, an AEI will adapt. We assume that the institution can observe the environment where agents interact (P_e), the institutional state of the agents participating in the institution (P_a), and its own state (P_i) to assess whether its goals are accomplished or not. Since an AEI has no access whatsoever to the inner state of any participating agent, only the *institutional (social) state* of an agent (P_a) can change. Therefore, each agent (a_i) can be fully characterized by his institutional state $P_{a_i} = \langle a_{i_1}, \ldots, a_{i_m} \rangle$ where $a_{i_j} \in \mathbb{R}$, $1 \leq j \leq m$ is an observable value of agent a_i. Taking the traffic as an example of an AEI, the speed of a car could be an example of an observable value of an agent; the number of lanes could be an example of an observable value of the environment; and the number of polices the institution uses to control the cars could be an example of an observable value of the state of the institution.

Formally, we define the goals of an AEI as a finite set of constraints $G = \{c_1, ..., c_p\}$ where each c_i is defined as an expression $g_i(V) \lhd [m_i, M_i]$ where $m_i, M_i \in \mathbb{R}$, $\lhd$ stands for either $\in$ or $\notin$. Additionally, g_i is a function over the reference values $V = \langle v_1, \ldots, v_q \rangle$, where each v_j results from applying a function h_j upon the agents' properties, the environmental properties and/or the institutional properties; $v_j = h_j(P_a, P_e, P_i),\ \ 1 \leq j \leq q$. In this manner, each goal is a constraint upon the reference values where each pair m_i and M_i defines an interval associated to the constraint. Continuing with the traffic example, an example of an institutional goal could be to minimize the number of accidents. Thus, the institution achieves its goals if all $g_i(V)$ values satisfy their corresponding constraints of belonging (at least to a certain degree) to their associated intervals. This is measured by means of a satisfaction function that computes the goal satisfaction degree (see [4] for further details).

The AEI definition includes the mechanisms to support the adaptation with the *normative transition function* (δ), and with the *PS transition function* (γ). An AEI employs norms to constrain agents' behaviors and to assess the consequences of their actions within the scope of the institution. We focus on norms describing prohibitions parametrically. So that each norm $N_i \in N$, $i = 1, \ldots, n$, has a set of parameters $\langle p^N_{i,1}, \ldots, p^N_{i,m_i} \rangle \in \mathbb{R}^{m_i}$. In fact, this parameters correspond to the variables in the *norm transition function* that will allow the institution to adapt. Continuing with the same traffic example, an example of a norm could be to stop always before to enter in an intersection and it norm can be parametrized by an associated fine applied if a car does not fulfill it. Notice that our AEI can not learn new norms, it only can adapt its norms by changing their parameters. On the other hand, adapting a PS involves the definition of a set of parameters whose values will be changed by the PS transition function. We define each scene in the performative structure, $S_i \in PS$, $i = 1, \ldots, t$, as having a set of parameters $\langle p^R_{i,1}, ..., p^R_{i,q_i} \rangle \in \mathbb{N}^{q_i}$ where $p^R_{i,j}$ stands for the number of agents playing role r_j in scene S_i. Thus, changing the values of these parameters means changing the performative structure.

The AEI definition includes the mechanisms to support the adaptation with the *normative transition function* (δ), and with the *PS transition function* (γ). We propose to use learning methods to learnt the *normative transition function* (δ), and the *PS transition function* (γ). Next section details the learning model used to adapt the AEI by changing those parameters.

3 Learning Model

Our aim is that at run-time an AEI could adapt its regulations to any population. We propose to learn the *norm transition function* (δ) and the *PS transition function* (γ) in two different steps in an overall learning process. In previous work [4] we have approached the first learning step, which corresponds to learn the best parameters for a set of predefined populations. In this work we focus on the second learning step: how to adapt the parameters to any population. As shown in Figure 1, in an initial step our AEI learns by simulation the best parameters for a collection of different agent populations. For each population of agents (A), the algorithm explores the space of parameter values ($I_1, .., I_k$) in search for the ones that lead the AEI to best accomplish its goals (G) for this population of

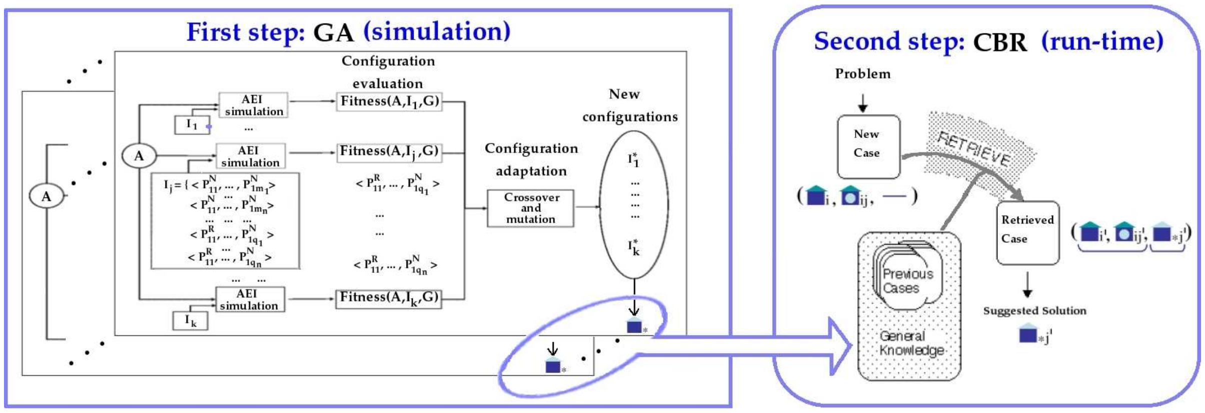


Fig. 1. Learning Model in two steps.

agents. Afterwards, we propose to use a Case-Based Reasoning (CBR) approach as a second step because it allows the AEI to solve situations that have been learned previously. We assume that agent populations that behave in similar way caused similar situations that may require similar solutions. Thus, at a second step an AEI identifies, in run-time, those situations for which its goals are not accomplished and uses CBR to retrieve a solution (regulation parameters) from the most similar situation in the knowledge base.

3.1 Applying CBR

Case Based Reasoning (CBR) [5] is based on learning from experience. The idea is to search in the experience (memory) of the system for similar situations, called cases, and using the corresponding solution to solve the current problem. In general, a new problem in a CBR system is solved by retrieving similar cases, reusing the case solution, revising the reused solution, and retaining the new experience. In this work we focus our attention in the first step of the CBR cycle, namely the retrieve process. Nevertheless, before addressing it, it is necessary to choose a representation for cases.

Case Definition The representation of cases is central to any CBR system. Cases must be represented based on the knowledge of the problem domain in order to choose the main features that better describe the case and thus that better help the processes involved in the CBR cycle. As to AEIs, we differentiate the following main features to be considered to represent cases:

- **AEI parameters' values.** They represent the parameters' values of some institution, namely the norm parameters' values and the performative structure parameters' values that an AEI uses for regulating agents.
- **Runtime behaviour.** They represent the global behaviour of the institution at runtime for some agent population when the institution uses the *AEI parameters' values.*

– **Best AEI parameters' values.** They represent the learned parameters' values of the institution for the previous agent population. In other words: the solution. Thus, they correspond to the parameters that the institution must apply in order to accomplish its institutional goals given both previous AEI parameters' values and runtime behaviour.

More precisely, regarding AEIs, we propose the definition of a case as a tuple (N^p,PS^p,V,pop,N^{p*},PS^{p*}), where:

– (N^p,PS^p) stands for the AEI parameters' values:
  - N^p stands for the current norm parameters' values;
  - PS^p stands for the current performative structure parameters' values;
– (V,pop) stands for the runtime behaviour:
  - V stands for the current set of reference values;
  - pop stands for statistic data that characterises the behaviour of the agents' population at runtime[1];
– (N^{p*},PS^{p*}) stands for the best AEI parameters' values:
  - N^{p*}: represents the best values for the norm parameters given the current norm parameters values (N^p) and the runtime behaviour (V,pop); and
  - PS^{p*}: represents the best values for the performative structure parameters given the current performative structure parameters values (PS^p) and the runtime behaviour (V,pop).

Thus, a case represents how an AEI (using N^p as norm values and PS^p as performative structure values) regulating a population of agents (showing the runtime behaviour described by pop and V) should change its regulations (to the N^{p*} and the PS^{p*} values). Notice that each case is an entry of the *normative transition function* (δ) and the *PS transition function* (γ). That is, the set of all cases approximate both transition functions.

Similarity function In order to compare two cases we must define an appropriate similarity function based on our representation of cases. We use aggregated distance function to compute the degree of similarity between a new case C^i and a case C^j in the case base:

$$S(C^i, C^j) = w_1 \cdot s\_AEI(C^i, C^j) + w_2 \cdot s\_V(C^i, C^j) + w_3 \cdot s\_pop(C^i, C^j) \quad (1)$$

where s_AEI corresponds to the distance of the AEI parameters' values (N^p, PS^p), s_V and s_pop correspond to the distance of the runtime behaviour (V,pop), and $w_1, w_2, w_3 \leq 0$ are weighting factors such that $w_1 + w_2 + w_3 = 1$. The s_AEI, s_V and s_pop distance functions are computed as the distance average of their attributes. To assess the distance between the values of an attribute we use:

$$sim(attr^i, attr^j) = \frac{|attr^i - attr^j|}{max(attr) - min(attr)} \quad (2)$$

[1] Notice that this data corresponds to reference values.

where $min(attr)$ and $max(attr)$ correspond to the limits of the interval of values of the attribute considered in the domain.

The Retrieval process In order to retrieve the most similar case to the problem case C^i without comparing all cases in the case base, we propose to perform this process in two steps:

1. Compare the AEI parameters' values, (N^p,PS^p), of the problem case C^i with the collection of all the AEI parameters' values in the case base using s_AEI and select the set of AEI parameters' values that best match.
2. Access the set of examples in the case base with these AEI parameters' values. Afterwards, we compare case C^i with these examples and select the case that best matches it based on distance function S.[2]

We use the first step with the idea that the most similar case must have similar AEI values because the runtime behaviour depends a lot of the AEI parameters' values. In fact, this is our hypothesis since we want to change the AEI parameters' values to change in some way the population behaviour and thus modify the runtime behaviour in order to achieve the institutional goals. The first step makes easy and fast the access to the most similar cases because we concentrate on only comparing the cases with similar AEI parameters' values. Thus, we do not need to compare all the cases of the case base. Moreover, we only need to compute once the distance function s_AEI for all cases with the same values of AEI parameters' values.

4 Case Study: Traffic Control

In order to test our model, we have considered and implemented the Traffic Regulation Authority as an Autonomic Electronic Institution, and cars moving along the road network as external agents interacting inside a traffic scene. Getting into more detail, we focus on a two-road junction where no traffic signals are considered. Therefore, cars must only coordinate by following the traffic norms imposed by the AEI. Our case study considers the performative structure to be a single traffic scene with two agent roles: one institutional role played by police agents; and one external role played by car agents.

We assume institutional agents to be in charge of detecting norm violations so that we will refer to them as police agents. The performative structure is parametrized by the number of agents playing the police role. Each police agent is able to detect only a portion of the total number of norm violations that car agents actually do. Norms within this normative environment are related to actions performed by cars. We consider two priority norms: the 'right hand-side priority norm', that prevents a car reaching the junction to move forward or to turn left whenever there is another car on its right; and the 'front priority

[2] Notice that we use a distance function as similarity function where low values imply high similarity.

norm', that applies when two cars reaching the junction are located on opposite lines, and one of them intends to turn left. Additionally, norms are parametrized by the associated penalties that are imposed to those cars refusing or failing to follow them. Cars do have a limited amount of points so that norm offenses cause points reduction. The institution forbids external agents to drive without points in their accounts.

In this work we focus on homogeneous populations where all agents in the population share the same behaviour. We propose to model each population based on three parameters (henceforth referred to as agent norm compliance parameters): $\langle fulfill\_prob, \mathit{high\_punishment}, \mathit{inc\_prob} \rangle$; where $fulfill\_prob \in [0, 1]$ stands for the probability of complying with norms that is initially assigned to each agent; $\mathit{high\_punishment} \in \mathbb{N}$ stands for the fine threshold that causes an agent to consider a fine to be high enough to reconsider the norm compliance; and $\mathit{inc\_prob} \in [0, 1]$ stands for the probability increment that is added to $fulfill\_prob$ when the fine norm is greater than the fine threshold (high_punishment). Car agents decide whether to comply with a norm based on their norm compliance parameters along with the percentage (between 0 and 1) of police agents that the traffic authority has deployed on the traffic environment. To summarise, agents decide whether they keep on moving –regardless of violating norms– or they stop –in order to comply with norms– based on a probability that is computed as:

$$\mathit{prob} = \begin{cases} \mathit{police} \cdot fulfill\_prob & fine \leq \mathit{high\_punishment} \\ \mathit{police} \cdot (fulfill\_prob + \mathit{inc\_prob}) & fine > \mathit{high\_punishment} \end{cases} \tag{3}$$

The institution can observe the external agents' institutional properties (P_a) along time. Considering our road junction case study, we identity different reference values, $V = \langle \mathit{col}, off, \mathit{crash}, \mathit{block}, \mathit{expel}, \mathit{police} \rangle$ where *col* indicates total number of collisions for the last t_w ticks ($0 \leq t_w \leq t_{now}$), off indicates the total number of offenses accumulated by all agents for the last t_w ticks, *crash* counts the number of cars involved in accidents for the last t_w ticks, *block* describes how many cars have been blocked by other cars for the last t_w ticks, *expel* indicates the number of cars that have been expelled out of the environment due to running out of points for the last t_w ticks, and finally, *police* indicates the percentage of police agents that the institution deploys in order to control the traffic environment.

The institution tries to accomplish its institutional goals by specifying the penalties of both priority norms and by specifying how many police agents should be deployed in the traffic scene. In this work we focus on four institutional goals: (i) minimize the number of collisions; (ii) minimize the number of offenses; (iii) minimize the number of expelled cars; (iv) and minimize the percentage of police agents to deploy to control the traffic environment. Notice, though, that these offences do not refer to offences detected by police agents but to the real offences that have been actually carried out by car agents.

5 Empirical Evaluation

As a proof of concept of our proposal in section 3, we extend the experimental setting for the traffic case study employed in [4]. The environment is modeled as a 2-lane road junction and populated with 10 homogeneous cars (endowed with 40 points each). Cars correspond to external agents without learning skills. They just move based on their random trajectories and the probability of complying with a norm (based on the function defined in (3)). During each discrete simulation, the institution replaces those cars running out of points by new cars, so that the cars' population is kept constant.

The four institutional goals, related to the *col*, off, *expel* and *police* reference values, are combined in a weighted addition, with weights 0.4, 0.4, 0.1 and 0.1 respectively. Thus, the first two goals are considered to be more important. The goal satisfaction is measured by combining the degree of satisfaction of these four institutional goals.

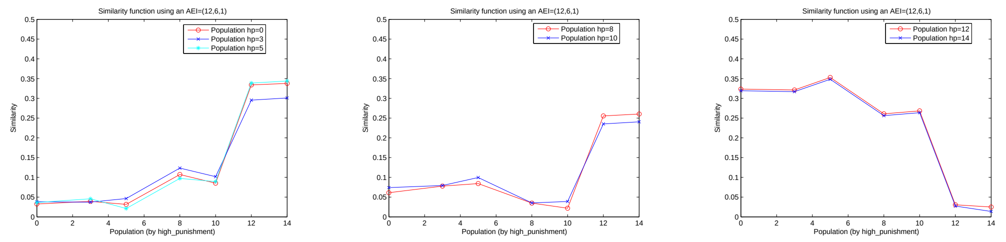
Table 1. Agent populations employed to generate the case base.

Populations	Pop. 1	Pop. 2	Pop. 3	Pop. 4	Pop. 5	Pop. 6	Pop. 7
$fulfill\_prob$	0.5	0.5	0.5	0.5	0.5	0.5	0.5
$high\_punishment$	0	3	5	8	10	12	14
inc_prob	0.4	0.4	0.4	0.4	0.4	0.4	0.4
$fine^*_{right}$	2	5	8	11	13	14	15
$fine^*_{front}$	1	4	6	9	12	13	15
$police^*$	1	1	1	1	1	1	1

As mentioned in section 3, (during training period) an AEI generates an initial base of cases from simulations of a set of prototypical populations. Following the tuple case definition introduced in section 3.1, $(N^p, PS^p, V, pop, N^{p*}, PS^{p*})$, we define a case C^i in this scenario as follows:

- $N^p = (fine_{right}, fine_{front})$ are the values of both norms' parameters;
- $PS^p = (police)$ is the value of the performative structure parameter;
- $V = (col, crash, off, block, expel)$ are the reference values;
- pop = $(mean\_off, median\_off, mean\_frequency\_off, median\_frequency\_off)$ contains the mean number of offenses, the median number of offenses, the mean of the frequency of offenses, and the median of the frequency of offenses carried out by agents for the last t_w ticks $(0 \leq t_w \leq t_{now})$;
- $N^{p*} = (fine^*_{right}, fine^*_{front})$ are the best values for both norms' parameters;
- $PS^{p*} = (police^*)$ is the best value for the parameter of the performative structure.

Table 1 shows the seven populations we have considered to generate the case base. They are characterized by their norm compliance parameters, being $fulfill\_prob = 0.5$ and $inc\_prob = 0.4$ for all of them, whereas $high\_punishment$ varies



(a) Three populations fulfill two norms with probability 0.9.

(b) Two populations fulfill the right norm with probability 0.9 and the front norm with probability 0.5.

(c) Two populations fulfill two norms with probability 0.5.

Fig. 2. Distance between populations when the AEI uses the same parameters values ($fine_{right} = 12, fine_{front} = 6$ and $police = 1$).

from 0 to 14. The $fine^*_{right}$, $fine^*_{front}$ and $police^*$ values in Table 1 are taken to be the best AEI parameters' values (N^{p*}, PS^{p*}).

5.1 Similarity function

We use the aggregated distance function defined in (1) to compute the degree of similarity between two cases. We have set the weights as follows: $w_1 = 0.1$, $w_2 = 0.5$, and $w_3 = 0.4$. Regarding the attributes of the AEI parameters' values, the $fine_{front}$ and $fine_{right}$ values are in the interval $[0, 15]$, and the *police* values are in the interval $[0, 1]$. However, the attributes of the runtime behaviour have not known limited values. We have established limits based on the values of the initial generated cases. Thus, we have established that the *col* values are in the interval $[0, 300]$, $crash \in [0, 400]$, $off \in [0, 500]$, $block \in [0, 200]$, $expel \in [0, 900]$, $mean\_off \in [0, 30]$, $median\_off \in [0, 30]$, $mean\_frequency\_off \in [0, 2]$, and $median\_frequency\_off \in [0, 2]$. Since the values of these attributes can be out of the proposed interval, we force distance to be 1 when $|attr^i - attr^j| > max(attr) - min(attr)$.

First of all, we have tested whether the distance function and the weights selected are suitable for the traffic domain. For this purpose, we have generated a little case base of only seven cases by simulating each population in Table 1. In order to create this case base, all seven populations have been run with the same AEI parameters: $fine_{right} = 12$, $fine_{front} = 6$ and $police = 1$. Afterwards, in order to test the distance function, we have created seven new cases simulating another time each population in Table 1 using the very same AEI parameters' values and have compared each one with the seven cases in the case base. Notice that two simulations of the same population using the very same AEI parameters' values do not create the very same case, because the runtime behaviour in both simulations may be similar but not exactly the same.

Figure 2 shows the results of testing similarities for the seven new cases with the seven ones in the base case. These seven new cases could be grouped by the population behaviour regarding the norm compliance. Since population of first three cases have an *high_punishment* lower than both norms' fines, cars fulfill both norms (with probability 0.9). However, populations with *high_punishment* 8 and 10 fulfill the right norm with probability 0.9 and the front norm with probability 0.5. Whereas, populations with *high_punishment* 12 and 14 fulfill both norms with probability 0.5. Figure 2 shows three charts corresponding to cases grouping by this behaviour. Thus, chart 2(a) shows the distance for the three first cases whose cars fulfill both norms with probability 0.9. We can see how these three cases are similar when compared with the seven cases in the case base, and also that the distance among them is less than with respect to other cases. Chart 2(b) shows the distance for cases using populations with *high_punishment* 8 and 10 whose cars fulfill the right norm with probability 0.9 and the front norm with probability 0.5. Chart 2(c) shows distance for cases using populations with *high_punishment* 12 and 14 whose cars fulfill both norms with probability 0.5. In the three charts we can see how distances are similar among cases created with populations that have similar behaviour. This figure also shows that if two different populations regulated by the very same norms behave in very similar manner, an AEI cannot differentiate them. This effect is because the AEI can only observe the external behaviour of populations. In any case, these results allow us to conclude that the proposed distance function is suitable. Next step is to test at run-time the proposed CBR approach.

5.2 Case Base

With the aim that at run-time the AEI could adapt its regulations to any population, we create a case base using populations in Table 1 and the corresponding best AEI parameters' values. In order to create the case base we have considered as AEI parameters' values $fine_{right} \in \{0, 3, 6, 9, 12, 15\}$, $fine_{front} \in \{0, 3, 6, 9, 12, 15\}$, and $police \in \{0.8, 0.9, 1\}$. Overall we have considered 108 different AEI parameters' values, as the result of combining $fine_{right}$, $fine_{front}$, and *police* values. To create cases for our case base, we have simulated each population in Table 1 with all 108 AEI parameters' values, so we have generated a total of 756 cases for the seven agent populations. To create each case, we have simulated the traffic model during 2000 ticks. Once finished the simulation, we generate a case by saving the AEI parameters' values (N^p, PS^p) used in this simulation, the runtime behaviour for the 2000 ticks (V, pop), and the best AEI parameters' values (N^{p*}, PS^{p*}) corresponding to the population used in this simulation.

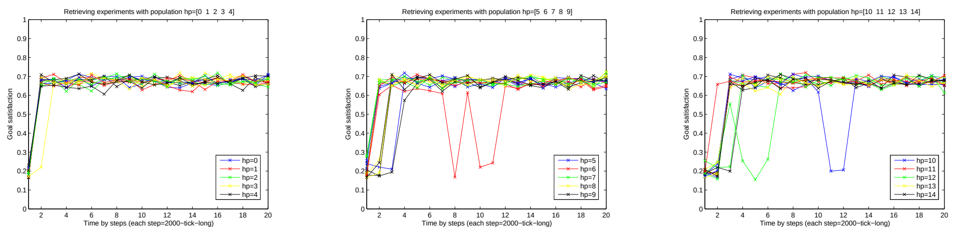
5.3 Retrieving

We have designed an experiment to test the retrieval process and therefore our approach. That is, we want to test if at run-time the AEI is able to self-configure its parameters for different agent populations by using the proposed CBR approach. Since we are testing our approach and we are not interested in efficiency issues, we employ the traffic simulator to recreate a run-time execution. We launch simulations of 2000 ticks during 20 times, namely steps (overall 40000 ticks). At each step, once the simulation finishes, we check the goal satisfaction degree and change the AEI parameters' values using the CBR approach when required. Although this allows us to change the population of agents at any step we have run the experiments using the same population in 20 simulations. For all experiments, the AEI starts with (0,0,0.8) parameters, that correspond to no fine for both norms and a deployment of 80% of police agents. Thus, we expect that the AEI starts with a low goal satisfaction degree (caused by the parameters it is using) and it will be able to retrieve a similar case with whose parameters that do increase the goal satisfaction degree.

At each step, we launch a simulation with a certain population of agents and when the simulation finishes, the AEI decides, based on the goal satisfaction, if it has to retrieve a case or not. If the goal satisfaction is greater than a threshold the AEI continues with the same parameters for a new simulation in the next step. Otherwise (when the goal satisfaction is lower than the threshold) we launch the CBR engine to retrieve a case of the case base (see section 5.2) in order to adapt the AEI parameters, namely to adapt the institution, its regulation. The threshold is computed as a desired goal satisfaction value G^* minus an epsilon value ϵ. In our experiments, we have set $\epsilon = 0.03$ and $G^* = 0.65$, which corresponds to the minimum of the best goal satisfaction degrees for our populations. The problem case is generated from the AEI parameters' values used in the last simulation and the runtime behaviour in the last 2000 ticks. The CBR system retrieves the most similar case and uses the best AEI parameters' values of the retrieved case for next simulation. Thus, the goal satisfaction degree can be computed again to check if it is necessary to define a new problem case.

We have used fifteen different populations to test our approach. Each population is characterized by their norm compliance parameters, being $fulfill\_prob = 0.5$ and $inc\_prob = 0.4$ for all of them, whereas $high\_punishment$ varies from 0 to 14. Notice that seven of them are the ones used for generating cases[3] (when $high\_punishment \in \{0, 3, 5, 8, 10, 12, 14\}$) whereas the AEI has no prior cases about of the other eight populations (when $high\_punishment \in \{1, 2, 4, 6, 7, 9, 11, 13\}$). Figure 3 shows the results for fifteen populations, where each chart shows five populations. Each population is run three times. Thus, overall we have performed 45 experiments. For each experiment, the figure shows the goal satisfaction every 2000 ticks during 20 steps. On chart 3(a) we can see that at initial step the goal satisfaction is low (around 0.2) and how the AEI quickly rises it up and maintains it constant during the rest of steps (between 0.6 and 0.7). On chart 3(a) we can see how the goal satisfaction degree starts at 0.2 and quickly rises up to $0.6 - 0.7$ with the initial case retrievals. This effect repeats on charts 3(b) and 3(c) on figure 3. That is, the AEI is able to adapt quickly its parameters in all experiments. However, we observe that for some populations (when $high\_punishment$ is 6, 10 and 12) the goal satisfaction does not remain constant. In particular, the goal satisfaction for one of the populations with $high\_punishment = 6$ goes down three times (steps 8, 10 and 11) to values

[3] Notice that use the same population does not mean use the same case because the runtime behaviour may be similar in both cases but not exactly the same.



(a) (b) (c)

Fig. 3. Goal satisfaction for fifteen populations. (a) Populations with $high\_punishment \in \{0, 1, 2, 3, 4\}$; (b) Populations with $high\_punishment \in \{5, 6, 7, 8, 9\}$; and (c) Populations with $high\_punishment \in \{10, 11, 12, 13, 14\}$.

close to 0.2. These oscillations happen because given a population regulated by the very same AEI parameters' values there is a variability on the behaviour in different simulations, that causes a variability in goal satisfaction. Thus, it sometimes occurs that because of this variability the goal satisfaction drops below the threshold and causes to restart the retrieval process. After this, the AEI stabilizes quickly again the goal satisfaction degree.

In order to estimate the error caused by these oscillations we have computed the percentage of simulations with a goal satisfaction greather than the threshold (0.62). At first step all experiments have a goal satisfaction less than the threshold. At second step a 52% of experiments (23 of 45) have a goal satisfaction greather than it. The percentage goes up to 89% (40 of 45) at third step and to 95% to the fourth. That is, in our experiments, the AEI needs four simulations to adapt itself in a correct manner to a 95% of new cases. At the rest of simulations (from simulation 5 to simulation 20) the average of the percentage of experiments with a goal satisfaction greather than the threshold is around 98%. That is, there is an error arround the 2% caused by the oscillations. In any case, we can conclude that the AEI is able to adapt to the populations, that is with the initial cases retrievals the AEI is able to adapt its parameters to accomplish its goals for each population.

6 Discussion and Future work

Within the area of Multi-Agent Systems, adaptation has been usually envisioned as an agent capability where agents learn how to reorganise themselves. Along this direction, in [6] Gasser and Ishida present a general distributed problem-solving model which can reorganize its architecture; and Horling et al. [7] propose an approach where the members adapt their own organizational structures at runtime. The fact that adaptation is carried out by the agents composing the MAS is the most significant difference with the approach presented in this paper. On the other hand, it has been long stated [8] that agents working in a common society need norms to avoid and solve conflicts, make agreements, reduce complexity, or to achieve a social order. Most research in this area consider norm configuration at design time [9] instead of at run-time as proposed in this paper. Regarding the traffic domain, MAS has been previously applied to it. For example, Camurri et al. [10] propose two field-based mechanisms to control cars and traffic-lights in order to manage to avoid deadlocks and congestion. Additionally, Case-Based Reasoning has been applied before in multi-agent systems where agents use different CBR approaches to individual learning and to cooperative learning for distributed systems [11, 12].

This paper presents a Case-Base Reasoning approach as an extension of previous work which allows an AEI to self-configure its regulations. We have presented the initial step towards a Case-Based Reasoning system, centering our work on the retrieval and usage processes. We have proppposed a case description and the distance function to be used by a generic AEI. We have tested the retrieval process of our approach in the traffic AEI case study, where the AEI learns two traffic norms and the number of institutional agents in order to adapt the norms and the performative structure to dynamical changes of agent populations.

Preliminary results in this paper are promising but they show some oscillations of the goal satisfaction degrees for some populations. Although, the computed error is low (around 2%), currently we are tuning the function used to compute the goal satisfaction and the threshold value in order to reduce the error and do it less sensitive to the variability. Once solved this, we plan to continue our experiments on the retrieval process by changing the populations between simulations. We also plan to continue on finishing the learning by focusing our work in the other CBR processes. As future work, and since this basically represents a centralized scenario, we plan to develop a more complex traffic network, allowing us to propose a decentralized approach where different areas (i.e., junctions) are regulated by a distributed institution.

Acknowledgments. This work was partially funded by the Spanish Education and Science Ministry as part of the IEA (TIN2006-15662-C02-01) and the 2006-5-0I-099 projects. The first author enjoys an FPI grant (BES-2004-4335) from the Spanish Education and Science Ministry.

References

1. Luck, M., McBurney, P., Shehory, O., Willmott, S.: Agentlink Roadmap. Agenlink.org (2005)
2. Kephart, J.O., Chess, D.M.: The vision of autonomic computing. IEEE Computer **36**(1) (2003) 41–50
3. Esteva, M.: Electronic Institutions: from specification to development. IIIA PhD Monography. Vol. 19 (2003)
4. Bou, E., López-Sánchez, M., Rodríguez-Aguilar, J.A.: Towards self-configuration in autonomic electronic institutions. In: COIN 2006 Workshops. Number LNAI 4386, Springer (2007) 220–235
5. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI Commun. **7**(1) (1994) 39–59

6. Gasser, L., Ishida, T.: A dynamic organizational architecture for adaptive problem solving. In: Proc. of AAAI-91, Anaheim, CA (1991) 185–190
7. Horling, B., Benyo, B., Lesser, V.: Using Self-Diagnosis to Adapt Organizational Structures. Proceedings of the 5th International Conference on Autonomous Agents (2001) 529–536
8. Conte, R., Falcone, R., Sartor, G.: Agents and norms: How to fill the gap? Artificial Intelligence and Law (7) (1999) 1–15
9. Fitoussi, D., Tennenholtz, M.: Choosing social laws for multi-agent systems: Minimality and simplicity. Artificial Intelligence **119**(1-2) (2000) 61–101
10. Camurri, M., Mamei, M., Zambonelli, F.: Urban traffic control with co-fields. In: Proc. of E4MAS Workshop at AAMAS 2006. (2006) 11–25
11. Plaza, E., Ontañón, S.: Cooperative multiagent learning. Adaptative Agents and Multi-Agent Systems **LNAI 2636** (2003) 1–17
12. Ros, R., Veloso, M.: Executing Multi-Robot Cases through a Single Coordinator. In: Proc. of Autonomous Agents and Multiagent Systems. (2007) 1264–1266

A Normative Multi-Agent Systems Approach to the Use of Conviviality for Digital Cities

Patrice Caire

University of Luxembourg, Computer Science Department
L-1359, Luxembourg, 6, Rue Richard Coudenhove-Kalergi, Luxembourg

Abstract. Conviviality is a mechanism to reinforce social cohesion and a tool to reduce mis-coordination between individuals, groups and institutions in web communities, for example in digital cities. We use a two-fold definition of conviviality as a condition for social interactions and an instrument for the internal regulation of social systems. In this paper we discuss the use of normative multi-agent systems to analyze the use of conviviality for digital cities, by contrasting norms for conviviality with legal and institutional norms in digital cities. We show the role of the distinction among various kinds of norms, the explicit representation of norms, the violability of norms and the dynamics of norms in the context of conviviality for digital cities.

Keywords. Conviviality, multi-agent systems, normative systems, social computing, digital cities.

1 Introduction

The role of norms for conviviality is a condition for social interactions and an instrument for the internal regulation of social systems [1]. For example, in digital cities "government regulations extend laws with specific guidance to corporate and public actions" [2].

In this paper we raise the following question: how can normative multi-agent systems be used to model conviviality for digital cities? We approach this question focusing on conviviality in digital cities, and by contrasting the use of normative multi-agent systems for conviviality with legal and institutional norms in digital cities.

Our main question breaks down into the following research questions: What are digital cities, what are normative multi-agent systems, what is conviviality and finally, can normative multi-agent systems be applied to conviviality for digital cities?

The layout of this paper follows these sub-questions. In section 2 we give a brief overview on digital cities, in section 3 we explain norms in regards to the legal and institutional aspects of digital cities, in section 4 we present a literature survey on the notion of conviviality and in section 5 we examine the use of norms for conviviality.

2 Brief Overview of Digital Cities

In their simplest form, digital cities are web portals using physical cities as a metaphor for information spaces. Depending upon their goals, they combine social, political and economic activities. Following are three examples showing their diversity: The ecity of Luxembourg that provides to citizens and visitors, information over the real city of Luxembourg as well as online forms and services, while eLuxembourg provides similar facilities at country level and eEurope at the European level. These types of digital cities are also called eAdministration and eGovernments; MSN CitySearch and AOL Digital Cities that offer services, shopping, entertainment and more generally, local easy to find and search information, are also referred to as eCommerce portals. Finally, Second Life and the Habbo Hotel are virtual worlds that provide infrastructures to users, primarily to conduct social experiences through role playing while, at the same time, attracting advertisers and businesses by the size of their *massive multi-player communities*.

Observing that "Digital cities commonly provide both profit and non-profit services and have a dilemma in balancing the two different types of services", Ishida [3] raises the question whether public digital cities can compete with commercial ones. "Without profit services, digital cities become unattractive and fail to become a portal to the city. Without nonprofit services, the city may become too homogeneous like AOL digital cities as a result of pursuing economic efficiency."

2.1 The Goals of Digital Cities

Commercial digital cities as websites started as local portals run by private companies, such as phone, web and airline companies, competing with each other. Nowadays, global companies such as Yahoo! and AOL offer city guides with services: Shopping, entertainment, local information and maps. Their business goals are geared toward vertical markets and their revenues are generated by advertising. Their general trend is to provide information, easy to find and search for, good maintenance of systems and frequent updates. They are effective in Asia, where they complement government agencies, but limited in scope by their top-down controlled and selected content, lack of two-way interaction with users and main advertising purpose.

Public digital cities started in the US with American community networks, inspired by a tradition of community-centered, grass-roots engagements emphasizing freedom of speech and activism. Their original goal was to create virtual information spaces, such as the WELL, *Whole Earth'Lectronic Link* and Blacksburg Electronic Village. US public digital cities main challenges are: Lack of synergy between community networks, private companies and administrations and competition between profit and non-profit organizations. Today they align with eGovernments.

In Europe, public digital cities evolved through the European Community leadership. Goals are to share ideas and technologies between all cities to strengthen

European partnerships, use information and communication technologies to resolve social, economic and regional development issues and improve the quality of social services. Main challenge, shown by the relatively slow commercialization of services and information, is the difficulty to integrate grass-roots communities and commercial points of view.

2.2 The Organizations of Digital Cities

Commercial digital cities count on accumulating urban information; They are well maintained, use proprietary software and rely on search engines, ranking interest links by sponsors, for business opportunities. Early on, commercial digital cities recognized the importance of usability and have done well to make their services usable by many.

Public digital cities look toward open systems. The lack of funds and the complexity of their partnerships caused many downfalls (Digital Amsterdam). Public digital cities rely on high speed networks tightly coupled with physical cities (Helsinki) and platforms for community networks (Bologna). They have multilayer architectures: Information, interface and interaction layers (Digital Kyoto).

Asian digital cities, called *city informatization*, emerged as government initiatives. Their goal is to develop their country through technological innovation. There were attempts to integrate grass-roots activities and university driven projects in 1999 with Digital Kyoto and Shanghai but the greatest challenge still remains their top-down approach based on administration activity.

2.3 Summary

Commercial and public digital cities were originally very different but seem today to have more overlapps.

However, as yet, no one model has been identified. In the US for-profit businesses and non-profit organizations co-exist and compete, in EU the attempts are to coordinate administrations, companies and citizens while Asia pursues government directed growth. Governments'goals for digital cities consist in helping close geographic and social digital divides, with access everywhere and for all, in accelerating economic development, and making the governments of cities more efficient and accessible. Pluralism and participation are combined with multi-disciplinary approaches, synergy between administrations, companies and citizens and, most importantly, a shared vision between all stakeholders.

The success factors of digital cities consist in achieving participation of institutions and communities, in balancing top-down direction, needed for technical infrastructure, and grass-roots initiatives, necessary to insure citizens' cohesion and in finding an equilibrium between economic and civic motivations. Ultimately, digital cities need to deal with the same complexity as real cities to attract and retain usage, and to function as entities that augment their physical counterparts.

3 Legal and Institutional Norms in Digital Cities

In their introduction to normative multi-agent systems, Boella et al. give the following definition: “A normative multi-agent system is a multi-agent system together with normative systems in which agents on the one hand can decide whether to follow the explicitly represented norms, and on the other the normative systems specify how and in which extent the agents can modify the norms” [4]. We first discuss the distinction among various kinds of norms, and then we discuss three issues in this definition, illustrated by examples in digital cities.

3.1 The Different Kinds of norms

Several kinds of norms are usually distinguished in normative systems. Within the structure of normative multi-agent systems [5] distinguish “between regulative norms that describe obligations, prohibitions and permissions, and constitutive norms that regulate the creation of institutional facts as well as the modification of the normative system itself”. A third kind of norms, procedural norms, can also be distinguished “procedural norms have long been considered a major component of political systems, particularly democratic systems” states Lawrence who further defines procedural norms as “rules governing the way in which political decisions are made; they are not concerned with the content of any decision except one which alters decision-making procedures” [6].

Constitutive norms: Boella et al. note several aspects to constitutive norms, one is an intermediate concept exemplified by “X counts as a presiding official in a wedding ceremony”, “this bit of paper counts as a five euro bill” and “this piece of land counts as somebodys private property” [7]. As per Searle, “the institutions of marriage, money, and promising are like the institutions of baseball and chess in that they are systems of such constitutive rules or conventions” [8]. In digital cities, an example of constitutive norm is voting in the sense that going through the procedure counts as a vote.

Boella et al further believe that “the role of constitutive rules is not limited to the creation of an activity and the construction of new abstract categories. Constitutive norms specify both the behavior of a system and the evolution of the system” [5]. The dynamics of normative systems is here emphasized as in *norm revision*, certain actions count as adding new norms for instance amendments: “The normative system must specify how the normative system itself can be changed by introducing new regulative norms and new institutional categories, and specify by whom the changes can be done” [5]. Today “US government agencies are required to invite public comment on proposed rules” [2]. Citizens are therefore encouraged to propose their changes through the digital cities interface. All revisions are traced and searchable.

Two other aspects of constitutive norms are organizational and structural, that is, how roles define power and responsibilities and how hierarchies structure groups and individuals. “Not only new norms are introduced by the agents playing a legislative role, but also that ordinary agents create new obligations, prohibitions and permissions concerning specific agents” [5].

Regulative Norms: "Legal systems are often modeled using regulative norms, like obligations and permissions. However, a large part of the legal code does not contain prohibitions and permissions, but definitions for classifying the common sense world under legal categories, like contract, money, property, marriage. Regulative norms can refer to this legal classification of reality" [7]. A regulative norm expressed as obligation is for example that, to access the administration documents on the Luxembourg digital city website, citizens must use the file format PDF rather than Postscript. Regulative norms also express permission, rights and powers, for example computer systems access rights and voting rights: You are allowed to vote in Luxembourg if you are resident for more than 5 years or were born in Luxembourg. "Regulative norms are not categorical, but conditional: they specify all their applicability conditions" [5]. In NYC, for instance, to renew online your Driver's License the stipulation is: "You cannot change your address during this transaction. You must have a completed form MV-619 (Eye Test Report) for this transaction. Read the requirements before you begin this transaction" [9].

Procedural norms: Lawrence distinguishes two kinds of procedural norms, objective and subjective. "Objective procedural norms are rules which describe how decisions are actually made in a political system [. . .] they specify who actually makes decisions, who can try to influence decision makers, what political resources are legitimate and how resources may be used. Subjective procedural norms, on the other hand, are attitudes about the way in which decisions should be made" [6]. Procedural norms are instrumental for individuals working in a system, for example, back office procedures and processes in digital city administrations.

3.2 Explicit versus Implicit Representation of Norms

The first property of norms in the definition of normative multi-agent systems is that norms are explicitly represented; explicite meaning formalized and verbalized by some authorities, implicite meaning tacitely agreed upon, not specialized nor codified. Often, norms are given as requirements of computer systems but only implicitly represented, for example, a form in which you would be asked to state whether or not you keep a pet at home without mentioning to you the purpose of the information: if your answer is affirmative, either you could be requested to pay a license fee or the amount of the fee could directly be deducted from your bank account. An example of explicit representation of a norm is given by Paris digital city website with the stipulation that to create online library accounts, one must be over 18 years old, otherwise an authorization of the parents is required.

Implicit representations are opaque to users and prevent governments to fulfill the democratic promise that transparency and explicit representations deliver. As users' need for explanation and understanding of rules and regulations grows, representations have to become more explicit and personalized to their expectations. Similarly, governments' interest also reside in the explicit representation of norms that can be addressed through the development of mechanisms for knowledge representation and reasoning.

Current efforts are somewhat in-between implicit and explicit representation with tools for text representation and retrieval with more advanced ontologies, semantic links and search capabilities. To this effect, the US government launched in 2006 a business portal to help small businesses comply with Federal regulations, a need that was not being met by any other Federal government program [9].

3.3 The Violation of Norms

The second property in the definition of normative multiagent systems, that norms can be violated, is also seen as a condition for the use of deontic logic in computer science: "Importantly, the norms allow for the possibility that actual behavior may at times deviate from the ideal, i.e. that violations of obligations, or of agents rights, may occur" [10].

If norms cannot be violated then the norms are *regimented*. For example if, in access control, a service can only be accessed with a certificate, then this norm can be implemented in the system by ensuring that the service can only be accessed when the certificate is presented. Regimented norms correspond to preventative control, in the sense that norm violations are prevented. When norm violations are possible there is only detective control, in the sense that behavior must be monitored, and norm violations have to be detected and sanctioned. "Social order requires social control, *an incessant local (micro) activity of its units*, aimed at restoring the regularities prescribed by norms. Thus, the agents attribute to the normative system, besides goals, also the ability to autonomously enforce the conformity of the agents to the norms, because a dynamic social order requires a continuous activity for ensuring that the normative systems goals are achieved. To achieve the normative goal the normative system forms the subgoals to consider as a violation the behavior not conform to it and to sanction violations" [7].

Norms can be violated because they are soft constraints. In digital cities, disincentives are often the mechanism used to prevent users from infringing the norms. For example, the digital city of Issy clearly stipulates that malicious intruders into the digital city will be prosecuted. When norm violations are possible, there are normative multiagent systems in which the violations can trigger new obligations, the so-called contrary-to-duty obligations. With contrary-to-duty obligations, there is not only a distinction between ideal and bad behavior, but there is also a distinction between various degrees of sub-ideal behaviors.

3.4 Summary

In many electronic institutions the norms are fixed and cannot be changed within the system, even though in many organizations there are roles defined within the system. The question is whether digital cities are a collection of electronic institutions, whether manipulations and changes are allowed within the system.

The US Regulations' office may be contributing to bring answers to this questions as it now provides on its site *Regulations.gov* a national forum for users to comment on existing and pending federal rules, therefore encouraging a more dynamic process for the modification and expliciteness of their rules and regulations.

4 The Role of Conviviality

Looking at some definitions shows that the meaning of conviviality depends on the context of use (table 1): In sociology, conviviality typically describes a relation between individuals and emphasizes positive values such as equality and community life, while in technology, it refers to being easy to use.

Table 1. Definitions of conviviality

Etymological and Domain Specific Definitions
15th century ”convivial”, from latin, convivere ”to live together with, to eat together with”. (French Academy Dictionary)
Adj. Convivial: (of an atmosphere, society, relations or event) friendly and lively, (of a person) cheerfully sociable. (English Oxford Dictionary)
Technology: Quality pertaining to a software or hardware easy and pleasant to use and understand even for a beginner. User friendly, Usability. By extension also reliable and efficient. (Grand Dictionnaire Terminologique)
Sociology: Set of positive relations between the people and the groups that form a society, with an emphasis on community life and equality rather than hierarchical functions. (Grand Dictionnaire Terminologique)

A less common view of conviviality emerges when it becomes an instrument to exercise power and enforce one point of view over another [11]. Conviviality is then experienced as a negative force by the loosing side. We summarized from different sources, positive and negative roles of conviviality and present some excerpts as examples (table 2): The emphasis for positive sides is on sharing common grounds and on inclusiveness, whereas for negative sides, the emphasis is on coercive behaviors and division.

4.1 From Individuals to Groups

First used in a scientific and philosophical context [12], in 1964, as synonymous with *empathy*, conviviality allows individuals to identify with each other thereby experiencing each other's feelings, thoughts and attitudes. By extension, a community is convivial when it aims at sharing knowledge: Members trust each other, share commitments and interests and make mutual efforts to build conviviality and preserve it. A convivial learning experience is based on role swapping [13], teacher role alternating with learner role, emphasizing the concept of reciprocity

Table 2. The different roles of conviviality

Positive aspects *(Enabler)*	**Grey aspects** *(Ignorance)*	**Negative aspects** *(Threat)*
Share knowledge & skills	Ignore cultural diversity	Crush outsiders
Deal with conflict	Hide conflict	Fragmentation
Feeling of "togetherness"	Promote homogenization	Totalitarism
Equality	Political correctness	Reductionism
Trust	Non-transparent systematic controls	Deception

as key component and creating concepts such as learning webs, skill exchange networks and peer-matching communication, later expanded by Papert and the Constructionists with concepts such as *learning-by-making* [14].

But conviviality is also a social form of human interaction, [15] a way to reinforce group cohesion through the recognition of common values. "Thus the sharing of a certain kind of food and/or drink can be seen as a way to create and reinforce a societal group through a positive feeling of togetherness (being included in/or part of the group), on which the community's awareness of its identity is based." Physical experiences of conviviality are transformed into learning and knowledge sharing experiences: "To know is to understand in a certain manner that can be shared by others who form with you a community of understanding".

However, the instrumentalization of conviviality occurs when one group is favored at the expense of another, "truth realities about minorities are built from the perspective of the majority via template token instances in which conflict is highlighted and resolution is achieved through minority assimilation to majority norms [...] Conviviality is achieved for the majority, but only through a process by which non-conviviality is reinforced for the minority" [16].

4.2 From Groups to Institutions

Conviviality also means "individual freedom realized in personal interdependence" [17]; It is the foundation for a new society, one that gives its members the means, referred to as tools, for achieving their personal goals: "A convivial society would be the result of social arrangements that guarantee for each member the most ample and free access to the tools of the community and limit this freedom only in favor of another member's equal freedom". Conviviality is then an enhancement to social capital and seen as a condition for a civil society, one in which "communities are characterized by political equality, civic engagement, solidarity, trust, tolerance and strong associative life" [18]. Conviviality also describes both "institutional structures that facilitate social relations and technological processes that are easy to control and pleasurable to use" [19]. However, "Conviviality masks the power relationships and social structures that govern communities". The question is "whether it is possible for convivial institutions to exist, other than by simply creating another set of power relationships and social orders that, during the moment of involvement, appear to allow free rein to individual expression [. . .]. Community members may experience a sense of conviviality which is deceptive and which disappears as soon as the members return to the alienation of their fragmented lives" [11].

4.3 Summary

We summarize by first noting that conviviality is usually considered a positive concept but that a darker side emerges when it becomes the instrument of power relations. Then following our two-fold definition of conviviality as a condition for social interaction and an instrument for the internal regulation of social systems, we see the crucial uses for conviviality in digital cities as a mechanism to reinforce social cohesion and as a tool to reduce mis-coordinations between individuals.

5 The Use of Norms for Conviviality

Intelligent agents, with their artificial intelligence capabilities can assist users, act on their behalf, adapt and learn while performing non-repetitive tasks; with spontaneous interactions and innovative approaches based on dynamic notions such as conviviality, trust and behavior are required [20]. In this section we reconsider the issues discussed in the context of legal and institutional norms for digital cities, this time in the context of norms for conviviality.

5.1 The Different Kinds of Norms for Conviviality

Typically today, web communities use text-based multi-user synchronous and asynchronous conferencing capabilities such as web forums and chat rooms. It is considered bad practice to use offensive language in a public forum or a chat room; *Network etiquette* and sometimes FAQ outline dynamic set of guidelines to encourage behaviors conducive to pleasant, efficient and polite user interactions. The constitutive norm for the use of offensive language in a chat room would, in this example, be the definition of what constitutes offensive language for this particular chat room; a regulative norm, the fact that using offensive language is prohibited; and a procedural norm, the fact that if a member uses offensive language, then other members should not use the chat room to retaliate and send rebuffs.

5.2 Explicit vs. Implicit Representation of Conviviality

Norms for conviviality are social norms, and even though they can be communicated, they are typically not explicit. Explicit norms for conviviality often refer to cooperation among agents or between agents and humans. **Embodied Conversational Agents**, for example, are "autonomous agents with a human-like appearance and communicative skills [. . .] To be able to engage the user in a conversation and to maintain it, the agents ought to have capabilities such as to perceive and generate verbal and nonverbal behaviors, to show emotional states, to maintain social relationship" [21]. Conversational agents in [22] must be endowed with *conviviality*: an agent is convivial if it is rational and cooperative, conviviality being the essential and global characteristic of services that "emerges from the intelligence of the system and not from a set of local characteristics that vary depending upon the application context and the types of users". Consequently a list of criteria will by itself not suffice to express conviviality, additional critical factors are: the relations that bind the criteria together and the way these relations are perceived by individuals.

Intelligent tutoring systems provide further examples of intelligent agents that must *understand and express* the implicite and explicite social norms. [23] propose an eLearning recommendation system for student tutors, in which "convivial social relationships are based on mutual acceptance through interaction", e.g. on reciprocity, students helping each other. Looking at interpersonal factors, [24] propose *emotionally intelligent tutor agents* that try "to construct a model of the mental state of the student and is knowledgeable of the potential effects of tutoring acts on the mental state. These insights are used to determine the appropriate action sequence and the manner of executing the actions".

Reputation systems highlight the need for explicit social norms: Reputation is the "indispensable condition for the social conviviality in human societies" state [25], because it encourages transparent information as in their system, all agents' actions are instantaneously propagated throughout the system. Critical challenges raised by the development of such systems are ethical issues such as preserving students'privacy and securing information gathered to create social profiles and more generally, the need to develop guidelines to safeguard users. Research examples addressing the issues are socially translucent systems characterized by visibility, awareness and accountability [26], and study of place-based presence and trust evaluation [27]. These research examplify the challenges of formalizing implicite norms of conviviality with various degrees of expliciteness and most importantly, the difference between social norms and norms for conviviality.

5.3 The Violation of Conviviality

It is always possible to violate social norms and therefore conviviality. Ignoring cultural and social diversity is violating conviviality as it creates conviviality for a group at the expense of others. In digital cities, being ignored when asking advices to a city administrator represents a conviviality violation as it breaks the bilateral form expected from these communication acts to only allow for unilateral communication. The online Paris library assures members of a *kind and pleasant service* and proposes a free mediator service in case of difficulties dealing with city clerks, therefore providing a compensation mechanism.

5.4 Summary

By definition, conviviality is a regulative instrument for social systems; it reinforces the group's common values and encourages the auto-regulation of the group; Conviviality has a normative function. In table 3, we summarize the use of norms for conviviality by Comparing legal norms with social norms.

Table 3. Legal norms versus social norms

Type	**Legal Norms**	**Social Norms**
Kinds of norms	Consitutive, regulative, procedural	Consitutive, regulative, procedural: problematic
Norm representation	Usually explicit	Usually implicit
Norm violation	Not possible for preventive control systems	Always possible to violate
Norm modification	By regulators	Emerging

6 Conclusion

In this paper we contrast norms for conviviality with legal and institutional norms in digital cities. We consider the following issues. First, the kinds of norms typically distinguished in legal systems can be distinguished for norms of conviviality too. Second, norms for conviviality are often implicit, and we believe it is an important question when such norms should be made explicit. Third, the issue of violation of conviviality and ways to deal with it is of central concern in web communities like digital cities. Fourth, norms concerning conviviality should be able to change over time. Fifth, norms for conviviality can come from a wide variety of sources.

References

1. Caire, P.: A critical discussion on the use of the notion of conviviality for digital cities. In: Proceedings of Web Communities 2007. (2007) 193–200
2. Lau, G.T., Law, K.H., Wiederhold, G.: Analyzing government regulations using structural and domain information. IEEE Computer **38** (2005) 70–76
3. Ishida, T.: Understanding digital cities. In: Digital Cities. (2000) 7–17
4. Boella, G., van der Torre, L., Verhagen, H.: Introduction to normative multiagent systems. Computational & Mathematical Organization Theory **12** (2006) 71–79
5. Boella, G., van der Torre, L.W.N.: Regulative and constitutive norms in normative multiagent systems. In Dubois, D., Welty, C.A., Williams, M.A., eds.: Knowledge Representation, AAAI Press (2004) 255–266
6. Lawrence, D.G.: Procedural norms and tolerance: A reassessment. The American Political Science Review (1976)

7. Boella, G., van der Torre, L.W.N.: Constitutive norms in the design of normative multiagent systems. In Toni, F., Torroni, P., eds.: CLIMA VI. Volume 3900 of Lecture Notes in Computer Science., Springer (2005) 303–319
8. Searle, J.R.: Speech Acts: An Essay in the Philosophy of Language. Cambridge University Press (1970)
9. Caire, P.: A normative multi-agent systems approach to the use of conviviality for digital cities. In Boella, G., van der Torre, L., Verhagen, H., eds.: Normative Multi-agent Systems. Number 07122 in Dagstuhl Seminar Proceedings, Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Germany (2007)
10. Jones, A., Carmo, J. Handbook of Philosophical Logic. In: Deontic logic and contrary-to-duties. Kluwer Academic Publishers (2002) 265–344
11. Taylor, M.: Oh no it isn't: Audience participation and community identity. Trans, Internet journal for cultural sciences **1** (2004)
12. Polanyi, M.: Personal Knowledge : Towards a Post-Critical Philosophy. University Of Chicago Press (1974)
13. Illich, I.: Deschooling Society. Marion Boyars Publishers, Ltd. (1971)
14. Papert, S., Harel, I.: 1. In: Constructionism. Cambridge, MA: MIT Press. (1991)
15. Schechter, M.: Conviviality, gender and love stories: Plato's symposium and isak dinesen's babette's feast. Trans, Internet journal for cultural sciences **1** (2004)
16. Ashby, W.: Unmasking narrative: A semiotic perspective on the conviviality/non-conviviality dichotomy in storytelling about the german other. Trans, Internet journal for cultural sciences **1** (2004)
17. Illich, I.: Tools for Conviviality. Marion Boyars Publishers (1974)
18. Putnam, R.D.: Bowling alone: the collapse and revival of american community. In: Computer Supported Cooperative Work. (2000) 357
19. Lamizet, B.: Culture - commonness of the common? Trans, Internet journal for cultural sciences **1** (2004)
20. Caire, P.: Conviviality for ambient intelligence. In: Proceedings of Artificial Societies for Ambient Intelligence, Artificial Intelligence and Simulation of Behaviour (AISB'07). (2007) 14–19
21. Pelachaud, C.: Multimodal expressive embodied conversational agents. In Zhang, H., Chua, T.S., Steinmetz, R., Kankanhalli, M.S., Wilcox, L., eds.: ACM Multimedia, ACM (2005) 683–689
22. Sadek, M.D., Bretier, P., Panaget, E.: ARTIMIS: Natural dialogue meets rational agency. In: International Joint Conferences on Artificial Intelligence (2). (1997) 1030–1035
23. Gomes, E.R., Boff, E., Vicari, R.M.: Social, affective and pedagogical agents for the recommendation of student tutors. In: Proceedings of Intelligent Tutoring Systems. (2004)
24. Heylen, D., Nijholt, A., op den Akker, R., Vissers, M.: Socially intelligent tutor agents. In Rist, T., Aylett, R., Ballin, D., Rickel, J., eds.: IVA. Volume 2792 of Lecture Notes in Computer Science., Springer (2003) 341–347
25. Casare, S., Sichman, J.: Towards a functional ontology of reputation. In: AAMAS '05: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems, New York, NY, USA, ACM Press (2005) 505–511
26. Erickson, T., Kellogg, W.A.: Social translucence: an approach to designing systems that support social processes. ACM Trans. Comput.-Hum. Interact. **7** (2000) 59–83
27. ter Hofte, G.H., Mulder, I., Verwijs, C.: Close encounters of the virtual kind: a study on place-based presence. AI Soc. **20** (2006) 151–168

Embedding Landmarks and Scenes in a Computational Model of Institutions

Owen Cliffe, Marina De Vos, and Julian Padget

Department of Computer Science
University of Bath, BATH BA2 7AY, UK
{occ,mdv,jap}@cs.bath.ac.uk

Abstract. Over the last decade, institutions have demonstrated that they are a powerful mechanism to make agent interactions more effective, structured, coordinated and efficient. Different authors have tackled the problem of designing and verifying institutions from different angles. In this paper we propose a formalism that is capable of unifying and extending some of these approaches, as well as providing the necessary tools to assist in the design and verification processes. We demonstrate our approach with a non-trivial case-study.

1 Introduction

The concept of landmarks appears in [18] where they are used to identify a set of semantic properties relating to the state of a conversation, and which may furthermore be organized into sequences or patterns, while transition between landmarks is made by an appropriate sequence of one or more speech acts. A more detailed discussion follows in [12], where they are presented as propositions that are true in the state represented by the landmark (sic). The value of landmarks, and more specifically, their partial ordering into landmark patterns, is how they permit the identification of phases in a conversation protocol corresponding to the achievement of goals (and subgoals). Additionally, they form an integral part of realizing joint intention theory [7] as participants in a conversation interact with one another, *via* speech acts, to follow a common protocol and satisfy common goals. The utility of landmarks, from the electronic institution designer's perspective is their potential role in building a bridge [9, 1] between the rigidity of the protocols that feature in bottom-up design and the (relative) flexibility of norms that characterize top-down design.

The formal model put forward in [5] and its corresponding operationalization through Answer Set Programming (ASP) [3] aims to support the top-down design of electronic institutions through the provision of a domain-specific action language [17], called InstAL, tailored to the specification of institutions. Tools have been developed to translate InstAL into the SMODELS [14] syntax for processing by the answer set solver and furthermore the soundness and completeness of the institutional programs with respect to the formal model have been proven [4]. In this paper we explore the consequences of the correspondence between landmarks, as described in the literature, and the institutional states of our (executable) model, argue that the stronger logical framework of our formalism is advantageous and demonstrate the expressiveness of the Inst*AL* language through a non-trivial case-study.

2 The Institutional Framework

In this section we provide a brief description of our framework, starting with the formal model and following with the semantics. We then turn our attention to ASP as the underlying computational mechanism and the mapping from action language to ASP.

The Formal Model: Our model of an institution is a quintuple, $\mathcal{I} := \langle \mathcal{E}, \mathcal{F}, \mathcal{C}, \mathcal{G}, \Delta \rangle$, comprising three disjoint sets:

- events $\mathcal{E}$, which can be either institutional (generated within the institution) or exogenous (caused by events outside of the scope of the institution) . In particular, we define a subset of the exogenous events as *creation events*, $\mathcal{E}_+$, which contain events which account for the creation of an institution and a subset of the institutional events as *dissolution events*, $\mathcal{E}_\times$.
- fluents $\mathcal{F}$, being the four distinguished sets of fluents — powers $\mathcal{W}$, permissions $\mathcal{P}$, obligations $\mathcal{O}$, domain-specific fluents $\mathcal{D}$ — that constitute the state of the institution and hence the basis for reasoning about the institution.
- and an initial state Δ comprising the initial set of fluents in the institution

and two relations $\mathcal{C}$ and $\mathcal{G}$ over $\mathcal{X} \times \mathcal{E}$, where $\mathcal{X} = 2^{(\mathcal{F} \cup \neg \mathcal{F})}$ and $\phi \in \mathcal{X}$ represents a set of conditions which must be met in a given state in order for either relation to have an effect.

- $\mathcal{C}$ defines under which circumstances fluents are initiated and terminated.
- $\mathcal{G}$ implements the *count-as* operation and defines under which conditions in the institutional state the occurence of a given event will result in the generation of one or more new events.

Semantics: The semantics of this framework are defined by traces of exogenous events. Each trace induces a sequence of institutional states, called a model. Starting from the initial state, the first exogenous event will, using the $\mathcal{G}$, generate a set of events. Each of these events will possibly affect the next state by means of the $\mathcal{C}$ relation. The combined effect results in the next state of the model. This process continues until all exogenous events in the trace have taken place.

ASP: In *answer set programming* ([3]) a logic program is used to describe the requirements that must be fulfilled by the solutions of a certain problem. The answer sets of the program, usually defined through (a variant/extension of) the stable model semantics [10], then correspond to the solutions of the problem. The programs consist of a set clauses with negation-as-failure in the body. Assumptions are verified by eliminating negation from the program using the Gelfond-Lifschitz reduction and to check if this new positive program sustains the assumptions made. Tools for obtaining answers sets are called answer set solvers. For our system we use the SMODELS [14] solver.

The Mapping: The mapping of each actual institution $\mathcal{I}$ into an answer set program consists of two parts: (i) P_{base} which is identical for each institution and handles the occurrence of observed events, the semantics of obligations and rules to maintain the commonsense inertia of fluents , and (ii) $P^*_{\mathcal{I}}$ which is specific to the institution being modelled and represents the translation of it rules (norms and action semantics). Together they form the answer set program $P_{\mathcal{I}}$. In order to be able to use this program to reason about the institution, it is then combined with two other ASP programs: a trace program, containing a contraint on the length of traces of events being considered, and a query program expressing some constraint over the answer sets that shall be generated — the property or properties of the model that we wish to investigate.

InstAL : Our primary objective in this work is to be able to specify the behaviour of an institution in terms of its norms, and then to be able to test properties of the model of the institution thus defined. Consequently, we need a machine-processable representation. The engine for the verification is an answer set solver, so one approach would be to require the specification to be written in the input syntax for such a system, such as SMODELS, as outlined in [5]. However, while it may be useful for the designer to examine the code given to the answer set solver occasionally, it also necessarily contains low level support details that are less relevant to the task of institutional design. For this reason and because of the event-oriented nature of the specification, a domain-specific event language seems an appropriate medium, hence Inst*AL* .

We define the language Inst*AL* in order to simplify the process of specifying institutions. Individual institution specifications and multi-institution specifications are written as single Inst*AL* programs in a human-readable text format. These files can then be translated automatically into answer set programs that directly represent the semantics of the institutions specified in the original descriptions.

The language supports a simple set-based type system and syntax for the declaration of fluents, events, and institutions (bearing in mind the model also supports multi-institutional models as discussed in [6]). Normative fluents are pre-defined for power, permission and obligation. The designer may also specify static properties of an institution, that are initiated when the institution is created and never change. This provides a straightforward way to associate roles with institutions. Rather than give a formal syntax specification, for which there is not room here, we put forward and extended example in section 4 to illustrate the language features in a use-case. A detailed discussion of the Inst*AL* language can be found in [6, 4].

An Inst*AL* reasoning problem consists of the following:

1. One or more Inst*AL* institution descriptions each of which describes a single institution or a multi-institution.
2. A domain definition that grounds aspects of the descriptions. This provides the domains for types and any static properties referenced in the institution and multi-institution definitions.
3. A *trace program* which defines the set traces of exogenous events to investigate.
4. A *query program* which describes the desired property to validate with the Inst*AL* reasoning tool.

The reasoning process can be summarised as follows:

1. The Inst*AL* to ASP translator takes one or more single or multi-institution descriptions (in the Inst*AL* syntax described below), and domain definition files (described below) as input. Using these files, the translator generates a set of answer set programs which describe the semantics of the input institutions.

2. The translated institution programs along with a trace program and query program are then grounded by the LPARSE program (part of the SMODELS toolkit).
3. This grounded program description is then given as input to the SMODELS answer set solver. This produces zero or more answer sets. Each answer set corresponds to a possible model of the input institution for a given trace described by the trace program that matches the given query.
4. These answer sets may then be visualised and interpreted by the designer.

3 Landmarks and Scenes

As already discussed in the introduction, the essence of a landmark is a condition on a state in order for an action in some protocol to have effect. The relative sophistication of a landmark specification can be affected by the logic that is used to define the condition, but in many respects this is a technicality. For example [18] use first order logic augmented with modal operators for propositional attitudes and event sequences, [12] use dynamic propositional logic with modal operators from the previous work, while [9] (p.126) has atoms, implying the conjunction of positive values, within a Kripke model and [1] uses linear-time temporal logic. More important is the actual purpose of landmarks, as [12] states:

> Besides contributing to formal analyses of protocol families, the landmark-based representation facilitates techniques similar to partial order planning [13] for dynamically choosing the most appropriate action to use next in a conversation, allows compact handling of protocol exceptions, and in some cases, even allows short circuiting a protocol by opportunistically skipping some intermediate landmarks.

This highlights the relationship between agent actions and conventional AI planning and leads to the observation of the correspondence between landmarks and scenes (also mentioned in [9]). By scenes, we refer to the components of performative structure identified by Noriega [15] that are essentially sub-protocols of the larger institution or viewed bottom-up, an institution *may* be seen as the composition of numerous protocols that help agents achieve various sub-goals. What it is important to observe about Noriega's (and later in [16]) definition of the performative structure is how various conditions are imposed on the transitions from one scene to another, typically constraining the number and role of the agents that may move. A scene essentially encapsulates a self-contained protocol whose purpose is to achieve some sub-goal of the institution contributing to the objective of using the institution in the first place.

From this perspective, we can now turn to the relationship between our formalism and both landmarks and scenes, having established that both concepts serve to identify some (final) state in which a condition (capturing some institutional sub-goal) has been satisfied. Returning to the relations that drive our formalism (see section 2), the event generation function serves to create institutional facts, while the consequence relation focuses attention on the initiation and termination of fluents. The function is expressed as $\mathcal{C} : \mathcal{X} \times \mathcal{E} \rightarrow 2^{\mathcal{F}} \times 2^{\mathcal{F}}$. Where the first set in the range of the function describes which fluents are initiated by the given event and the second set represents those fluents terminated by the event. We use the notation $\mathcal{C}^{\uparrow}(\phi, e)$ to denote the fluents that are initiated by the event e in a state matching ϕ and the notation $\mathcal{C}^{\downarrow}(\phi, e)$ to denote those terminated by event e in a state matching ϕ.

From the description of event generation and the consequence relation, it can be seen that fluents are initiated and terminated in respect of an event and some conditions on the state of the institution. This corresponds exactly with the notion of landmark, in that an event takes the institution into a new state *but* this is predicated on the current state — that is, a condition. Thus landmarks arise naturally from our formalization and furthermore, the condition language would appear to be richer than in some earlier work because the condition may contain both positive and negative information, including the use of negation as failure and hence non-monotonic reasoning, since these are basic properties of answer set semantics. Our conclusion therefore is that our formalism provides landmarks for free and, thanks to ASP semantics, enriches the landmark description language over earlier examples.

In the literature cited above, landmarks appear to be restricted to speech acts, that is messages from participating agents. Our model goes further, as we also consider exogenous events that do not originate from participating agents or from institutional events. This makes our approach a convenient tool for reasoning with scenes, where the transition between the various scenes does not necessarily depend on agents' actions. Instead the transition markers could be linked to exogenous events which are taken into account when the institution reaches a certain state. At this point the consequence relation could be used to set the powers and permissions (and so the behaviour) of the participating agents. The Dutch auction protocol detailed in the next section uses this technique to distinguish between the various phases/scenes of the protocol.

4 The Dutch Auction Protocol

Informal Description of Dutch Auction: In this protocol a single agent is assigned to the role of auctioneer, and one or more agents play the role of bidders. The purpose of the protocol as a whole is either to determine a winning bidder and a valuation for a particular item on sale, or to establish that no bidders wish to purchase the item. The protocol is summarised as follows:

1. Round starts: The auctioneer selects a starting price for the item and informs each of the bidders present of this price. The auctioneer then waits for a given period of time for bidders to respond.
2. Upon receipt of the starting price, each bidder has the choice as to whether to send a message indicating their desire to bid on the item at that price, or to send no message indicating that they do not wish to bid on the item.
3. At the end of the prescribed period of time, if the auctioneer has received a single bid from a given agent, then the auctioneer is obliged to inform each of the participating agents that this agent has won the auction.
4. If no bids are received at the end of the prescribed period of time, the auctioneer must inform each of the participants that the item has not been sold.
5. If more than one bid was received then the auctioneer must inform each agent that a conflict has occurred.
6. In the case where the item is sold the protocol is finished.

7. In the case that no bids are received then the auctioneer may either start a new round of bidding at a lower price, or withdraw the item from sale.
8. In the case where a conflict occurs then the auctioneer must re-open the bidding at a higher price and start the round again in order to resolve the conflict.

We focus on the protocol for the round itself (items 1-6). In our description below we omit from the messages a definition of the item in question and the starting price. While the inclusion of these aspects in the protocol is possible, their inclusion does not change the structure of the protocol round so we leave them out for simplicity.

In the following paragraphs we go through the Inst*AL* code step by step. The full listing can be found in Figures 1 and 2. Each line of Inst*AL* code is labelled with DAR-FigureNr-LineNr for ease of reference.

The first lines indicate the name of the institution (DAR-1-1) and the types of agents, Bidder (DAR-1-2) and Auctioneer (DAR-1-3) that may participate in the institution. These types are used as placeholders in the Inst*AL* rules for the agents participating in a particular instance of the institution, then when instantiated all rules are grounded appropriately. The institution is created by one creation event `createdar` as specified by rule DAR-1-4.

Based on the protocol description above, the following agent messages are defined (DAR-1-8 – DAR-1-12): the auctioneer announces a price to a given bidder (`annprice`), the bidder bids on the current item (`annbid`), the auctioneer announces a conflict to a given bidder (`annconf`) and the auctioneer announces that the item is sold (`annsold`) or not sold (`annunsold`) respectively. Each exogenous action has a corresponding institutional event (DAR-1-16 – DAR-1-20 which accounts for a valid execution of the physical action performed. In all cases the two events are linked by an unconditional generates statement in the description (DAR-2-29,DAR-2-32,DAR-2-37,DAR-2-38,DAR-2-39).

In addition to the agent actions we also include a number of time-outs indicating the three external events (which are independent of agents' actions) that affect the protocol. For each time-out we define a corresponding institutional event suffixed by `dl` indicating a deadline in the protocol:

`priceto`, `pricedl`: A time-out indicating the deadline by which the auctioneer must have announced the initial price of the item on sale to all bidders. (DAR-1-5 and DAR-1-13)

`bidto`, `biddl`: A time-out indicating the expiration of the waiting period for the auctioneer to receive bids for the item (DAR-1-6 and DAR-1-14).

`desto`, `desdl`: A time-out indicating the deadline by which the auctioneer must have announced the decision about the auction to all bidders (DAR-1-7 and DAR-1-15).

We assume that the time-outs will occur in the order specified (that is, due to their durations it is impossible for this to be otherwise). We use the corresponding institution events in the protocol description and constrain the order in which they are empowered in the institution to ensure that while the exogenous events may occur in any order, the institution event may only occur once in each iteration and in the order specified (DAR-2-52 to DAR-2-59).

We define a single additional institution event `alerted(Bidder)` (DAR-1-23) that represents the event of a bidder being validly notified of the result of the auction. We

```
institution dutch;                                (DAR-1-1)

type Bidder;                                      (DAR-1-2)
type Auct;                                        (DAR-1-3)

create event createdar;                           (DAR-1-4)

exogenous event priceto;                          (DAR-1-5)
exogenous event bidto;                            (DAR-1-6)
exogenous event desto;                            (DAR-1-7)

exogenous event annprice(Auct,Bidder);            (DAR-1-8)
exogenous event annbid(Bidder,Auct);              (DAR-1-9)
exogenous event annconf(Auct,Bidder);             (DAR-1-10)
exogenous event annsold(Auct,Bidder);             (DAR-1-11)
exogenous event annunsold(Auct,Bidder);           (DAR-1-12)

inst event pricedl;                               (DAR-1-13)
inst event biddl;                                 (DAR-1-14)
inst event desdl;                                 (DAR-1-15)

inst event price(Auct,Bidder);                    (DAR-1-16)
inst event bid(Bidder,Auct);                      (DAR-1-17)
inst event conf(Auct,Bidder);                     (DAR-1-18)
inst event sold(Auct,Bidder);                     (DAR-1-19)
inst event unsold(Auct,Bidder);                   (DAR-1-20)

dest event badgov;                                (DAR-1-21)
dest event finished;                              (DAR-1-22)

inst event alerted(Bidder);                       (DAR-1-23)

fluent onlybidder(Bidder);                        (DAR-1-24)
fluent havebid;                                   (DAR-1-25)
fluent conflict;                                  (DAR-1-26)

initially pow(price(A,B)), perm(price(A,B)),
          perm(annprice(A,B)),
          perm(badgov),pow(badgov),
          perm(pricedl),pow(pricedl),
          perm(priceto),
          perm(biddl),
          perm(bidto),
          perm(desto);                            (DAR-1-27)
```

Fig. 1. InstAL for the Dutch Auction Round Institution Part 1

additionally specify a dissolution event `finished` (DAR-1-22) that indicates the end of the protocol.

For the sake of simplicity, we do not focus in detail on the effects of the auctioneer violating the protocol. Instead we define a dissolution institutional event `badgov` (DAR-1-21) that accounts for a*any* instances in which the auctioneer has violated the protocol. Once an auctioneer has violated the protocol, we choose to treat the remainder of the protocol as invalid and dissolve the institution.

```
initially obl(price(A,B),pricedl,badgov);                              (DAR-2-28)
annprice(A,B) generates price(A,B);                                    (DAR-2-29)
price(A,B) terminates pow(price(A,B));                                 (DAR-2-30)
price(A,B) initiates pow(bid(B,A)),perm(bid(B,A)),perm(annbid(B,A));   (DAR-2-31)

annbid(A,B) generates bid(A,B);                                        (DAR-2-32)
bid(B,A) terminates pow(bid(B,A)),perm(bid(B,A)),perm(annbid(B,A));    (DAR-2-33)
bid(B,A) initiates havebid,onlybidder(B) if not havebid;               (DAR-2-34)
bid(B,A) terminates onlybidder(_) if havebid;                          (DAR-2-35)
bid(B,A) initiates conflict if havebid;                                (DAR-2-36)

annsold(A,B) generates sold(A,B);                                      (DAR-2-37)
annunsold(A,B) generates unsold(A,B);                                  (DAR-2-38)
annconf(A,B) generates conf(A,B);                                      (DAR-2-39)
biddl terminates pow(bid(B,A));                                        (DAR-2-40)
biddl initiates pow(sold(A,B)),pow(unsold(A,B)),
        pow(conf(A,B)), pow(alerted(B)),perm(alerted(B));              (DAR-2-41)
biddl initiates perm(annunsold(A,B)),perm(unsold(A,B)),
        obl(unsold(A,B),desdl,badgov) if not havebid;                  (DAR-2-42)
biddl initiates perm(annsold(A,B)),perm(sold(A,B)),
         obl(sold(A,B), desdl, badgov) if havebid, not conflict;       (DAR-2-43)
biddl initiates perm(annconf(A,B)),perm(conf(A,B)),
         obl(conf(A,B), desdl, badgov) if havebid, conflict;           (DAR-2-44)
unsold(A,B) generates alerted(B);                                      (DAR-2-45)
sold(A,B) generates alerted(B);                                        (DAR-2-46)
conf(A,B) generates alerted(B);                                        (DAR-2-47)
alerted(B) terminates pow(unsold(A,B)), perm(unsold(A,B)),
 pow(sold(A,B)), pow(conf(A,B)), pow(alerted(B)),
 perm(sold(A,B)), perm(conf(A,B)), perm(alerted(B)),
 perm(annconf(A,B)),perm(annsold(A,B)),perm(annunsold(A,B));           (DAR-2-48)
desdl generates finished if not conflict;                              (DAR-2-49)
desdl terminates havebid,conflict,perm(annconf(A,B));                  (DAR-2-50)
desdl initiates pow(price(A,B)), perm(price(A,B)),
 perm(annprice(A,B)), perm(pricedl),pow(pricedl),
 obl(price(A,B),pricedl,badgov) if conflict;                           (DAR-2-51)
priceto generates pricedl;                                             (DAR-2-52)
pricedl terminates pow(pricedl);                                       (DAR-2-53)
pricedl initiates pow(biddl);                                          (DAR-2-54)

bidto generates biddl;                                                 (DAR-2-55)
biddl terminates pow(biddl);                                           (DAR-2-56)
biddl initiates pow(desdl);                                            (DAR-2-57)

desto generates desdl;                                                 (DAR-2-58)
desdl terminates pow(desdl);                                           (DAR-2-59)
```

Fig. 2. InstAL for the Dutch Auction Round Institution Part 2

Once the institution has been created, the auctioneer will receive power and permission to announce prices. We also provide empowerment and permission for the dissolution event `badgov`. Furthermore all deadlines are permitted but only pricing is empowered. This is specified by DAR-1-27.

The rules of the institution are driven by the occurrence of the time-outs described above and hence may be broken down in to three phases as follows:

1. In the first phase of the protocol the auctioneer must issue price statements to each of the bidders. We represent this in the protocol by defining an initial obligation on the auctioneer to issue a price to each bidder before the price deadline (DAR-2-28). Once this has taken place, the auctioneer is no longer permitted to issue a price (DAR-2-30).

   Once a price has been sent to the bidder, the bidder is empowered and permitted to bid in the round (note that we permit both the action of validly bidding itself, `bid(B,A)`, as well as the action of sending the message which may count as bidding, `annbid(B,A)` (DAR-2-31).
2. In the second phase of the protocol, bidders may choose to submit bids. These must be sent before the bid time-out event. In order to account for the final phase of the protocol, we must capture the cases when one bid, no bids or multiple bids (a conflict) occur. In addition, in a given round, we must also take into account that bids may be received asynchronously from different agents over a period of time. In order to capture which outcome of the protocol has occurred we use three domain fluents (DAR-1-24 – DAR-1-26) to record the state of the bidding: `onlybidder(Bidder), havebid, conflict`.

   The first of these fluents denotes the case where a single bid has been received and no others (and records the bidder which made this bid), the second fluent records cases where one or more bids have been received and the third records cases where more than one bid has been received.

   These fluents are determined in the second phase of the protocol using DAR-2-34, DAR-2-35 and DAR-2-36. The first rule accounts for the first bid that is received, and is only triggered if no previous bids have been made. The second rule accounts for any further bids and terminates the `onlybidder` fluent when a second bid is received. The final rule records a conflict if a bid is received and a previous bid has occurred.

   Once a bid has been submitted we do not wish to permit an agent to submit further bids, or for those further bids to be valid. In order to account for this we have line DAR-2-33.
3. In the third and final phase of the protocol the auctioneer must notify the bidding agents of the outcome of the auction. This phase is brought about by the occurrence of the `biddl` event which denotes the close of bidding. In order to account for this, we terminate each agents' capacity to bid further in the auction (DAR-2-40) and correspondingly initiate the auctioneer's power to bring about a resolution to the auction (DAR-2-41). To do so, we create an obligation upon the auctioneer to issue the right kind of response (`sold, unsold, conflict`) depending on outcome of the previous phase (`havebid,conflict`) before the next deadline (`desdl`) is announced. This is encoded by DAR-2-42 – refdar:obl2. For each outcome, the auctioneer is obliged and permitted to issue the appropriate response to every bidding agent before the decision deadline. If an auctioneer fails to issue the correct outcome to any agent before the final deadline then a violation will occur. The protocol follows these notifications using DAR-2-45 – DAR-2-46.

   Once an agent has been notified we wish to prohibit the auctioneer from notifying that agent again. We do this by introducing a rule which terminates the auctioneer's

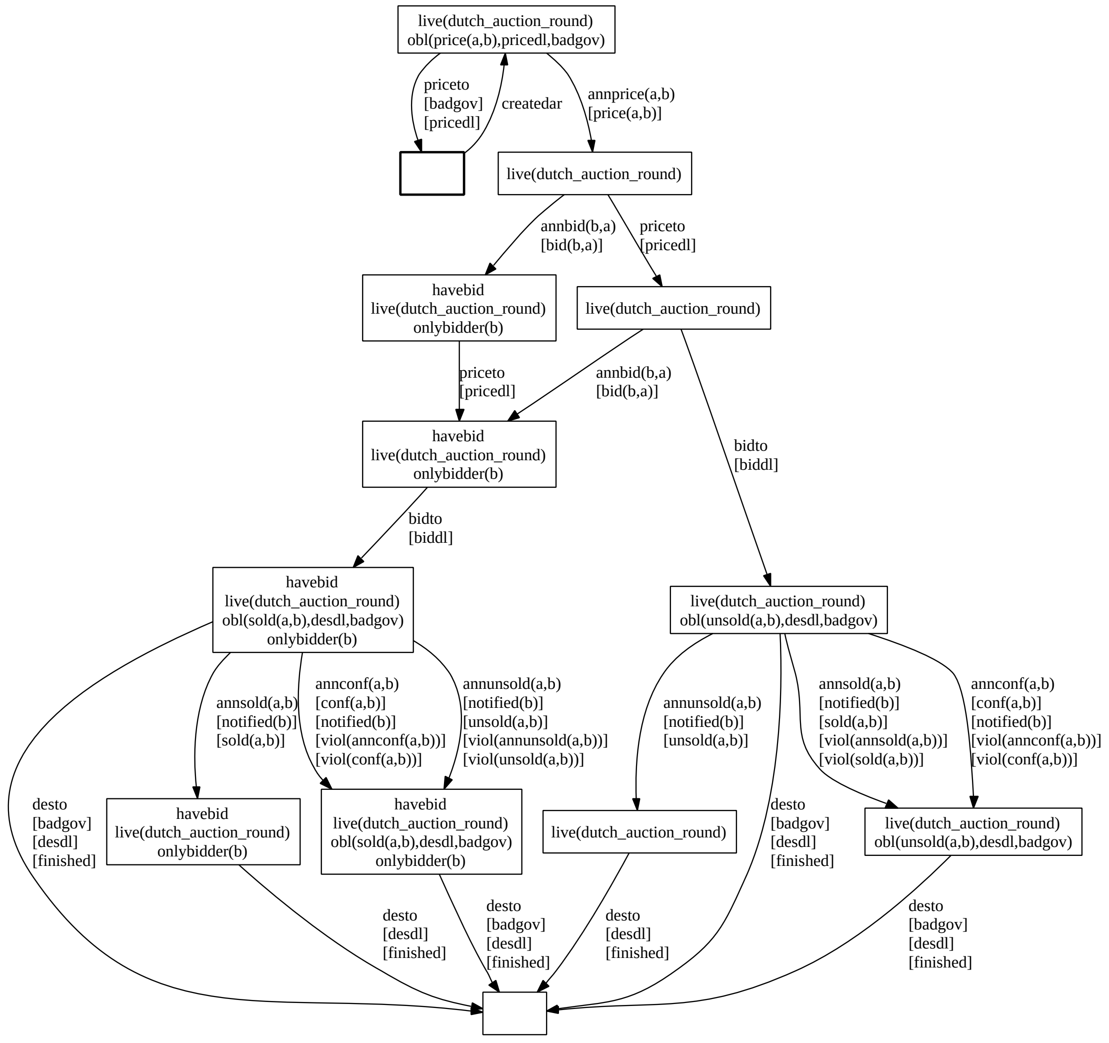


Fig. 3. States of the auction round for a single bidder

power and permission to issue more than one notification to any one agent (DAR-2-48).

Finally, when the deadline expires (the exogenous event `desto` triggers `desdl`) and either the protocol ends or the bidders have created a conflict. In the former case, DAR-2-49 ensures dissolution of the institution. In the conflict case, the auctioneer must re-open the bidding using a new round. We represent this by adding two lines. The first terminates the intermediate fluents which were used to represent the outcome of the protocol (`havebid` and `conflict`). This is established by DAR-2-50

. The second (DAR-2-51), initiates the obligation for the auctioneer to re-open the round by issuing a price to the bidders and all associated powers and permissions.

Verification: Once we have the Inst*AL* description of our institution, we can obtain an ASP program as described in Section 2. This program may then be combined with a trace program and query, allowing us to query properties and determine possible outcomes of this protocol.

The simplest type of verification procedure is to execute the program with no query. In this case all possible traces of the protocol will be provided as answer sets of the translated program.

Each answer set represents all possible sequences of states which may occur in the model and these may in turn be used to visualise all reachable states of the protocol (for a given number of agents). In order to execute the protocol we need to ground it with an auctioneer a and a bidder b. We could execute the translated program as is, however the answer sets of the program would include *all* traces of the protocol, including those containing actions which have no effect. Transitions of this kind may be of interest in some cases (we may be interested in the occurrence of associated violations for instance) however in this case we choose to omit them in order to reduce the number of answer sets to analyse. This can be achieved by specifying a query program which limits answer sets only to those containing traces in which a change of state occurs. For the technical details on this query program, see [4].

Solving the translated program with the associated query program yields a total of 60 answer sets corresponding to each possible trace where an effect occurs in each transition. By extracting the states from the answer set we may generate a graphical representation of the transition system which the protocol creates.

In order to include all possible states of the protocol we must select a large enough upper bound for the length of traces such that all possible states are reached. In general the selection of this upper bound depends on the program and query in question and it should be noted that the answer sets of the program represent *only* those solutions to the query which can be found in the given trace length.

In the case of the auction protocol examined here we had to establish this upper bound by the somewhat unsatisfactory process of iterating the solver process and determining the number states until no more states were found. For the example above, with only two agents, the longest traces which yield new states are of length 7, resulting in 33 answer sets.

Figure 3 illustrates all possible states for a single round of the protocol with one bidder (for a larger number of bidders, the state space will be considerably larger, growing exponentially with their number). Note that as there is only one bidder participating in the protocol conflicts cannot occur. For the sake of clarity we omit fluents relating to powers and permissions from the figure.

Further verification: In the above protocol we stated that when there was a conflict in the bidding for the protocol (that is, when two or more bidders issue valid bids) that the bidding should re-open. In order to ensure that this new round continues as before we must ensure that the institutional state at the beginning of a re-opened round is the same as the institutional state when the original round opened.

This property may be specified as a query program in our framework, as we now describe. In this case we are only interested in traces where a conflict has occurred. We specify this by adding the following constraints to the query program:

$$\begin{aligned} \texttt{hadconflict} &\leftarrow \texttt{holdsat}(\texttt{conflict}, \texttt{I}), \texttt{instant}(\texttt{I}). \\ \perp &\leftarrow \textbf{not}\ \texttt{hadconflict}. \end{aligned}$$

The first rule states that if there is any state where the `conflict` fluent occurs, then the literal `hadconflict` should be included in the answer set. The second rule states that we should not include any answer sets where the literal `hadconflict` is not included.

We are also only interested in traces where the protocol is re-started and bidding is re-opened. We add this constraint in a similar way, using two rules as follows:

$$\begin{aligned}\mathtt{restarted} &\leftarrow \mathtt{occurred(desdl, I)},\\ &\ \mathtt{holdsat(conflict, I), instant(I)}.\\ \bot &\leftarrow \textbf{not}\ \mathtt{restarted}.\end{aligned}$$

The first of these rules state that if the `desdl` event has occurred at any time we include the literal `restarted` in our answer set and the second rule states that we should only include answer sets where this literal is included.

In order to determine the fluents (if any) which differ between a state following the creation of the institution and a state following a protocol re-start, we mark these fluents using the literals `startstate(F)` indicating that fluent `F` is true in the start state of this trace, and `restartstate(F)` indicating that the fluent `F` was true in a state following a protocol re-start.

Literals of the form `startstate(F)` are defined using the following rule:

$$\begin{aligned}\mathtt{startstate(F)} \leftarrow\ &\mathtt{holdsat(F, I1)},\\ &\mathtt{occurred(createdar, I0)},\\ &\mathtt{next(I0, I1), ifluent(F)}.\end{aligned}$$

Which states that `F` is a fluent in the start state, if `F` holds at time instant `I1` and creation event `createdar` occurred at instant `I0` and that instant `I1` immediately follows instant `I0`.

We similarly define the fluents that hold in the re-start state with the rule:

$$\begin{aligned}\mathtt{restartstate(F)} \leftarrow\ &\mathtt{holdsat(F, I1), occurred(desdl, I0)},\\ &\mathtt{holdsat(conflict, I0), next(I0, I1), ifluent(F)}.\end{aligned}$$

which states that `F` holds in the restart state, if it held in the state `I1` which immediately followed the occurrence of the decision deadline `desdl` when a conflict held in that state.

We then define the following rules which indicate the differences between the start state and the re-start state:

$$\begin{aligned}\mathtt{missing(F)} &\leftarrow \mathtt{startstate(F), notrestartstate(F), ifluent(F)}.\\ \mathtt{added(F)} &\leftarrow \mathtt{restartstate(F), notstartstate(F), ifluent(F)}.\end{aligned}$$

These rules indicate that a fluent is present in the start state, but missing from the restart state (indicated by `missing(F)`), or missing in the start state, but present in the restart state (indicated by `added(F)`) respectively.

Finally we define the query constraint, in this case we are only interested in traces where a difference occurs between the start state and the restart state. We add these constraints using following rules:

$$\begin{aligned}\mathtt{invalid} &\leftarrow \mathtt{missing(F), ifluent(F)}.\\ \mathtt{invalid} &\leftarrow \mathtt{added(F), ifluent(F)}.\\ \bot &\leftarrow \textbf{not}\ \mathtt{invalid}.\end{aligned}$$

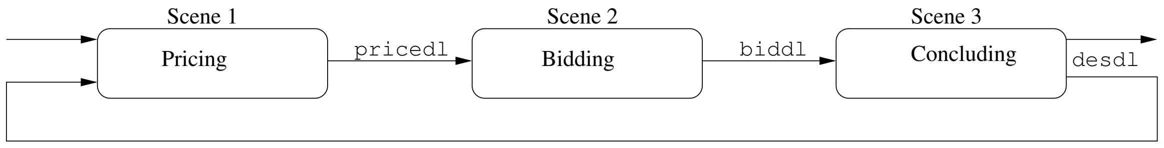


Fig. 4. Landmarks in the Dutch Auction round

The first two rules state that if a fluent `F` is either `missing` or `added`, then the literal `invalid` is true. The third rule constrains answer sets of the program to only those containing the literal `invalid`.

These rules, when combined with the translated program of the institution allow us to determine which fluents have changed between the start state and end state of the protocol.

Given the translated program and the query program described above, we obtain no answer sets for the protocol as defined, indicating that it is indeed the case that there are no fluents which differ in the state following a protocol restart and the state following the creation of the institution. This result is consistent with the original description of the protocol and will permit subsequent rounds following a conflict to continue in the same way as the original round. The same query holds true for auctions including three or four bidders.

The Scene Perspective: Although the Inst*AL* language does not explicitly allow for the definition of scenes (i.e. no special constructs are available), it is straightforward to achieve this with the available language constructs. The auction protocol discussed above, can be seen as composed of three scenes each marked by the occurrence of a deadline (except for the start of the protocol). Figure 4 provides the scene transition diagram. Each of these deadlines is the result of an exogenous event generated by the environment (e.g. DAR-2-52). The occurrence of such a deadline, changes the empowerment and permissions of the agents involved in the protocol (e.g. DAR-2-50). Rules are provided to assure the correct transition through the scenes (e.g. DAR-2-53).

5 Conclusions

Due to page restrictions, we are unable to include a separate overview of related work. Instead we have added pointers to related work wherever possible throughout the presentation. An extensive discussion of the relation between our framework and other normative systems, such as [2, 8, 11, 17, 19] to name but a few, can be found in [5, 4].

In this article we have demonstrated that the formal system described in [5] can easily deal with non-trivial institutions. Furthermore, we have shown that our characterisation can deal directly with landmarks and scenes, thus linking it more clearly with earlier work on institutional specification.

References

[1] H. Aldewereld. *Autonomy vs. Conformity: an Institutional Perspective on Norms and Protocols.* PhD thesis, Utrecht, 2007.

[2] Alexander Artikis. *Executable Specification of Open Norm-Governed Computational Systems*. PhD thesis, Department of Electrical & Electronic Engineering, Imperial College London, Sept. 2003.

[3] C. Baral. *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge Press, 2003.

[4] O. Cliffe. *Specifying and Analysing Institutions in Multi-Agent Systems Using Answer Set Programming*. PhD thesis, Dept. Computer Science, University of Bath, June 2007.

[5] O. Cliffe, M. De Vos, and J. A. Padget. Answer set programming for representing and reasoning about virtual institutions. In K. Inoue, K. Satoh, and F. Toni, editors, *CLIMA VII*, volume 4371 of *Lecture Notes in Computer Science*, pages 60–79. Springer, 2006.

[6] O. Cliffe, M. De Vos, and J. A. Padget. Specifying and reasoning about multiple institutions. In J. Vazquez-Salceda and P. Noriega, editors, *COIN 2006*, volume 4386 of *Lecture Notes in Computer Science*, pages 63–81. Springer, 2007.

[7] P. R. Cohen and H. Levesque. Intention is choice with commitment. *Artificial Intelligence*, 42:213–261, 1990.

[8] M. Colombetti, N. Fornara, and M. Verdicchio. The role of institutions in multiagent systems. In *Proceedings of the Workshop on Knowledge based and reasoning agents, VIII Convegno AI*IA 2002, Siena, Italy*, 2002.

[9] V. Dignum. *A Model for Organizational Interaction*. PhD thesis, Utrecht, 2004.

[10] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In *Proc. of fifth logic programming symposium*, pages 1070–1080. MIT PRESS, 1988.

[11] L. Kamara, A. Artikis, B. Neville, and J. Pitt. Simulating computational societies. In P. Petta, R. Tolksdorf, and F. Zambonelli, editors, *Proceedings of workshop on engineering societies in the agents world (esaw)*, LNCS 2577, pages 53–67. Springer, 2003.

[12] S. Kumar, M. J. Huber, P. R. Cohen, and D. R. McGee. Toward a formalism for conversation protocols using joint intention theory. *Computational Intelligence*, 18(2):174–228, 2002. doi:10.1111/1467-8640.00187.

[13] S. Minton, J. Bresina, and M. Drummond. Total order and partial order planning: A comparative analysis. *Journal of Artificial Intelligence Research*, 2:227–262, 1994.

[14] I. Niemelä and P. Simons. Smodels: An implementation of the stable model and well-founded semantics for normal LP. In J. Dix, U. Furbach, and A. Nerode, editors, *Proceedings of the 4th International Conference on Logic Programing and Nonmonotonic Reasoning*, volume 1265 of *LNAI*, pages 420–429, Berlin, July 28–31 1997. Springer.

[15] Pablo Noriega. *Agent mediated auctions: The Fishmarket Metaphor*. PhD thesis, Universitat Autonoma de Barcelona, 1997.

[16] J. A. Rodríguez-Aguilar. *On the Design and Construction of Agent-mediated Institutions*. PhD thesis, Universitat Autonoma de Barcelona, 2001.

[17] M. Sergot. (C+)++: An Action Language For Representing Norms and Institutions. Technical report, Imperial College, London, Aug. 2004.

[18] I. Smith, P. Cohen, J. Bradshaw, M. Greaves, and H. Holmback. Designing conversation policies using joint intention theory. In *Proceedings of International Conference on Multi Agent Systems*, pages 269–276, 1998. doi:10.1109/ICMAS.1998.699064.

[19] P. Yolum and M. P. Singh. Flexible protocol specification and execution: applying event calculus planning using commitments. In *AAMAS '02: Proceedings of the first international joint conference on Autonomous agents and multiagent systems*, pages 527–534. ACM Press, 2002.

Semantical Concepts for a Formal Structural Dynamics of Situated Multiagent Systems

Antônio Carlos da Rocha Costa and Graçaliz Pereira Dimuro

Escola de Informática – PPGINF, Universidade Católica de Pelotas
96.010-000 Pelotas, RS, Brazil, {rocha,liz}@atlas.ucpel.tche.br

Abstract. This paper introduces semantical concepts to support a formal structural dynamics of situated multiagent systems. Multiagent systems are seen from the perspective of the Population-Organization model, a minimal semantical model where the performance of organizational roles by agents, and the realization of organizational links by social exchanges between agents, are the key mechanisms for the implementation of an organization structure by a population structure. The structural dynamics of a multiagent system may then be modelled as a set of transformations on the system's overall population-organization structure. We illustrate the proposed approach to structural dynamics by introducing a small set of operational rules for an exchange value-based dynamics of organizational links. The paper sets the stage for further work on structural dynamics where other structural elements, besides organizational links, are taken into account.

1 Introduction

PopOrg, a minimal population-organization based model, was introduced in [1] in order to support the study of the structural dynamics of multiagent systems (MAS). Both time-invariant and time-variant versions of the model were introduced, but no specific mechanism was presented to account for any possible structural dynamism.

In this paper, we improve the above mentioned work by refining that model with the notion that social interactions are exchanges performed between agents. Also, we present an exchange value-based mechanism able to account for some aspects of the structural dynamics of multiagent systems. We combine the two ideas to define a simple set of operational rules for an exchange value-based dynamics of organizational links.

The work sets the stage for further studies on the structural dynamics of multiagent systems by establishing the basis of a mechanism where further aspects of the structural dynamics of such systems can be considered, besides the dynamics of links.

We remark that the paper is based on a distinction between the notions of intensional and extensional descriptions of systems: intensional descriptions deal with subjective aspects pertaining to the internal functioning of the agents that operate in a system (like norms, values, etc.), while extensional descriptions deal with objective aspects pertaining to the external functioning of those agents (like actions performed, objects exchanged, etc.). The main concerns of the paper are, thus, an extensional description of the structural dynamics of multiagent systems organizations, and a possible way to articulate such extensional dynamics with the intensional aspect of the exchange values involved in the interactions between the agents that participate in the organizations.

On the other hand, we note that the process model that underlies the structural dynamics of the population-organizational model [1] is similar to the general signal-based denotational model that underlies some declarative languages devised to specify real-time reactive systems [2]. This encourages the view that the PopOrg model may suitably be construed as an adequate model for multiagent systems situated in environments presenting real-time constraints. In fact, it is only natural to expect that it is precisely in the case of situated multiagent systems that the issues of structural dynamics arise crucially (because of the pressures for the adaptation of the system to the variations in the environment – this point is further explored in Sect. 5, on related works).

The paper is organized as follows. In Sec. 2, we revisit the Population-Organization model, refining its notion of interaction through a general notion of social exchange. In Sec. 3, we summarize the particular exchange values approach to social interactions [3] that we adopt, reviewing its notion of exchange value and its model of social exchange. Section 4 illustrates the general purpose of the paper by joining the revisited Population-Organization model with the adopted system of social exchanges, allowing for a simple mechanism able to support a preliminary model of exchange value-based dynamics of organizational links. Section 5 concludes the paper by summarizing related work and exploring further aspects of the proposal.

A technical remark: we use the following coordinate-wise notation, when dealing with vectors (n-tuples) of sets (taking $expr_0 \Leftrightarrow expr_1 \wedge \ldots \wedge expr_n$) :

$$(X_1, \ldots, X_n) \subseteq (Y_1, \ldots, Y_n) \equiv_{def} X_i \subseteq Y_i, i = 1, \ldots n. \qquad (1)$$

$$\bigcup\{(X_1, \ldots, X_n) \mid expr_0\} \equiv_{def} (\cup\{X_1 \mid expr_1\}, \ldots, \cup\{X_n \mid expr_n\}) \qquad (2)$$

2 The Population-Organization Model

The Population-Organization model of multiagent systems, introduced in [1], emphasizes the modelling of systems composed of a small group of agents, adopting an interactionist point of view [3, 4]. In such model, the organizational structures of the system are implemented by the system's population of agents through two main mechanisms: the assignment of organizational roles to agents, and the realization of organizational links between roles by the social exchanges that are established between the agents that perform those roles. Of course, in such model, the central components of the structural dynamics of the systems are the operations of creation and deletion of elements like organizational roles, organizational links, agents and exchange processes.

2.1 The Time-Invariant Population-Organization Model

The time-invariant Population-Organization model, $PopOrg = (Pop, Org, imp)$, is construed as a pair of structures, the population structure Pop and the organization structure Org, together with an implementation relation imp.

The Time-Invariant Population Structure The *population* of a multiagent system consists of the set of agents that inhabit it. The *population structure* of a multiagent system is its population set together with the set of all behaviors that the agents are able to perform, and the set of all exchange processes that they can establish between them (for simplicity, we consider only pairwise exchanges).

Let T be a discrete sequence of time instants. The *population structure* of a time-invariant multiagent system is a tuple

$$Pop = (Ag, Act, Bh, Ep, bc, ec) \tag{3}$$

where:

- Ag is a finite non-empty set of agents, called the *population* of the system;
- Act is the finite set of all *actions* (communication actions and actions on concrete objects of the environment) that may be performed by the agents of the system;
- $Bh \subseteq [T \rightarrow \wp(Act)]$ is the set containing all possible agent behaviors, modeled as functions that specify, for each time $t \in T$, a set of actions $X \in \wp(Act)$ that an agent may perform at that time, each behavior determining a sequence of sets of actions available for the agents to perform in the system;
- $Ep \subseteq [T \rightarrow \wp(Act) \times \wp(Act)]$ is the set containing all possible *exchange processes* that two agents may perform in the system, each process given by a function that specifies, for each $t \in T$, a pair of set of actions $(X_1, X_2) \in \wp(Act) \times \wp(Act)$, determining a sequence of exchanges available for any two agents to perform, by executing together or interleaving appropriately their corresponding actions;
- $bc : Ag \rightarrow \wp(Bh)$ is the *behavioral capability* function, such that for each agent $a \in Ag$, the set of all behaviors that a is able to perform in the system is $bc(a)$;
- $ec : Ag \times Ag \rightarrow \wp(Ep)$ is the *exchange capability* function, such that for each pair of agents $a_1, a_2 \in Ag$, the set of all exchange processes that a_1 and a_2 may perform between them is $ec(a_1, a_2)$;
- $\forall a_1, a_2 \in Ag\ \forall e \in ec(a_1, a_2)\ \forall t \in T$:
  $$Prj_1(e(t)) \subseteq \bigcup\{b(t) \mid b \in bc(a_1)\}\ \wedge\ Prj_2(e(t)) \subseteq \bigcup\{b(t) \mid b \in bc(a_2)\},$$
  where Prj_1, Prj_2 are projection functions, so that the agents' exchange capabilities are constrained by their joint behavioral capabilities.

Given $t \in T$ and $a \in Ag$, we note that $bc(a)(t) = \{act \mid act \in b(t), b \in bc(a)\}$ is the set of all possible actions that agent a may perform at time t, given its behavioral capability $bc(a)$. We also note that, in general, the exchange capability $ec(a_1, a_2)$ of a pair of agents $a_1, a_2 \in Ag$ should be deducible from their respective behavioral capabilities $bc(a_1)$ and $bc(a_2)$, and from any kind of restriction that may limit their set of possible exchanges (e.g., social norms, inherited habits, etc.), but since we are presenting an extensional model where such intensional, subjective restrictions take no part, it is sensible to include ec explicitly in the description of the population structure.

By the same token, the behavioral capability $bc(a)$ of an agent $a \in Ag$ should be deducible from any *internal description* of a where its set of behaviors is constructively defined, but since we are taking an external (observational) point of view of the agents, we include bc explicitly in the model.

Finally, we note that the definition of Pop is given in time-invariant terms. However, in general, any of the sets Ag, Act, Bh, Ep of the population structure, and both the behavioral and exchange capabilities, bc and ec, are time-variant (see Sect. 2.2).

The Time-Invariant Organization Structure The *time-invariant organization structure* of a time-invariant population structure $Pop = (Ag, Act, Bh, Ep, bc, ec)$ is a structure $Org = (Ro, Li, lc)$, where

– $Ro \subseteq \wp(Bh)$ is the set of *roles* existing in the organization, a role being given by a set of behaviors that an agent playing the role may have to perform;
– $Li \subseteq Ro \times Ro \times Ep$ is the set of *links* that exist in the organization between pairs of roles, each link specifying an exchange process that the agents performing the linked roles may have to perform;
– $lc : Ro \times Ro \to \wp(Li)$ is the link capability of the pairs of roles, that is, the set of links that the pairs of roles may establish between them;
– $\forall l \in Li\ \exists r_1, r_2 \in Ro : l \in lc(r_1, r_2)$, that is, every link has to be in the link capability of the two roles that it links.

Clearly, the PopOrg model adopts a process-based view of organizations.

The Time-Invariant Implementation Relation Population and organization structures are formally defined in a quite independent way. A population structure induces no more than a loose restriction on the set of organization structures that may be imposed on it: the behavioral capability function bc constrains the set of possible roles that an agent may have in any possible organization and, indirectly, the set of possible exchange processes in which it may participate, thus, also the set of possible organizational links that it may have with any other agent in that system.

The fact that a given organization structure is operating over a population structure, influencing the set of possible exchanges that the agents may have between them, is represented by an *implementation relation* $imp \subseteq (Ro \times Ag) \cup (Li \times Ep)$, where
– $Ro \times Ag$ is the set of all possible *role supports*, i.e., the set of all possible ways of assigning roles to agents, so that if $(r, a) \in imp$, then the social role r is supported by agent a, so that a is said to play role r (possibly in a shared, non-exclusive way) in the given organization;
– $Li \times Ep$ is the set of all possible *link supports*, i.e., the set of all possible ways of supporting links, so that if $(l, e) \in imp$, link l is said to be supported (in a possibly shared, non-exclusive way) by the exchange process e, and so indirectly supported by the agents that participate in e and that play the roles linked by l.

We note that an organization implementation relation imp does not need to be one-to-one: many roles may be assigned to the same agent, many agents may support a given role, many links may be supported by a given exchange process, many exchange processes may support a given link. Moreover, this relation may be partial: some roles may be assigned to no agent, some agents may be have no roles assigned to them, some links may be unsupported, some exchange processes may be supporting no link at all. The agents that have at least one role assigned to them are said to constitute the *support* of the organization in the population. [1]

This flexibility is important when defining the structural dynamics of MAS, because it allows for the definition of "improper" structural states, i.e., structural states where the system's organization is not properly implemented by the sytem's population, which is relevant for the end goal of dealing with the concept of organizational integrity [1].

A *proper implementation relation* is an implementation relation that respects organizational roles and organizational links by correctly translating them in terms of

[1] Note that agents that do not belong to an organization's support may interfere with the functioning of that organization by influencing the behaviors of the supporting agents.

agents, behaviors and exchange processes. Given an implementation relation $imp \subseteq (Ro \times Ag) \cup (Li \times Ep)$, a social role $r \in Ro$ is said to be *properly implemented* by a subset $A \subseteq Ag$ of agents whenever the following conditions hold:

(i) $\forall a \in A : (r, a) \in imp$, i.e., all agents in A participate in the implementation of r;

(ii) $\forall t \in T : \bigcup\{b(t) \mid b \in r\} \subseteq \bigcup\{b'(t) \mid b' \in bc(a), a \in A\}$, i.e., the set of behaviors required by r may be performed by the agents of A (in a possibly shared, non-exclusive way).

A link $l = (r_1, r_2, e) \in Li$ is *properly implemented* by a subset $E \subseteq ec(a_1, a_2)$ of the exchange processes determined by the exchange capability of two agents a_1, a_2, whenever the following conditions hold:

(i) $\forall e \in E : (l, e) \in imp$, i.e., every exchange process in E helps to support the link;

(ii) r_1 e r_2 are properly implemented by the agents a_1 and a_2, respectively; and

(iii) $\forall t \in T : e(t) \subseteq \bigcup\{e'(t) \mid e' \in E\}$, i.e., the exchange process required by l may be performed by the ones of E (in a possibly shared, non-exclusive way).

A time-invariant population-organization structure $PopOrg = (Pop, Org, imp)$ is *properly implemented* if and only imp is a proper implementation relation.

2.2 The Time-Variant Population-Organization Model

Time-Variant Population Structures Time-variant structures change as time goes by. There are three main kinds of possible changes in the momentary population structure $Pop = (Ag, Act, Bh, Ep, bc, ec)$ of a multiagent system: (p1) a change in the behavioral capability $bc(a)$ of an agent $a \in Ag$; (p2) a change in the exchange capability $ec(a_1, a_2)$ of a pair of agents $(a_1, a_2) \in Ag \times Ag$; (p3) a change in the population Ag.

Changes of the kind (p1) may be due either to internal changes in the agent or to changes in the set of passive objects (e.g., tools) with which the agent operates. Changes of the kind (p2) may be due either to changes in the behavioral capability of one of the agents, to changes in the exchange medium (e.g., communication channel) used by the agents, or to changes in some social norm that regulates the exchanges. Changes of the kind (p3) are due to agents entering or leaving the system.

Let T be the time structure, $\mathbf{Ag}$ and $\mathbf{Act}$ be universes of agents and actions, respectively, and $\mathbf{Bh}$ and $\mathbf{Ep}$ universes of behaviors and exchange processes defined over $\mathbf{Ag}$ and $\mathbf{Act}$, in a way similar to that in Sect. 2.1(3). A *time-variant population structure* is a structure $POP = (AG, ACT, BH, EP, Bc, Ec)$ where, for all $t \in T$:

- $AG^t \in \wp(\mathbf{Ag})$ is the system's population, at time t;
- $ACT^t \in \wp(\mathbf{Act})$ is the set of possible agent actions, at time t;
- $BH^t \in \wp(\mathbf{Bh})$ is the set of possible agent behaviors, at time t;
- $EP^t \in \wp(\mathbf{Ep})$ is the set of possible exchange processes between agents, at time t;
- $Bc^t : AG^t \to \wp(BH^t)$ is the behavioral capability function of agents, at time t;
- $Ec^t : AG^t \times AG^t \to \wp(EP^t)$ is the exchange capability function, at time t.

The state at time t of a time-variant population structure, denoted by $POP^t = (AG^t, ACT^t, BH^t, EP^t, Bc^t, Ec^t)$, fixes the population of the system, the set of possible behaviors of each agent and the set of possible exchange processes between each pair of agents, but not the behaviors and exchange processes themselves, which at each time will be chosen from among those possibilities according to the particular internal states of the agents, and the particular states of the (social and physical) environment. Note, however, that the intensional, subjective reasons for such choices are not modelled in the extensional PopOrg model.

Time-Variant Organization Structures There are five main kinds of possible changes in a momentary organization structure $Org = (Ro, Li, lc)$: (o1) a change in a role $r \in Ro$; (o2) a change in a link $l \in Li$; (o3) a change in the set of roles Ro; (o4) a change in the set of links Li; (o5) a change in the link capability lc of the pairs of roles.

A change of kind (o1) may be due, e.g., to a change in the behavior of one of more agents performing the role. A change of the kind (o2) may be due, e.g., to a change in an exchange process that supports the link. Changes of the kind (o3) are either the appearance or the disappearance of roles in the system. Changes of the kind (o4) are either to the appearance or to the disappearance of organizational links in the system. A change of kind (o5) may be due, e.g., to a redistribution of the set of links between organization roles. All such changes may be due to the so-called "reorganization operations" of multiagent systems [5]. The reasons for such operations are essentially of an intensional nature and, thus, are not explicitly represented in the extensional PopOrg model (but their realizations as behavioral processes, and their possible extensional effects, may be explicitly modelled). We note that Sect. 4 of this paper is mainly concerned with changes of kind (o4), that is, changes in the set of links of an organization structure.

Let T be the time structure, and $\mathbf{Ro} \subseteq \wp(\mathbf{Bh})$ and $\mathbf{Li} \subseteq \wp(\mathbf{Ep})$ be the universes of roles and links, respectively. The *time-variant organization structure* of a time-variant population structure $POP = (AG, ACT, BH, EP, Bc, Ec)$ is a structure $ORG = (RO, LI, Lc)$, where for all $t \in T$:

- $RO^t \in \wp(\mathbf{Ro})$ and $LI^t \in \wp(\mathbf{Li})$ are, respectively, the set of possible roles and the set of possible links at time t;
- $Lc^t : RO^t \times RO^t \rightarrow \wp(LI^t)$ is the link capability function at time t.

For each $t \in T$, the organization state $ORG^t = (RO^t, LI^t, Lc^t)$ fixes the sets of possible roles RO^t, links LI^t and link capability function Lc^t that the system may have at that time. Note that a time-invariant organization structure may be modelled as a constant time-variant organization structure.

Time-Variant Implementation Relations As a consequence of any change (p1)-(p3) or (o1)-(o5), the implementation relation imp may be changed either (r1) in the way it relates roles and agents or (r2) in the way it relates links and exchange processes. Besides being changed in its mapping, imp may be changed also in its properness.

Let $POP = (AG, ACT, BH, EP, Bc, Ec)$ be a time-variant population structure and $ORG = (RO, LI, Lc)$ its time-variant organization structure. A *time-variant implementation relation* for ORG over POP is a time-indexed set of implementation relations IMP, with $IMP^t \subseteq (RO^t \times AG^t) \cup (LI^t \times EP^t)$. A *time-variant population-organization structure* is a structure $POPORG = (POP, ORG, IMP)$, where

- $POP = (AG, ACT, BH, EP, Bc, Ec)$, $ORG = (RO, LI, Lc)$ and IMP are, respectively, a time-variant population structure, a time-variant organization structure, and a time-variant implementation relation, as defined above;
- at each $t \in T$, the state of $POPORG$ is given by $POPORG^t = (POP^t, ORG^t, IMP^t)$, where $POP^t = (AG^t, ACT^t, BH^t, EP^t, BC^t, EC^t)$ and $ORG^t = (RO^t, LI^t, Lc^t)$ are such that $IMP^t \subseteq (RO^t \times AG^t) \cup (LI^t \times EP^t)$.

We note that this definition does not guarantee that the relation IMP is proper at each time. That is, we assume that time-variant population-organization structures may pass through structural states where the population improperly implements the organization.

Multiagent Systems with Structural Dynamics The *structural dynamics* of a multiagent system [1] is the dynamics that deals with the way the structure of the system varies in time, thus, it is the dynamics of the system's population and organization.

Let $\mathbf{PopOrg} = (\mathbf{Pop}, \mathbf{Org}, \mathbf{imp})$ be the universe of all possible population-organization structures, with $\mathbf{Pop} = (\mathbf{Ag}, \mathbf{Act}, \mathbf{Bh}, \mathbf{Ep}, \mathbf{bc}, \mathbf{ec})$, $\mathbf{Org} = (\mathbf{Ro}, \mathbf{Li}, \mathbf{lc})$ and $\mathbf{imp} \subseteq (\mathbf{Ro} \times \mathbf{Ag}) \cup (\mathbf{Li} \times \mathbf{Ep})$ are the universes of all possible time-invariant population structures, organization structures and implementation relations, respectively.

A *multiagent system with dynamic structure* is a structure $MAS = (\mathbf{PopOrg}, D)$ where, for each $t \in T$, $D^t \subseteq \mathbf{PopOrg} \times \mathbf{PopOrg}$ is the system's *overall structural dynamics*, such that for any structural state $PopOrg \in \mathbf{PopOrg}$, at time $t \in T$, there is a set of *possible next structural states*, denoted by $D^t(PopOrg) \subseteq \mathbf{PopOrg}$.

Given a particular initial population-organization structure $PopOrg^{t_0}$, the dynamics of its structure is a time-variant population-organization structure $POPORG$, where it holds that $POPORG^{t+1} \in D^t(POPORG^t)$, for any $t \in T$. The choice of the particular next structural state $POPORG^{t+1}$ that will be assumed by the MAS at time $t+1$ is made, at time $t \in T$, on the basis of various intensional, subjective factors extant in the system, like, e.g., preferences of agents, social norms, political powers, etc.

In particular cases, it may happen that the system's overall structural dynamics may be separated into three coordinated sub-structural dynamics $D^t = D^t_P \times D^t_O \times D^t_I$: the *population* dynamics $D^t_P \subseteq \mathbf{Pop} \times \mathbf{Pop}$, the *organizational* dynamics $D^t_O \subseteq \mathbf{Org} \times \mathbf{Org}$, and the *implementation* dynamics $D^t_I \subseteq \mathbf{imp} \times \mathbf{imp}$. In such special cases, the coordination between the system's overall dynamics and the three sub-structural dynamics may be given compositionally by:

$$\begin{aligned}&(Pop', Org', imp') \in D^t((Pop, Org, imp)) \Leftrightarrow \\ &Pop' \in D^t_P(Pop) \;\wedge\; Org' \in D^t_O(Org) \;\wedge\; imp' \in D^t_I(imp)\end{aligned}$$

3 Systems of Exchange Values

In this section, we introduce one of the possible intensional, subjective factor that may influence the evolution of the dynamical structure of a multiagent system, namely, the system of exchange values with which the agents may assess the quality of the exchanges they are having in the system. We adopt here one particular model of system of exchange values [3], which we have used in previous works (e.g., [6]).

This exchange value-based approach to social interactions (cf. also [4]) considers that every social interaction is an exchange of services between the agents involved in it. Exchange values are, then, the values with which agents evaluate the social exchanges they have with each other.

A *service* is any action or behavior that an agent may perform, which influences positively (respect., negatively) the behavior of another agent, favoring (respect., disfavoring) the effort of the latter to achieve a goal. The *evaluation* of a service involves not only affective and emotional reactions, but also comparisons to social standards. Typical evaluations are expressed using *qualitative values* such as: good, very good, bad, very bad, etc. So, they are of a neatly subjective, qualitative, intensional character.

With those evaluations, a qualitative economy of exchange values arises in the social system. Such qualitative economy requires various rules for its regulation. Most of those rules are either of a moral or of a juridical character [3].

Exchange behaviors between two agents α and β can be defined as sequences of exchange steps performed between them. Two kinds of exchange steps are identified [3], called $I_{\alpha\beta}$ and $II_{\alpha\beta}$. Steps of the kind $I_{\alpha\beta}$ are steps in which agent α takes the initiative to perform a service for agent β, with qualitative *cost* (investment) $r_{I\alpha\beta}$. Subsequently, β receives the service, and gets a *benefice* (satisfaction) of qualitative value $s_{I\beta\alpha}$.

If β was to pay back α a return service immediately, he would probably try to "calibrate" his service so that it would have cost r equal to $s_{I\beta\alpha}$, so that α would get a return benefice with value s equal to $r_{I\alpha\beta}$, in order for the exchange to be fair (if the two agents were prone to be fair in their exchanges). The definition of exchange steps assumes, however, that the return service will not be performed immediately, so that a kind of bookkeeping is necessary, in order for the involved values not to be forgotten.

That is the purpose of the two other values involved in the exchange step: $t_{I\beta\alpha}$ is the *debt* that β assumes with α for having received the service and not having payed it back yet; , $v_{I\alpha\beta}$ is the *credit* that α gets on β for having performed the service and not having being payed yet. A *fair* exchange step ([3] calls it an *equilibrated* exchange step) is one where all the involved values are qualitatively equal: $r_{I\alpha\beta} \approx s_{I\beta\alpha} \approx t_{\beta\alpha} \approx v_{I\alpha\beta}$.

To take account of differences between qualitative exchange values, such values are assumed to be comparable with respect to their relative qualitative magnitudes. That is, if EV is the set of qualitative exchange values, it is assumed that values in V can be compared by an order relation $\preceq$, so that $(EV, \preceq)$ is a (partially) ordered set. Thus, e.g., if it happened that $s_{I\beta\alpha} \preceq r_{I\alpha\beta}$, then agent α made an investment, during his service, that was greater than the benefice that agent β got from it.

An exchange step of kind $II_{\alpha\beta}$ is performed in a different way. In it, agent α charges agent β for a credit with qualitative value $v_{II\alpha\beta}$, which he has on β. Subsequently, β acknowledges a debt with value $t_{II\beta\alpha}$ with α, and performs a return service with value $r_{II\beta\alpha}$. In consequence, α gets a return satisfaction with value $s_{II\alpha\beta}$. Fairness for $II_{\alpha\beta}$ steps is defined similarly as for $I_{\alpha\beta}$ steps.

It is assumed that exchange values can be qualitatively added and subtracted from each other, so that *balances of temporal sequences* of exchange steps can be calculated. Besides the above mentioned conditions, one further condition is required in order that a sequence of exchange steps be fair: $\sum v_{II\alpha\beta} \approx \sum v_{I\alpha\beta}$, that is, α should charge a sum of credits which is exactly the total credit he has on β, no more, no less.

In summary, [3] introduces a qualitative algebra with which one can model and analyze social exchanges between agents, determining in a qualitative way the degree of fairness of those exchanges. Note that such algebra operates on 8-tuples of the form

$$(r_{I_{\alpha\beta}}, s_{I_{\beta\alpha}}, t_{I_{\beta\alpha}}, v_{I_{\alpha\beta}}, v_{II_{\alpha\beta}}, t_{II_{\beta\alpha}}, r_{II_{\beta\alpha}}, s_{II_{\alpha\beta}}). \tag{4}$$

4 Exchange Value-based Dynamics of Social Links

This section illustrates one of the possible uses of our extensional model for the structural dynamics of organizations of MAS by showing how it can support the intensional rules of an elementary exchange value-based dynamics of organizational links.

4.1 An Elementary Exchange Value-based Dynamics of Social Links

Other things being equal, the fact that a sequence of exchange steps between two agents is fair, or not, may be a determinant factor in the attitude of those agents toward the possibility of the continuation of the interaction. That is, given enough chances, self-interested agents will tend to establish continued exchanges only with agents from whom they may establish exchanges that are at least fair, if not beneficial, for them [4]. Particular personality traits and various social factors (power, prestige, etc.), however, may interfere with self-interests and lead the agents to seek social exchanges that happen to be far from equilibrium ([6] illustrates this in the context of multiagent systems).

To simplify the issues, we assume that a MAS of self-interested agents adheres to the following rationales concerning the dynamics of organizational links:

- *exchange value-based rationale for the* creation *of an organizational link*: a new organizational link in the MAS is created as soon as an exchange process is positively assessed by the agents playing the roles that will be linked by the link (the exchange process is said to be *officially incorporated* as a link into the organization);
- *exchange value-based rationale for the* destruction *of an organizational link*: a link stops to exist in the multiagent system as soon as the balance of exchange values involved in the exchange processes that implement the link stops to be beneficial to any of the agents performing the roles linked by link (the exchange process is said to be *officially excluded* from the organization of the multiagent system).

We leave open for the agents to apply subjective criteria to determine if any of the conditions mentioned in the above rationales "really" occurred or not. If the social organization has a central control, able to discover at each moment which are the links that the agents would like to establish next between them, then it is up to that central control to determine if enough has been observed in order to create or destroy a link in the organization. If the agents are autonomous, then it is up to them to determine that.

If the agents are autonomous, they may thus disagree on which links should be created or destroyed. In this case, the dynamics of links is open to argumentation and negotiation between them. Then, for organizations based on autonomous agents, no general method can be given for the determination of how the dynamics of inks should evolve. Such dynamics is tightly coupled to the personality traits and social biases that the agents may show with respect to the evaluation of their exchanges.

On the other hand, for organizations where the definitions of the roles prescribe not only the behaviors that the agents playing such roles must have, but also the criteria with which they should evaluate the interactions in which they get involved, it is possible to derive the dynamics of links from the evaluation rules embedded in the roles.

The former case characterizes organizations where the dynamics of links can only be established (at best) *a posteriori*, i.e., after knowing which agent is playing which role in the organization. The latter case characterizes more manageable organizations, where the dynamics of links can be established by an *a priori* analysis of the roles.

4.2 The Rules of the Elementary Exchange Value-based Dynamics of Links

We introduce, now, a minimal set of intensional rules for the exchange value-based dynamics of organizational links in multiagent systems, formalizing the rationales for self-interested agents exposed above.

For simplicity, we consider the case where the organization structure is time-variant, the population structure is time-invariant, each role is implemented by just one single agent, and each link implemented by just one single exchange process.

Let $Pop = (Ag, Act, Beh, Ep, bc, ec)$ be a *time-invariant* population structure, $ORG = (EP, RO, LI)$ be a *time-variant* organization structure implemented by Pop, and let IMP be the *time-variant* implementation relation. They constitute a time-variant population-organization structure $PopORG = (Pop, ORG, IMP)$, which is assumed here to vary just in the set of organizational links, and in their implementations.

There may happen two kinds of changes in the set of links LI^t, at the time $t+1 \in T$: (1) either a new link l is created, so that $LI^{t+1} = LI^t \cup \{l\}$; or (2) a link l is removed from LI^t, so that $LI^{t+1} = LI^t - \{l\}$.

The problem we face here is that of the formalization of the conditions under which, at a moment $t+1$, a link l is added to (or removed from) the set of links LI^t.

Let $EV = (EV, \preceq)$ be the scale of exchange values used by agents $a_1, a_2 \in Ag$ to evaluate their exchanges, and $BEV = EV^8$ be the set of 8-tuples of exchange values that represent balances of exchange values, defined in Sect. 3(4). Let $bal : Ag \times Ag \times Ep \times T \rightarrow BEV$ be so that $bal(a_1, a_2, e, t)$ is the balance of exchange values that agents a_1 and a_2 have accumulated, at time t, along the exchanges they performed through the exchange process $e \in Ep$.

We assume that each agent of the agents $a_1, a_2 \in Ag$ is able to perform an analysis of every possible balance $bal(a_1, a_2, e, t)$ of exchange values that may arise between them, and judge if that balance is beneficial, fair, or harmful for himself. That is, we assume that there exists a (subjective) judgement function $jdg^t(a, bal(a_1, a_2, e, t)) \in \{+1, 0, -1\}$, which we may write as $a \models^t bal(a_1, a_2, e, t) \approx v$, for $v \in \{+1, 0, -1\}$ and $a \in \{a_1, a_2\}$.

Then, the dynamics of organizational links in the Population-Organization model of multiagent systems with self-interested agents is determined by a set of operational rules containing at least the rules introduced below.

Let $[\tau, \tau'], [\tau, \tau') \subseteq T$ respectively be a closed and a right end-open interval of time, with $\tau < \tau'$. Let $a_1, a_2 \in Ag$ be agents respectively playing roles $r_1, r_2 \in Ro$ during the interval $[\tau, \tau']$, that is, $(r_1, a_1), (r_2, a_2) \in IMP^t$, for all $t \in [\tau, \tau']$.

Consider a link $l \in \mathbf{Li}$ between roles $r_1, r_2 \in \mathbf{Ro}$ such that $l \notin LI^t$, for $t \in [\tau, \tau')$, and an exchange process $e \in Ep$ that may possibly support l during the interval $[\tau, \tau']$. Let IMP^t and LI^t be fixed, for all $t \in [\tau, \tau')$. Assume also that $l \in Lc^t(r_1, r_2)$, for all $t \in [\tau, \tau']$.

Let $jdg^t(a, bal(a_1, a_2, e, [\tau, \tau']))$ denote the judgement, at $t \in T$, of the balance of values accumulated in the interval $[\tau, \tau'] \subseteq T$, and let $jdg^t(a, bal(a_1, a_2, e, [\tau, \tau'])) \succeq 0$ mean $jdg^t(a, bal(a_1, a_2, e, [\tau, \tau'])) \approx 0 \vee jdg^t(a, bal(a_1, a_2, e, [\tau, \tau'])) \approx +1$. In this context, the following rule, controlling the introduction of l in $LI^{\tau'}$, is compatible with an exchange value-based account of the link dynamics of the considered system:

$$\frac{a_1 \models^{\tau'} bal(a_1, a_2, e, [\tau, \tau']) \succeq 0 \qquad a_2 \models^{\tau'} bal(a_1, a_2, e, [\tau, \tau']) \succeq 0}{LI^{\tau'} = LI^{\tau} \cup \{l\} \;\wedge\; IMP^{\tau'} = IMP^{\tau} \cup \{(l, e)\}} \; LI_{intro(l)}$$

Analogously, consider an exchange process $e \in Ep$ that supported a link $l \in LI^t$ between roles $r_1, r_2 \in RO^t$ during the interval $[\tau, \tau')$, and that IMP^t and LI^t are fixed, for all $t \in [\tau, \tau')$. Assume that $l \in Lc^t(r_1, r_2)$, for all $t \in [\tau, \tau']$. In this context, for $a \in \{a_1, a_2\}$, the following rule, controlling the elimination of l from LI^τ, is compatible with an exchange value-based account of the link dynamics of the considered system:

$$\frac{a \models^{\tau'} bal(a_1, a_2, e, [\tau, \tau']) \approx -1}{LI^{\tau'} = LI^\tau - \{l\} \ \wedge \ IMP^{\tau'} = IMP^\tau - \{(l, e)\}} \ LI_{elim(l,a)}$$

Note, on the other hand, that the two rules should to be subject to the *proviso* that the interval $[\tau, \tau']$ is large enough to allow the agents to make sound judgements, the notion of "large enough" depending on intensional factors outside de PopOrg model. [2]

5 Related Works and Conclusion

We have presented a temporal extensional model to support a formal dynamics of multiagent systems, by revisiting the PopOrg model and refining it with the notion that social interactions are exchanges. We strived to clearly separate the extensional, structural aspects of the problem, from the intentional, subjective ones. The former deal with the set of possible ways the structure of a multiagent system evolves in time, while the latter deal with the possible causes of the particularities of such evolution.

To illustrate the way the intensional and the extensional aspects of the structural dynamics of a multiagent system may be combined, we made use of an exchange value-based mechanism for the modeling of the subjective assessment of social exchanges, allowing the agents to decide on the start, continuation and termination of an organizational link, thus showing that an intensional mechanism may operate as a causal element in the extensional structural dynamics of the system.

The analysis of organizations from the deontic point of view [7] places itself in the intensional perspective, concerning the expression of regulations (essentially constraints) about the structure and functioning of a multiagent system.

The notion of structural dynamics considered in this paper is closely related to the notion of reorganization of a multiagent system as analyzed, e.g., in [5] and references cited therein. There, the concern is not only with the intensional regulatory mechanism of the structural evolution of the system, but also with the determination of the extensional set of possibilities that such structural evolution presents to the agents that operate in the system.

The denotational and operational semantics of real-time and reactive systems [2] defined models for such systems which are formally keen to most models of multiagent

[2] As an aside, we claim that $\{LI_{intro(l)}, LI_{elim(l,a)}\}$ is the minimal set of rules upon which should lie any exchange value-based dynamics of organizational links, in the PopOrg model, when self-interested agents are considered. Of course, more realistic examples of link dynamics would require additional rules to take care of more complex situations, e.g., rules to deal with links implemented by two or more exchange processes. On the other hand, issues such as the protection of the organization against malicious agents (e.g., agents that provoke the elimination of links by providing a negative evaluation to every exchange), are issues that concern intensional norms related to the security of the organization, which should be reflected in the extensional rules describing the dynamics of the organization, but which should not be dealt with initially at this extensional level.

systems. The similarity comes not from chance, for the agent-based systems were originally developed as models of reactive real-time systems [8]. One readily recognizes, for instance, that reactive programs in state-based specification languages for reactive systems [2] are similar in spirit to the so called procedural knowledge representation that was originally used to specify the behavior of BDI agents [8]: both are means for representing "reactive plans".

Since a signal [2] is essentially a temporal sequence of values of a certain type, signals are similar to the temporal sequences used in the PopOrg model [1]. The similarity is not weakened by our using structural objects as values of the temporal sequences, while the declarative languages designed for the specification of reactive real-time systems use simple data values in signals. Such differences and similarities only stress the need to develop the study of multiagent systems in the perspective of a situated approach, where the system is placed to operate in connection to a real environment.

The PopOrg model was introduced as a minimal model able to deal with the structural dynamical aspects of the functioning of multiagent systems. So, the two components that one would like to add to it in a future work, to allow for the tackling of two essential aspects of such systems, are: first, a mechanism for constituting organizational groups of agents within the system; and, second, the notion of an external environment, the latter being the essential component for construing the system as a situated one.

Thus, it seems to us that the work we presented here produced the core elements for an adequate consideration of the structural dynamics of multiagent systems. They seem to become specially useful when considering systems situated in real environments, whose structural and functional variations press the systems to keep their structures continuously adapted to the demands of those environments.

Acknowledgements. This work was partially supported by FAPERGS and CNPq. We would like to thank the referees for their very valuable comments, many of which could not be incorporated in this version of the paper due to the lack of time and space.

References

1. Demazeau, Y., Costa, A.C.R.: Populations and organizations in open multi-agent systems. In: 1st National Symposium on Parallel and Distributed AI (PDAI'96), Hyderabad, India (1996)
2. Benveniste, A., Berry, G.: The synchronous approach to reactive and real-time systems. Proc. of the IEEE **79** (1991) 1270–1282
3. Piaget, J.: Sociological Studies. Routlege, London (1995)
4. Homans, G.: Social Behavior – Its Elementary Forms. Harcourt, Brace & World, N. Y. (1961)
5. Hübner, J.F., Boissier, O., Sichman, J.S.: Programming MAS reorganisation with moise+. In Meyer, J., Dastani, M., Bordini, R., eds.: Foundations and Practice of Programming Multi-Agent Systems. Number 06261 in Dagstuhl Seminars, IFBI (2006)
6. Dimuro, G.P., Costa, A.C.R., Gonçalves, L.V., Hübner, A.: Centralized regulation of social exchanges between personality-based agents. In Noriega, P., Vázquez-Salceda, J., Boella, G., Boissier, O., Dignum, V., Formara, N., Matson, E., eds.: Coordination, Organizations, Institutions and Norms in MAS II. Number 4386 in LNAI, Springer (2007) 16–23
7. Boella, G., van der Torre, L., Verhagen, H.: Introduction to normative multiagent systems. In Boella, G., van der Torre, L., Verhagen, H., eds.: Normative Multi-agent Systems. Number 07122 in Dagstuhl Seminar Proceedings, IBFI (2007)
8. Georgeff, M., Lansky, A.: Procedural knowledge. Proc. of the IEEE **74** (1986) 1383–1398

On the Multimodal Logic of Normative Systems

Pilar Dellunde

IIIA - CSIC and Universitat Autonoma de Barcelona
pilar@iiia.csic.es

Abstract. We introduce Multimodal Logics of Normative Systems as a contribution to the development of a general logical framework for reasoning about normative systems over logics for Multi-Agent Systems. Given a multimodal logic L, for every modality $\Box_i$ and normative system η, we expand the language adding a new modality $\Box_i^\eta$ with the intended meaning of $\Box_i^\eta \phi$ being "ϕ is obligatory in the context of the normative system η over the logic L". In this expanded language we define the Multimodal Logic of Normative Systems over L, for any given set of normative systems N, and we give a sound and complete axiomatisation for this logic, proving transfer and model checking results. The special case when L and N are axiomatised by sets of Sahlqvist or shallow modal formulas is studied.

Keywords: Fusions of Logics, Multimodal Logics, Normative Systems, Multi-Agent Systems, Model Theory, Sahlqvist Formulas

1 Introduction

Recent research on the logical foundations of Multi-Agent Systems (MAS) has centered its attention in the study of normative systems. The notion of electronic institution is a natural extension of human institutions by permitting not only humans but also autonomous agents to interact with one another. Institutions are used to regulate interactions where participants establish commitments and to facilitate that these commitments are upheld, the institutional conventions are devised so that those commitments can be established and fulfilled (see [1] for a general reference of the role of electronic institutions to regulate agents interactions in MAS). Over the past decade, normative systems have been promoted for the coordination of MAS and the engineering of societies of self-interested autonomous software agents. In this context there is an increasing need to find a general logical framework for the study of normative systems over the logics for MAS.

Given a set of states S and a binary accessibility relation R on S, a normative system η on the structure (S, R) could be understood as a set of constraints $\eta \subseteq R$ on the transitions between states, the intended meaning of $(x, y) \in \eta$ being "the transition from state x to state y is not legal according to normative system η". Several formalisms have been introduced for reasoning about normative systems over specific logics, two examples are worth noting: Normative ATL (NATL), proposed in [2] and Temporal

Logic of Normative Systems (NTL) in [3]. NATL is an extension to the Alternating-Time Temporal Logic and contains cooperation modalities of the form $<< \eta : C >> \phi$ with the intended interpretation that "C has the ability to achieve ϕ within the context of the normative system η". NTL is a conservative generalization of the Branching-Time Temporal Logic CTL. In NTL, the path quantifiers A ("on all paths...") and E ("on some path...") are replaced by the indexed deontic operators O_η ("it is obligatory in the context of the normative system η that..") and P_η ("it is permissible in the context of the normative system η that...").

The Multimodal Logic of Normative Systems introduced in this article is a contribution to define a general logical framework for reasoning about normative systems over logics for MAS, for this purpose we generalize to arbitrary logics the approaches taken in [2] and [3]. At the moment, we are far from obtaining a unique formalism which addresses all the features of MAS at the same time, but the emerging field of combining logics is a very active area and has proved to be successful in obtaining formalisms which combine good properties of the existing logics. In our approach, we regard the Logic of Normative Systems over a given logic L, as being the fusion of logics obtained from L and a set of normative systems over L, this model-theoretical construction will help us to understand better which properties are preserved under combinations of logics over which we have imposed some restrictions and to apply known transfer results (for a general account on the combination of logics, we refer to [4] and [5], and as a general reference on multimodal logic, to [6]). There are some advantages of using these logics for reasoning about MAS: it is possible to compare whether a normative system is more restrictive than the other, check if a certain property holds in a model of a logic once a normative system has restricted its accessibility relation, model the dynamics of normative systems in institutional settings, define a hierarchy of normative systems (and, by extension, a classification of the institutions) or present a logical-based reasoning model for the agents to negotiate over norms.

This paper is structured as follows. In Section 2 we present an example in order to motivate the introduction of the general framework. In Section 3 we give a sound and complete axiomatisation for the Multimodal Logic of Normative Systems, proving transfer results and we address a complexity issue for model checking. In Section 4 we restrict our attention to logics with normative systems that define elementary classes of modal frames, we have called them *Elementary Normative Systems (ENS)* and we prove completeness and canonicity results for them. Elementary classes include a wide range of formalisms used in describing MAS, modelling different aspects of agenthood, some temporal logics, logics of knowledge and belief, logics of communication, etc. Finally, in Section 5 we come back to our first example in Section 2, showing how our framework can be applied to multiprocess temporal structures, Section 6 is devoted to future work.

2 Multiprocess Temporal Frames and Normative Systems

In a multi-agent institutional environment, in order to allow agents to successfully interact with other agents, they share the dialogic framework. The expressions of the communication language in a dialogic framework are constructed as formulas of the type $\iota(\alpha_i : \rho_i, \alpha_j : \rho_j, \phi, \tau)$, where ι is an illocutionary particle, α_i and α_j are agent terms, ρ_i and ρ_j are role terms and τ is a time term. An scene is specified by a graph where the nodes of the graph represent the different states of the conversation and the arcs connecting the nodes are labelled with illocution schemes.

Several formalisms for modelling interscene exchanges between agents have been introduced using multimodal logics. For instance, in [7] the authors provide an alternating offers protocol to specify commitments that agents make to each other when engaging in persuasive negotiations using rewards. Specifically, the protocol details, how commitments arise or get retracted as a result of agents promising rewards or making offers. The protocol also standardises what an agent is allowed to say or what it can expect to receive from its opponent. The multimodal logic presented in [7] introduces modalities $\Box_\phi$ for expressions ϕ of the communication language.

More formally, given a finite set of propositional atomic formulas, we could define the set of formulas of such a multimodal communication language in the following way:

$$\phi ::= p \mid \top \mid \bot \mid \neg\alpha \mid \alpha \wedge \alpha \mid \Box_{\phi_1}\alpha \mid \ldots \mid \Box_{\phi_k}\alpha$$

where p is an atomic propositional formula, α is a propositional formula and $\phi_1, \ldots, \phi_k$ are formulas of the communication language.

The standard Kripke semantics of these logics can be given by means of multiprocess temporal frames. We say that $\Xi = (S, R_{\phi_0}, \ldots, R_{\phi_k})$ is a *multiprocess temporal frame* if and only if S is a set of states and for every $i \leq k$, R_{ϕ_i} is a binary relation on S such that $R = \bigcup_{i \leq k} R_{\phi_i}$ is a serial relation (that is, for every $s \in S$ there is $t \in S$ such that $(s, t) \in R$). A *multiprocess temporal model* is a Kripke model with a multiprocess temporal frame.

Let M be a multiprocess temporal model and $w \in M$, the satisfiability relation for the modalities $\Box\phi_i$ is defined as usual:

$$M, w \models \Box_{\phi_i}\alpha \text{ iff for all } w' \in M \text{ such that } wR_{\phi_i}w'$$

$$M, w' \models \alpha$$

Some examples of the protocols introduced in [7] can be formalised by formulas of the following form: $\Box_{\phi_1} \ldots \Box_{\phi_l}\bot$. For instance, with the formula $\Box_{Offer(i,x)}\Box_{Offer(i,y)}\bot$, with $x \neq y$, we can express that it is not allowed to agent i to do two different offers one immediately after the other. Let us see now how formulas like $\Box_{\phi_1} \ldots \Box_{\phi_l}\bot$ can be understood as sets of constraints on the transitions between states. Given a multiprocess temporal frame $\Xi = (S, R_{\phi_0}, \ldots, R_{\phi_k})$, consider the following set of finite sequences of elements of S:

$$\Delta_\Xi = \{(a_0, \ldots, a_m) : \forall j < m, \exists i \leq k \text{ such that } a_j R_{\phi_i} a_{j+1}\}$$

Then, a *normative system* η on the frame Ξ could be defined as a subset of Δ_Ξ. Intuitively speaking, a sequence $(a_0, \ldots, a_m) \in \eta$ if and only if this sequence of transitions is not legal according to normative system η. In our previous example, given a frame, the formula $\Box_{Offer(i,x)}\Box_{Offer(i,y)}\bot$, can be regarded as the following normative system (that is, the following set of finite sequences of the frame):

$$\{(a_0, a_1, a_2) : \text{ such that } a_0 R_{Offer(i,x)} a_1 \text{ and } a_1 R_{Offer(i,x)} a_2\}$$

Thus, any model satisfying the protocol introduced by $\Box_{Offer(i,x)}\Box_{Offer(i,y)}\bot$ can not include such sequences.

When defining an scene in an electronic institution we could be interested in comparing different protocols in order to show which of them satisfy some desired properties. In order to do so we could extend our multimodal language with additional modalities $\Box^{\eta}_{\phi_i}$, one for each normative system we want to consider. Next section is devoted to the study of the logical properties of these languages and later on, we will come back to our example applying this general framework.

3 Multimodal Logics of Normative Systems

We introduce first some notation and basic facts about multimodal languages. A *finite modal similarity type* $\tau = \langle F, \rho \rangle$ consists of a finite set F of modal operators and a map $\rho : F \to \omega$ assigning to each $f \in F$ a finite arity $\rho(f) \in \omega$. Finite propositional modal languages of type τ are defined in the usual way by using finitely many propositional variables, the operators in F and the boolean connectives $\wedge, \vee, \neg, \to, \leftrightarrow, \top, \bot$. For monadic modalities we use the usual notation $\Box_f$.

A *modal finitary structural consequence relation* $\vdash$ of similarity type τ is a relation between sets of formulas and formulas of the finite propositional modal language of type τ satisfying:

- $\phi \in \Gamma \Rightarrow \Gamma \vdash \phi$
- If $\Gamma \subseteq \Delta$ and $\Gamma \vdash \phi$, then $\Delta \vdash \phi$
- If $\Gamma \vdash \Delta$ and $\Delta \vdash \phi$, then $\Gamma \vdash \phi$
- $\Gamma \vdash \phi \Rightarrow s\Gamma \vdash s\phi$, for all substitutions s
- If $\Gamma \vdash \phi$, then there exist a finite subset Γ_0 of Γ with $\Gamma_0 \vdash \phi$
- $\vdash \phi$, for every classical tautology ϕ
- $p, p \to q \vdash q$
- For every $f \in F$,

$$p_0 \leftrightarrow q_0, \ldots, p_{\rho(f)} \leftrightarrow q_{\rho(f)} \vdash f(p_0, \ldots, p_{\rho(f)}) \leftrightarrow f(q_0, \ldots, q_{\rho(f)})$$

And we say that a subset Λ of modal formulas is a *classical modal logic* of similarity type τ iff there exists a modal finitary structural consequence relation $\vdash$ of similarity type τ such that $\Lambda = \Lambda(\vdash)$, where $\Lambda(\vdash) = \{\phi : \emptyset \vdash \phi\}$. It is said that that Λ is *consistent* if $\bot \notin \Lambda$.

Given a type $\tau = \langle F, \rho \rangle$, a *Kripke frame* of type τ is an structure $(S, R_f)_{f \in F}$, where S is nonempty and for every $f \in F$, R_f is a binary relation on S.

Definition 1 *A normative system over a Kripke frame $(S, R_f)_{f \in F}$ is a subset of the following set of finite sequences of S:*

$$\{(a_0, \ldots, a_m) : \forall j < m, \exists f \in F \textit{ such that } a_j R_f a_{j+1}\}$$

Observe that Definition 1 extends to the multimodal setting the definition of normative system introduced in Section 2 of [3]. Examples of classical modal logics with semantics based on Kripke frames are Propositional Dynamic Logic (PDL), Alternating-Time Temporal Logic (ATL) and Computational Tree Logic (CTL), but CTL*, the Full Computational Tree Logic is not a classical modal logic because it is not closed under uniform substitution.

Now we introduce in the language a new finite set of symbols N to denote normative systems. Given a finite propositional modal language of type $\tau = \langle F, \rho \rangle$, for every normative system $\eta \in N$, let τ^η be the type whose modalities are $\{f^\eta : f \in F\}$ and $\tau^N = \bigcup_{\eta \in N} \tau^\eta$. For every set of formulas Γ, let us denote by Γ^η the set of formulas of type τ^η obtained from Γ by substituting every occurrence of the modality f by f^η. The monadic operators $\Diamond_f$ are defined in the usual way as abbreviations $\Diamond_f \phi \equiv \neg \Box_f \neg \phi$ and we have also the corresponding $\Diamond_f^\eta$.

Given a classical modal logic L with semantics based on Kripke frames, we define the *Multimodal Logic of Normative Systems* over L, denoted by L^N, as being the smallest classical modal logic in the expanded language τ^N which contains L and L^η, for every $\eta \in N$.

Theorem 2 *Let L be a consistent classical modal logic axiomatised by a set Γ of formulas. Then,*

1. $\Gamma^N = \Gamma \cup \bigcup \{\Gamma^\eta : \eta \in N\}$ *is an axiomatisation of* L^N.
2. L^N *is a conservative extension of L.*
3. *If L is a decidable logic, then* L^N *is decidable.*

Proof: Since we have introduced a finite set of disjoint similarity types $\{\tau^\eta : \eta \in N\}$, we can define the fusion $\bigoplus < L^\eta : \eta \in N >$ of disjoint copies of the logic L. Observe that, so defined, $L^N = \bigoplus < L^\eta : \eta \in N >$ and Γ^N is an axiomatisation of L^N. Then, by an early result of Thomason [8], L^N is a conservative extension of L. Finally we can apply Theorem 6.11 of [9], to obtain the corresponding transfer result. $\Box$

In [10] a weak notion of normality is introduced to prove some additional transfer results for the fusion of logics. Let us assume that our classical modal logics satisfy the two conditions of Definition 2.5 of [10]:

1. For every $f \in F$, the semantics of $f(p_0, \dots, p_{\rho(f)})$ is a monadic first-order formula.
2. For each R_f, there is a derived connective $\Box_f$ such that the formula $\Box_f p$ expresses $\forall x(yR_f x \rightarrow Px)$ and is closed under the necessitation rule: If $\phi \in \Lambda$, then $\Box_f \phi \in \Lambda$.

This second condition corresponds to the notion of normality, but it is weaker than the usual normality requirement. Observe that the operators U and S (until and since) of Temporal Logic are only normal in the first position and not in the second. However, they satisfy conditions 1. and 2., the binary ordering $<$ can be associated with U and the binary ordering $>$ can be associated with S, thus condition 1. is satisfied. The monadic modalities H and G are derivable connectives, that satisfy the requirement of condition 2.

Following the lines of the proof of Theorem 2, by using Theorems 3.6 and 3.10 of [10], we can obtain the following transfer theorem:

Theorem 3 *Let L be a consistent classical modal logic axiomatised by a set Γ of formulas and such that satisfies conditions 1. and 2. above. Then, If L is complete and sound over the class of frames C, then L^N is also complete and sound over the class of frames $\bigoplus < C^\eta : \eta \in N >$.*

As an application of Theorems 2 and 3 we obtain that the Multimodal Logic of Normative Systems over the logics CTL and PDL, has a sound and complete axiomatisation, is decidable and has the Finite Model Property, because CTL and PDL are decidable and complete over the class of finite frames.

We end this section by introducing a model checking result. Given a frame $\Xi = (S, R_f)_{f \in F}$, we say that a subset of S is *connected* if for every $s, t \in S$, $(s,t) \in (\bigcup \left\{ (R_f \cup R_f^{-1} : f \in F \right\})^*$, where for any relation R, R^* denotes the transitive closure of R. We say that the frame Ξ is connected if its domain S is a connected set. Observe that, for every classical modal logic L that satisfies conditions 1. and 2. stated above and it is complete with respect to a class of connected frames, by Theorem 3, the Multimodal Logic of Normative Systems over L is also complete with respect to a class of connected frames.

Theorem 4 *Let L be a classical modal logic in a finite similarity type $\tau = \langle F, \rho \rangle$ and let $(S, R_f^\eta)_{f \in F, \eta \in N}$ be a finite model of the Multimodal Logic of Normative Systems over L such that the restriction of the model $(S, R_f^\eta)_{f \in F, \eta \in N}$ to the similarity type τ^η is connected. Then, the complexity of model checking a formula ϕ of type τ^N is*

$$O(\textstyle\sum_{\eta \in N} m_\eta + n \cdot k) + \textstyle\sum_{\eta \in N}((O(k) + O(n)) \cdot C_L(m_\eta, n, k))$$

where $m_\eta = \sum_{f \in F} \left| R_f^\eta \right|$, $n = |S|$, k is the length of the formula ϕ and $C_L(m_\eta, n, k)$ is the complexity of model checking for logic L as a function of m_η, n and k.

Proof: By Theorem 2, L^N is a conservative extension of L and for every $\eta \in N$ the restriction of the model $(S, R_f^\eta)_{f \in F, \eta \in N}$ to the similarity type τ^η is a model of L and is connected by assumption. This fact allows us to generalize the result on temporal logics of Theorem 5.2 of [11]. We can express the complexity of a combined model checker for L^N in terms of a model checker for L. □

For example, in the case of the Multimodal Logic of Normative Systems over CTL, the overall cost of the model checker for this logic is linear in the size of the model and in the length of the formula.

4 Elementary Normative Systems

There are some advantages of using Multimodal Logics of Normative Systems for reasoning about MAS: it is possible to compare whether a normative system is more restrictive than the other, check if a certain property holds in a model of a logic once a normative system has restricted its accessibility relation, model the dynamics of normative systems in institutional settings, define a hierarchy of normative systems (and, by extension, a classification of the institutions) or present a logical-based reasoning model for the agents to negotiate over norms. Up to this moment we have introduced an extensional definition of normative system (see Definition 1), in this section we present our first attempt to classify normative systems, we restrict our attention to normative systems defined by certain sets of first-order formulas, but only over some class of normal multimodal logics with standard Kripke semantics.

The choice of Sahlqvist formulas in this section is due, on the one hand, to the fact that a wide range of formalisms for MAS can be axiomatised by a set of such formulas (see next section). On the other hand, for the good logical properties of these logics (canonicity, transfer results, etc.). In Section 3 we have presented a general setting for dealing with any classical modal logic. Now, we focus only on some particular kind of logics. We want to study the specific properties of their normative systems that can be proved by using only the fact that these logics are axiomatised by sets of Sahlqvist formulas.

Given a set of modal formulas Σ, the *frame class defined by* Σ is the class of all frames on which each formula in Σ is valid. A frame class is *modally definable* if there is a set of modal formulas that defines it, and it is said that the frame class is *elementary* if it is defined by a first-order sentence of the frame correspondence language (the first-order language with equality and one binary relation symbol for each modality). An *Elementary Normative System* (ENS) is a propositional modal formula that defines an elementary class of frames and a normative system in any frame.

Throughout this and next section we assume that our modal languages have standard Kripke semantics and their modal similarity types have only a finite set of monadic modalities $\{\Box_f : f \in F\}$ and a finite set of propositional variables. Given a classical modal logic L and a set of Elementary Normative Systems N over L, for every $\eta \in N$ we generalize the notion introduced in Section 3 by defining the *Multimodal Logic of Normative Systems* over L and N, denoted by L^N, as being the smallest normal logic in the expanded language which contains L, N and every L^η. We now present a sound and complete axiomatisation and prove some transfer results in the case that L is axiomatised by a set of Sahlqvist formulas and N is a set of Sahlqvist formulas. We denote by $L(\eta)$ the smallest normal logic of similarity type τ^η which includes $L^\eta \cup \{\eta\}$.

Definition 5 (Sahlqvist formulas) *A modal formula is* positive (negative) *if every occurrence of a proposition letter is under the scope of an even (odd) number of negation signs. A* Sahlqvist antecedent *is a formula built up from* $\top$, $\bot$, *boxed atoms of the form* $\Box_{i_1} \ldots \Box_{i_l} p$, *for* $i_j \in I$ *and negative formulas, using conjunction, disjunction and diamonds. A* Sahlqvist implication *is a formula of the form* $\phi \rightarrow \varphi$, *when* ϕ *is a Sahlqvist antecedent and* φ *is positive. A* Sahlqvist formula *is a formula that is obtained from Sahlqvist implications by applying boxes and conjunction, and by applying disjunctions between formulas that do not share any propositional letters.*

Observe that $\bot$ and $\top$ are both Sahlqvist and ENS formulas. Intuitively speaking, $\bot$ is the trivial normative system, in $\bot$ every transition is forbidden in every state and in $\top$ every action is legal. In the sequel we assume that for every set N of ENS, $\top \in N$.

Theorem 6 *Let L be a classical normal modal logic axiomatised by a set Γ of Sahlqvist formulas and N a set of ENS Sahlqvist formulas, then:*

1. $\Gamma^N = \Gamma \cup N \cup \bigcup\{\Gamma^\eta : \eta \in N\}$ *is an axiomatisation of* L^N.
2. L^N *is complete for the class of Kripke frames defined by* Γ^N.
3. L^N *is canonical.*
4. *If L and* L^η *are consistent, for every* $\eta \in N$*, and* ***P*** *is one of the following properties:*
   - *Compactness*
   - *Interpolation Property*
   - *Halldén-completeness*
   - *Decidability*
   - *Finite Model Property*[1]

   then L^N *has* ***P*** *iff L and* $L(\eta)$ *have* ***P****, for every* $\eta \in N$.

Proof: $1-3$ follows directly from the Sahlqvist's Theorem. The main basic idea of the proof of 4 is to apply the Sahlqvist's Theorem to show first that for every $\eta \in N$, the smallest normal logic of similarity type τ^η which includes $\Gamma^\eta \cup \{\eta\}$ is $L(\eta)$, is a complete logic for the class of Kripke frames defined by $\Gamma^\eta \cup \{\eta\}$ and is canonical (observe that this logic is axiomatised by a set of Sahlqvist formulas). Now, since for every Elementary Normative System $\eta \in N$ we have introduced a disjoint modal similarity type τ^η, we can define the fusion of the logics $\bigoplus < L(\eta) : \eta \in N >$. It is enough to check that $L^N = \bigoplus < L(\eta) : \eta \in N >$ (remark that $L^\top = L$) and using transfer results for fusions of consistent logics (see for instance [12] and [10]) we obtain that L^N is a conservative extension and that decidability, compactness, interpolation, Hállden-completeness and the Finite Model Property are preserved. □

We study now the relationships between normative systems. It is interesting to see how the structure of the set of all the ENS over a logic L (we denote it by $N(L)$) inherits its properties from the set of first-order counterparts. A natural relationship could be defined between ENS, the relationship of being one *less restrictive* than another, let us denote it by $\preceq$. Given η, η', it is said that $\eta \preceq \eta'$ iff the first-order formula $\phi_{\eta'} \rightarrow \phi_\eta$ is valid (when for every $\eta \in N$, ϕ_η is the translation of η). The relation $\preceq$ defines a partial order on $N(L)$ and the pair $(N(L), \preceq)$ forms a complete lattice with least upper bound $\bot$ and greatest lower bound $\top$ and the operations $\wedge$ and $\vee$.

Now we present an extension of the Logic of Elementary Normative Systems over a logic L with some inclusion axioms and we prove completeness and canonicity results. Given a set N of ENS, let I^{N^+} be the following set of formulas:

$$\left\{\Box_{i_1} \ldots \Box_{i_l} p \rightarrow \Box^\eta_{i_1} \ldots \Box^\eta_{i_l} p : i_j \in I, \eta \in N\right\}$$

and I^{N^*} the set:

$$\left\{\Box^{\eta'}_{i_1} \ldots \Box^{\eta'}_{i_l} p \rightarrow \Box^\eta_{i_1} \ldots \Box^\eta_{i_l} p : i_j \in I, \eta \preceq \eta', \eta, \eta' \in N\right\}$$

[1] For the transfer of the Finite Model Property it is required that there is a number n such that each $L(\eta)$ has a model of size at most n.

Corollary 7 *Let L be a normal modal logic axiomatised by a set Γ of Sahlqvist formulas and N a set of ENS Sahlqvist formulas, then:*

1. *$\Gamma^{N^+} = \Gamma^N \cup I^{N^+}$ is an axiomatisation of the smallest normal logic with contains L^N and the axioms I^{N^+}, is complete for the class of the Kripke frames defined by Γ^{N^+} and is canonical. We denote this logic by L^{N^+}.*
2. *$\Gamma^{N^*} = \Gamma^N \cup I^{N^*} \cup I^{N^+}$ is an axiomatisation of the smallest normal logic with contains L^N and the axioms $I^{N^*} \cup I^{N^+}$, is complete for the class of the Kripke frames defined by Γ^{N^*} and is canonical. We denote this logic by L^{N^*}.*
3. *If L^N is consistent, both L^{N^+} and L^{N^*} are consistent.*

Proof: Since for every $i_j \in I$ every $\eta, \eta' \in N$, the formulas $\Box_{i_1} \ldots \Box_{i_l} p \to \Box^{\eta}_{i_1} \ldots \Box^{\eta}_{i_l} p$ and $\Box^{\eta'}_{i_1} \ldots \Box^{\eta'}_{i_l} p \to \Box^{\eta}_{i_1} \ldots \Box^{\eta}_{i_l} p$ are Sahlqvist, we can apply Theorem 6. In the case that L^N is consistent, consistency is guaranteed by the restriction to pairs $\eta \preceq \eta'$ and for the fact that η and η' are ENS. $\Box$

Observe that for every frame $(S, R_f, R^{\eta}_f)_{f \in F, \eta \in N}$ of the logic L^{N^*},

$$R^{\eta}_{i_1} \circ \ldots \circ R^{\eta}_{i_l} \subseteq R_{i_0} \circ \ldots \circ R_{i_l},$$

and for $\eta \preceq \eta'$, $R^{\eta}_{i_1} \circ \ldots \circ R^{\eta}_{i_l} \subseteq R^{\eta'}_{i_1} \circ \ldots \circ R^{\eta'}_{i_1}$, where $\circ$ is the composition relation.

We end this section introducing a new class of modal formulas defining elementary classes of frames, the shallow formulas (for a recent account of the model theory of elementary classes and shallow formulas we refer the reader to [13]).

Definition 8 *A modal formula is* shallow *if every occurrence of a proposition letter is in the scope of at most one modal operator.*

It is easy to see that every closed formula is shallow and that the class of Sahlqvist and shallow formulas don't coincide: $\Box_1(p \vee q) \to \Diamond_2(p \wedge q)$ is an example of shallow formula that is not Sahlqvist. Analogous results to Theorem 6 and Corollary 7 hold for shallow formulas, and using the fact that every frame class defined by a finite set of shallow formulas admits polynomial filtration, by Theorem 2.6.8 of [13], if L is a normal modal logic axiomatised by a finite set Γ of shallow formulas and N is a finite set of ENS shallow formulas, then the frame class defined by Γ^N has the Finite Model Property and has a satisfiability problem that can be solved in NEXPTIME.

5 Some examples

Different formalisms have been introduced in the last twenty years in order to model particular aspects of agenthood (temporal Logics, logics of knowledge and belief, logics of communication, etc). We show in this section that several logics proposed for describing Multi-Agents Systems are axiomatised by a set of Sahlqvist or shallow formulas and therefore we could apply our results to the study of their normative systems. Let us come back to our previous example of Section 2, the multiprocess temporal frames. We have introduced first this basic temporal logic of transition systems, not because it is specially interesting in itself, but because is the logic upon which other temporal logics are built and because it is a clear and simple example of how our framework can work.

Remember that $\Xi = (S, R_0, \ldots, R_k)$ is a *multiprocess temporal frame* if and only if S is a set of states, for every $i \leq k$, R_i is a binary relation on S such that $R = \bigcup_{i \leq k} R_i$ is a serial relation (that is, for every $s \in S$ there is $t \in S$ such that $(s, t) \in R$). It is easy to see that $\Xi = (S, R_0, \ldots, R_k)$ is a multiprocess temporal frame if and only if the formula of the corresponding multimodal language

$$\Diamond_0 \top \vee \ldots \vee \Diamond_k \top \text{ (MPT)}$$

is valid in Ξ. Let us denote by $MPTL$ the smallest normal logic containing axiom (MPT). For every nonempty tuple $(i_1, \ldots, i_l)$ such that for every $j \leq l$, $i_j \leq k$, consider the formula $\Box_{i_1} \ldots \Box_{i_l} \bot$. Observe that every formula of this form is shallow and ENS. We state now without proof a result on the consistency of this kind of normative systems over $MPTL$ that will allow us to use the logical framework introduced in the previous section.

Proposition 9 *Let N be a finite set of normative systems such that for every $\eta \in N$, there is a finite set X of formulas of the form $\Box_{i_1} \ldots \Box_{i_l} \bot$ such that η is the conjunction of all the formulas in X, $\bot \notin X$ and the following property holds:*

If $\Box_{i_1} \ldots \Box_{i_l} \bot \notin X$, there is $j \leq k$ such that $\Box_{i_1} \ldots \Box_{i_l} \Box_j \bot \notin X$.

Then, the logic $MPTL^N$ is consistent, complete, canonical, has the Finite Model Property and has a satisfiability problem that can be solved in NEXPTIME.

In general, a normal multimodal logic can be characterized by axioms that are added to the system K_m, the class of *Basic Serial Multimodal Logics* is characterized by subsets of axioms of the following form, requiring that AD(i) holds for every i,

- $\Box_i p \rightarrow \Diamond_i p$ AD(i)
- $\Box_i p \rightarrow p$ AT(i)
- $\Box_i p \rightarrow \Box_j p$ AI(i)
- $p \rightarrow \Box_i \Diamond_j p$ AB(i,j)
- $\Box_i p \rightarrow \Box_j \Box_k p$ A4(i,j,k)
- $\Diamond_i p \rightarrow \Box_j \Diamond_k p$ A5(i,j,k)

An example of a Kripke frame of $MPTL$ in which none of the previous axioms is valid is $\Xi = (\{0, 1, 2\}, \{(0, 1), (2, 0)\}, \{(1, 2)\})$. In particular, our example shows that the Multimodal Serial Logic axiomatised by $\{AD(i) : i \leq k\}$, is a proper extension of $MPTL$. Observe that any logic in the class BSML is axiomatised by a set of Sahlqvist formulas, therefore we could apply the framework introduced before to compare elementary normative systems on these logics.

Another type of logics axiomatised by Sahlqvist formulas are many Multimodal Epistemic Logics. Properties such as positive or negative introspection can be expressed by $\Box_i p \rightarrow \Box_i \Box_k p$ and $\neg \Box_i p \rightarrow \Box_i \neg \Box_i p$ respectively. And formulas like $\Box_i p \rightarrow \Box_j p$ allow us to reason about multi-degree belief.

The Minimal Temporal Logic K_t is axiomatised by the axioms $p \to HFp$ and $p \to GPp$ which are also Sahlqvist formulas. Some important axioms such as linearity $Ap \to GHp \wedge HGp$, or density $GGp \to Gp$, are Sahlqvist formulas, and we can express the property that the time has a beginning with an ENS. By adding the next-time modality, X, we have an ENS which expresses that every instant has at most one immediate successor.

6 Future work

Along this work, in Sections 4 and 5, we have dealt only with multimodal languages with monadic modalities, but by using the results of Goranko and Vakarelov in [14] on the extension of the class of Sahlqvist formulas in arbitrary polyadic modal languages to the class of inductive formulas, it would be possible to generalize our results to polyadic languages.

We will proceed to apply our results to different extended modal languages, such as reversive languages with nominals (in [14], the elementary canonical formulas in these languages are characterized) or Hybrid Logic (in [13], Hybrid Sahlqvist formulas are proved to define elementary classes of frames). Future work should go beyond Elementary Normative Systems and consider the study of sets of normative systems expressed by other formal systems.

Acknowledgements The author wishes to express her thanks to Carles Sierra, Pablo Noriega and the reviewers of this paper for their helpful comments. The author was partially supported by Project TIN2006-15662-C02-01.

References

1. P. Noriega. Fencing the Open Fields: Empirical Concerns on Electronic Institutions, in O. Boissier, V. Dignum, G. Lindemann, E. Matson, S. Ossowski, J. Padget, J. S. Sichman and J. Vázquez-Salceda (ed.) *Coordination, Organizations, Institutions, and Norms in Multi-Agent Systems*, Springer LNCS 3913 (2006) 82–98.
2. W. van der Hoek and M. Wooldridge. On obligations and normative ability: towards a logical analysis of the social contract, *Journal of Applied Logic*, 3 (2005) 396–420.
3. T. Ågotnes, W. van der Hoek, J.A. Rodríguez-Aguilar, C. Sierra and M. Wooldridge. On the Logic of Normative Systems, *Twentieth International Joint Conference on AI, IJCAI07*, AAAI Press (2007) 1175–1180.
4. D. M. Gabbay *Fibring Logics*, Oxford Logic Guides, 38 (1999).
5. A. Kurucz. Combining Modal Logics, in P. Blackburn, J. van Benthem and F. Wolter (ed.) *Handbook of Modal Logic* Chapter 15, Elsevier (2007) 869–928.
6. P. Blackburn, J. van Benthem and F. Wolter (ed.) *Handbook of Modal Logic* Chapter 15, Elsevier (2007).
7. S. D. Ramchurn, C.Sierra, Ll. Godo, N. R. and Jennings, N. R. 2006. Negotiating using rewards. In Proceedings of the Fifth international Joint Conference on Autonomous Agents and Multiagent Systems (Hakodate, Japan, May 08 - 12, 2006). AAMAS '06. ACM Press, New York, NY, 400–407.

8. S. K. THOMASON. Independent Propositional Modal Logics, *Studia Logica*, 39 (1980) 143–144.
9. F. BAADER, S. GHILARDI AND C. TINELLI. A new combination procedure for the word problem that generalizes fusion decidability results in modal logics, Information and Computation, 204 (2006) 1413–1452.
10. M. FINGER, M. A. WEISS. The Unrestricted Combination of Temporal Logic Systems, *Logic Journal of the IGPL*, 10 (2002) 165–189.
11. M. FRANCESCHET, A. MONTANARI AND M. DE RIJKE. Model Checking for Combined Logics with an Application to Mobile Systems *Automated Software Engineering*, 11 (2004) 289–321.
12. F. WOLTER. Fusions of modal logics revisited, in M. KRACHT, M. DE RIJKE, H. WANSING AND M. ZAKHARYASHEV (eds.) *Advances in Modal Logic* CSLI, Stanford, CA. (1998).
13. B. D. T. CATE. Model Theory for extended modal languages, *Ph.D Thesis, Institute for Logic, Language and Computation, Universiteit van Amsterdam*, ILLC Dissertation Series DS-2005-01 (2005).
14. V. GORANKO AND D. VAKARELOV. Elementary Canonical Formulae: extending Sahlqvist's Theorem, *Annals of Pure and Applied Logic*, 141 (2006) 180–217.

A Distributed Architecture for Norm Management in Multi-Agent Systems

A. García-Camino[1], J. A. Rodríguez-Aguilar[1], and W. Vasconcelos[2]

[1]IIIA, Artificial Intelligence Research Institute
CSIC, Spanish Research Council
Campus UAB, 08193 Bellaterra, Spain
{andres,jar}@iiia.csic.es

[2]Dept. of Computing Science
University of Aberdeen
Aberdeen AB24 3UE, UK
wvasconcelos@acm.org

Abstract. Norms, that is, obligations, prohibitions and permissions, are useful abstractions to facilitate coordination in open, heterogeneous multi-agent systems. We observe a lack of distributed architectures and non-centralised computational models for norms. We propose a model, *viz.*, normative structures, to regulate the behaviour of autonomous agents taking part in simultaneous and possibly related activities within a multi-agent system. This artifact allows the propagation of normative positions (that is, the obligations, prohibitions and permissions associated to individual agents) as a consequence of agents' actions. Within a normative structure, conflicts may arise – one same action can be simultaneosely forbidden and obliged/permitted. This is due to the concurrent and dynamic nature of agents' interactions in a multi-agent system. However, ensuring conflict freedom of normative structures at design time is computationally intractable, and thus real-time conflict resolution is required: our architecture support the distributed management of normative positions, including conflict detection and resolution.

1 Introduction

An essential characteristic of open, heterogeneous multi-agent systems (MASs) is that agents' interactions are regulated to comply with the conventions of the system. Norms, that is, obligations, prohibitions and permissions, can be used to represent such conventions and hence as a means to regulate the observable behaviour of agents [3,18]. There are many contributions on the subject of norms from sociologists, philosophers and logicians (*e.g.*, [10,18]). Recently, proposals for computational realisations of normative models have been presented. Some of them operate in a centralised manner (*e.g.* [5,9,13]) which creates bottlenecks and single points-of-failure. Others (*e.g.* [3,12]), although distributed, aim at the regulation of communication between agents without taking into account that some of the normative positions (*i.e.*, their permissions, prohibitions and obligations) generated as a result of agent interaction may also affect other agents not involved in the communication.

The class of MASs we envisage consists of multiple, simultaneous and possibly related agent interactions, or *activities*. Each agent may simultaneously participate in several activities, and may change from one activity to another.

An agent's actions within one activity may have consequences – These are captured as normative positions that define, influence or constrain the agent's future behaviour. For instance, a buyer agent who ran out of credit may be forbidden from making further offers, or a seller agent is obliged to deliver the goods after closing a deal. Within a MAS normative conflicts may arise due to the dynamic nature of the MAS and simultaneous agents' actions. A normative conflict arises, for instance, when an action is simultaneously prohibited and obliged. Such conflicts ought to be identified and resolved. This analysis of conflicts can be carried out in each activity. However, ensuring conflict-freedom on a network of agent conversations (or activities) at design time is computationally intractable as shown in [7].

We propose means to handle conflicting normative positions in open and regulated MASs in a distributed manner. In realistic settings run-time conflict detection and resolution is required. Hence, we require a tractable algorithm for conflict resolution along the lines of the one presented in [7]. The only modification required for that algorithm is that it should return a list of updates (or *normative commands*), that is, the norms to be added and removed, instead of the resulting set of norms obtained from the updates.

We need an architecture to incorporate the previously mentioned algorithm. Among other features, we require our architecture to be distributed, regulated, open, and heterogeneous. These features are included in other architectures such as AMELI [3]. However, the propagation of normative positions to several agents or to an agent not directly involved in the interaction and the resolution of normative conflicts has not yet been addressed.

We thus propose an extension of the architecture presented in [3] fulfilling these features. We extend AMELI by including a new type of agent, *viz.*, the *normative managers*, also adding interaction protocols with this new type of agent, allowing for a novel conceptual differentiation of administrative (or "internal") agents. Thus, the main contribution of the paper is a distributed architecture to regulate the behaviour of autonomous agents and manage normative aspects of a MAS, including the propagation of normative positions to different conversations and the resolution of normative conflicts.

This paper is organised as follows. In Section 2 we present a scenario to illustrate and motivate our approach. Normative structures are introduced in Section 3. Section 4 presents our distributed architecture and, in Section 5, we comment on related work. Finally, we draw conclusions and report on future work in Section 6.

2 Scenario

We make use of a contract scenario in which companies come together at an online marketplace to negotiate and sign contracts in order to get certain tasks done. The overall transaction procedure may be organised as five distributed activities, represented as nodes in the diagram in Figure 1. The activities involve different participants whose behaviour is coordinated through protocols.

After registering at the marketplace, clients and suppliers get together in an activity where they negotiate the terms of their contract, *i.e.* actions to be performed, prices, deadlines and other details. The client will then participate in a *payment* activity, verifying his credit-worthiness and instructing his bank to transfer the correct amount of money. The supplier in the meantime will delegate to specialised employees the actions to be performed in the *work* activity. Finally, agents can leave the marketplace conforming to a predetermined *exit* protocol. The marketplace accountant participates in most of the activities as a trusted provider of auditing tools.

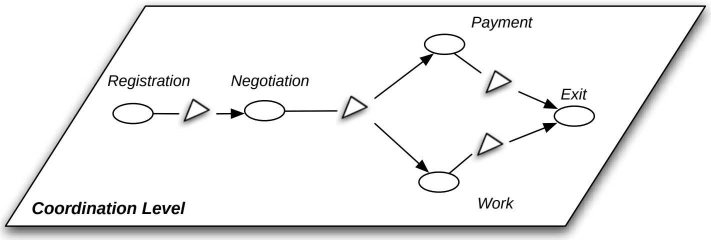


Fig. 1: Activity Structure of the Scenario

3 Normative Structure

We address a class of MASs in which interactions are carried out via illocutionary speech acts [14] exchanged among participating agents, along the lines of agent communication languages such as FIPA-ACL [6]. In these MASs, agents interact according to protocols which are naturally distributed. We observe that in some realistic scenarios, speech acts in a protocol may have an effect on other protocols. Certain actions bring about changes in the *normative positions* of agents – their "social burden": what each agent is permitted, obliged and forbidden to do. We use the term normative command to refer to the addition or removal of a normative position. Henceforth we shall refer to the application of a normative command as the addition or removal of a given normative position. Occurrences of normative positions in one protocol may also have consequences for other protocols.

We propose to extend the notion of MAS, regulated by protocols, with an extra layer called *normative structure* (NS). This layer consists of normative scenes, which represent the normative state, i.e. the set of illocutions uttered and normative positions, of the agents participating in a given activity, and normative transitions, which specifies by means of a rule the conditions under which some normative positions are to be generated or removed in the given normative scenes. The formal definition of normative structure is presented in [7], and here we informally discuss it.

Fig. 2 shows an example of how a normative structure relates with the coordination level. A normative transition is specified between the negotiation and payment activities denoting that there is a rule that may be activated with the state of negotiation activity and that may modify the state of the payment activity. In our example, the rule would be that whenever a client accepts an offer of a supplier, an obligation on the former to pay the latter is created in the payment activity. The rule connecting the payment and the work activity would specify that whenever a client fulfils its payment obligation, an obligation on the worker to complete the contracted task is generated in the work activity.

We are concerned with the propagation and distribution of normative positions within a network of distributed, normative scenes as a consequence of agents' actions. In [7] the formal semantics of NSs was defined via a mapping to Coloured Petri Nets. Conflicts may arise after the addition of new formulae. Hence, if a new norm does not generate any conflict then it can be directly added. If a conflict arises, the algorithm presented in [11] is used to decide whether to ignore the new normative position or to remove the conflicting ones.

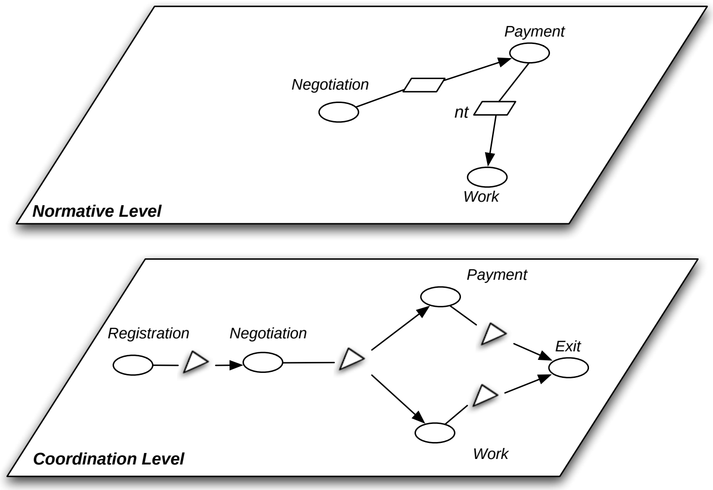


Fig. 2: Normative Structure and Coordination Level

4 Proposed Distributed Architecture

We propose an architecture to address the regulation of the behaviour of autonomous agents and the management of the normative state(s) of the MASs, including the propagation of normative positions and the resolution of normative conflicts. We assume the existence of a set of agents that interact in order to pursue their goals – we do not have control on these agents' internal functioning, nor can we anticipate it. We require the following features of our architecture:

Regulated The main goal of our architecture is to restrict the effects of agent behaviour in the specified conditions without hindering the autonomy of external agents.

Open Instead of reprogramming the MAS for each set of external agents, we advocate persistent, longer-lasting MASs where agents can join and leave them. However, agents' movements may be restricted in certain circumstances.

Heterogeneous We leave to each agent programmer the decision of which agent architecture include in each external agent. We make no assumption concerning how agents are implemented.

Mediatory As we do not control external agents internal functioning, in order to avoid undesired or unanticipated interactions, our architecture should work as a "filter" of messages between agents.

Distributed To provide the means for implementing large regulated MAS, we require our architecture to be distributed in a network and therefore spreading and alleviating the workload and the message traffic.

Norm propagative Although being distributed, agent interactions are not isolated and agent behaviour may have effects, in the form of addition or removal of normative positions, in later interactions possibly involving different agents.

Conflict Resolutive Some conflicts may arise due to normative positions being generated as result of agent's behaviour. Since ensuring a conflict-free MAS at design time is computationally intractable, we require that resolution of normative conflicts would be applied by the MAS. This approach promotes consistency since there is a unique, valid normative state established by the system instead of a lot of different state versions due to a conflict resolution at agent's level.

To accomplish these requirements, we extend AMELI, the architecture presented in [3]. That architecture is divided in three layers:

Autonomous agent layer It is formed by the set of external agents taking part in the MAS.

Social layer An infrastructure that mediates and facilitates agents' interactions while enforcing MAS rules.

Communication layer In charge of providing a reliable and orderly transport service.

External agents intending to communicate with other external agents need to redirect their messages through the social layer which is in charge of forwarding the messages (attempts of communication) to the communication layer. In specified conditions, erroneous or illicit messages may be ignored by the social layer in order to prevent them from arriving at their addressees.

The social layer presented in [3] is a multi-agent system itself and the agents belonging to it are called *internal agents*. We propose to extend this architecture by including a new type of agent , the normative manager (NM_1 to NM_p in fig. 3), and by adding protocols to accommodate this kind of agent. We call AMELI$^+$ the resulting architecture.

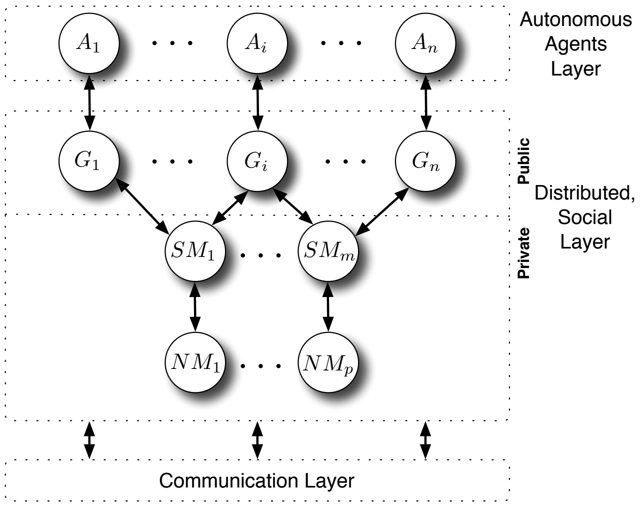


Fig. 3: AMELI$^+$architecture

In AMELI$^+$, internal (administrative) agents are of one of the following types:

Governor (G) Internal agent representing an external agent, that is, maintaining and informing about its social state, deciding or forwarding whether an attempt from its external agent is valid. One per external agent.

Scene Manager (SM) Internal agent maintaining the state of the activity[1], deciding whether an attempt to communicate is valid, notifying any changes to normative managers and resolving conflicts.

Normative Manager (NM) This new type of internal agent receives normative commands and may fire one or more normative transition rules.

In principle, only one NM is needed if it manages all the normative transition rules. However, in order to build large MAS and avoid bottlenecks, we propose the distribution of rules into several NMs.

To choose the granularity of the normative layer, i.e. to choose from one single NM to one NM per normative transition, is an important design decision that we leave for the MAS designers. After choosing the granularity, the NMs are assigned to handle a possibly unary set of normative transitions. Recall that each normative transition includes a rule. The SMs involved in the firing of the rules are given a reference to the NM that manages the rule, i.e. its address or identifier depending on the communication layer. External agents may join and leave activities, always following the conventions of the activities. In these cases, its governor registers (or deregisters) with the SM of that scene.

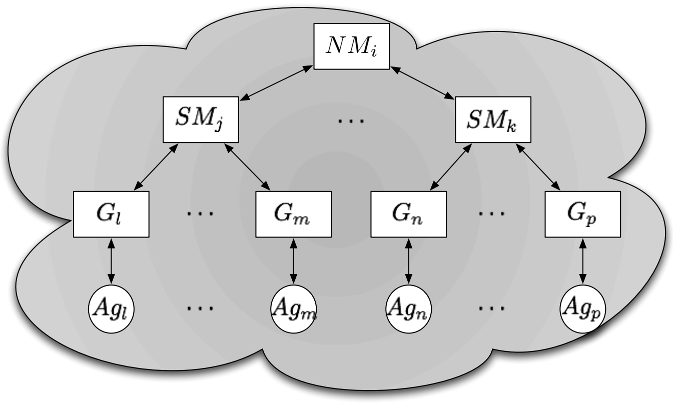


Fig. 4: Channels involved in the activation of a rule

4.1 Social Layer Protocols

Fig. 4 shows the communication within the social layer – it only occurs along the following types of channels:

Agent / Governor This type of channel is used by the external agents sending messages to their respective governors to request information or to request a message to be delivered to another external agent (following the norms of the MAS). Governors use this type of channel to inform their agents about new normative positions generated.

Governor / Scene Manager Governors use this type of channel to propagate unresolved attempts to communicate or normative commands generated as a result of such attempts. SMs use this type of channel to inform governors in their scenes about new normative commands generated as a result of attempts to communicate or conflict resolution.

[1] Hereafter, activities are also referred to as scenes following the nomenclature of AMELI.

Scene Manager / Normative Manager This type of channel is used by SMs to propagate normative commands that NMs may need to receive and the ones resulting from conflict resolution. NMs use this channel to send normative commands generated by the application of normative transition rules.

Fig. 5 shows an enactment of a MAS in our architecture. Agents send attempts to governors (messages 1, 4 and 7) who, after finding out the normative commands attempts generate, propagate the new normative commands to SM_{s1} and SM_{s2} (messages 2, 5 and 8) who, in turn, propagate them to the NM (messages 3, 6 and 9). As a normative transition rule is fired in the NM, a normative command is sent to SM_{s3} (message 10). After resolving any conflicts, SM_{s3} sends the new normative commands to all the involved governors (messages 11 and 11′) who, in turn, send them to their represented agents (messages 12 and 12′).

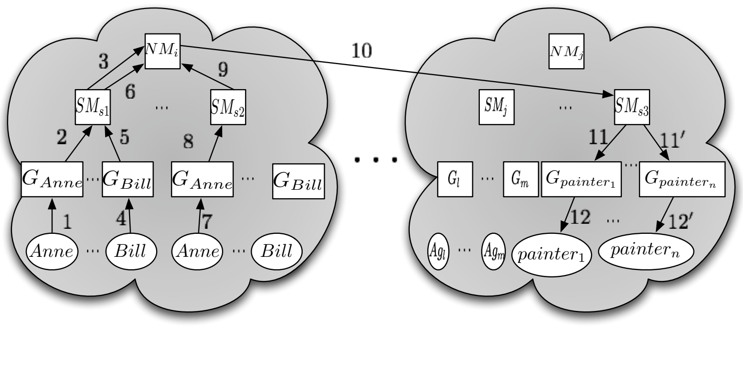


Fig. 5: Enactment of a normative transition rule

As the figure of the previous example shows, our architecture propagates attempts to communicate (and their effects) from agents (shown on the bottom of Fig 5) to the NMs (shown at the top of the figure). NMs receive events from several SMs whose managed state may be arbitrarily large. Since NMs only need the normative commands that may cause any of its rules to fire, NMs subscribe only to the type of normative commands they are supposed to monitor. For instance, if a rule needs to check whether there exists a prohibition to paint in a scene $work1$ and whether there exists the obligation of informing about the completion of the painting job, then the NM will subscribe to all the normative commands adding or removing prohibitions to paint in scene work1 as well as all normative commands managing obligations to inform about the completion of the painting job.

In the following algorithms, Δ refers to essential information for the execution of the MAS, i.e. a portion of the state of affairs of the MAS that each internal agent is managing. As introduced above, depending on the type of the internal agent, it manages a different portion of the state of affairs of the MAS, e.g. a governor keeps the social state of the agent, and a scene manager keeps the state of a given scene. These algorithms define the behaviour of internal agents and are applied whenever a message msg is sent by an agent (ag_i), a governor (g_i), a SM (sm_i) or a NM (nm_i) respectively.

When an external agent sends to its governor an attempt to communicate (messages 1, 4 and 7 in Fig. 5), the governor follows the algorithm of Fig. 6(a). This algorithm checks whether the attempt to communicate generates normative

```
algorithm $G\_process\_att(ag_i, msg)$
input $ag_i, msg$
output $\emptyset$
begin
01 $new\_cmmds := get\_norm\_cmmds(msg, \Delta)$
02 foreach $c \in new\_cmmds$ do
03     $\Delta := apply(c, \Delta)$
04     $sm := scene\_manager(c)$
05     $send(c, ag_i)$
06     $send(c, sm)$
07 endforeach
08 if $new\_cmmds = \emptyset$ then
09     $sm := scene\_manager(msg)$
10     $send(msg, sm)$
11 endif
end
```

(a) G response to an agent attempt

```
algorithm $NM\_process\_cmmd(sm_i, msg)$
input $sm_i, msg$
output $\emptyset$
begin
01 foreach $cmmd \in msg$ do
02     $\Delta := apply(cmmd, \Delta)$
03     $ncs := get\_RHS\_from\_fired\_rules(\Delta)$
04     foreach $c \in ncs$ do
05         $sm := scene\_manager(c)$
06         $send(c, sm)$
07     endforeach
08 foreach
end
```

(b) NM response to a command

```
algorithm $SM\_process\_att(g_i, msg)$
input $g_i, msg$
output $\emptyset$
begin
01 $new\_cmmds := get\_norm\_cmmds(msg, \Delta)$
02 foreach $c \in new\_cmmds$ do
03     $\Delta := apply(c, \Delta)$
04     $send(c, g_i)$
05     foreach $\langle nm, ev \rangle \in subscriptions$ do
06         if $unify(c, ev, \sigma)$ then
07         $send(c, nm)$
08         endif
09     endforeach
10 endforeach
11 if $new\_cmmds = \emptyset$ then
12     $s := scene(msg)$
13     $c := content(msg)$
14     $send(rejected(s, c), g_i)$
15 endif
end
```

(c) SM response to a forwarded attempt

```
algorithm $SM\_process\_cmmd(nm_i, msg)$
input $nm_i, msg$
output $\emptyset$
begin
01 $\Delta' := apply(msg, \Delta)$
02 if $inconsistent(\Delta')$ then
03     $msg := resolve\_conflicts(\Delta, msg)$
04 endif
05 foreach $cmmd \in msg$ do
06     $\Delta := apply(cmmd, \Delta)$
07     foreach $\langle nm, ev \rangle \in subscriptions$ do
08         if $unify(c, ev, \sigma)$ then
09             $send(c, nm)$
10         endif
11     endforeach
12     foreach $g \in governors(cmmd)$ do
13         $send(cmmd, g)$
14     endforeach
15 endforeach
end
```

(d) SM response to a command

Fig. 6. Internal Agents Algorithms

commands (line 1), i.e. it is accepted[2]. This check may vary depending on the type of specification and implementation of the scenes: e.g. using Finite State Machines (FSM), as in [3], or executing a set of rules, as in [9].

If the attempt generates normative commands (line 2), they are applied to the portion of the state of affairs the governor is currently managing creating a new partial state (line 3). These normative commands are sent to the external agent (line 5) and to the scene manager (messages 2, 5 and 8 in Fig. 5) in charge of the scene where the normative command should be applied (line 6). Otherwise, the attempt is forwarded to the SM of the scene the attempt was generated in (line 10).

If the governor accepts the attempt (after the check of line 1), it sends the SM a notification.The SM then applies the normative command received and forwards it to the NMs subscribed to that event (messages 3, 6 and 9 in Fig. 5).

However, if the governor does not take a decision, i.e. normative commands are not generated, the governor sends the attempt to the SM who should decide whether it is valid or not by following the algorithm of Fig. 6(c). This algorithm,

[2] In our approach, an ignored attempt would not generate any normative command.

like the one in Fig. 6(a), checks whether the received attempt generates normative commands in the current scene state, i.e. the portion of the state of affairs referring to that scene (line 1). If this is the case (line 2), they are applied to the current state of the scene (line 3) and forwarded to the governor that sent the attempt (line 4) and to the NMs subscribed to that normative commands (line 7). Otherwise (line 11), a message informing that the attempt has been rejected is sent to the governor mentioned (line 14).

In both cases, if the attempt is accepted then the normative manager is notified and it follows the algorithm of Fig. 6(b) in order to decide if it is necessary to send new normative commands to other scene managers. This algorithm processes each normative command received (line 1) by applying it to the state of the NM (line 2) and checking which normative transition rules are fired and obtaining the normative commands generated (line 3). Each of them are propagated to the SM of the scene appearing in the normative command (line 6, message 10 in Fig. 5).

If normative commands are generated, SMs receive them from the normative manager in order to resolve possible conflicts and propagate them to the appropriate governors. In this case, the SMs execute the algorithm of Fig. 6(d). This algorithm applies the normative command received on the scene state creating a temporary state for conflict checking (line 1), then checks if the new normative command would raise an inconsistency (line 2). If this is the case, it applies the conflict resolution algorithm presented in [7], returning the set of normative commands needed to resolve the conflict (line 3). Each normative command caused by the message sent by the NM or by conflict resolution, is applied to the scene state (line 6) and it is sent to the subscribed NMs (lines 7-11) and to the governors (messages 11 and 11' in Fig. 5) of the agents appearing in the normative command (lines 12-14).

NMs are notified about the resolution of possible conflicts in order to check if the new normative commands fire normative transition rules. If NMs receive this notification, they follow again the algorithm of Fig. 6(b) as explained above. When governors are notified by a SM about new normative commands, they apply the normative command received to the normative state of the agent and notify to its agent about the new normative command (messages 12 and 12' in Fig. 5).

In our approach, conflict resolution is applied at the SM level requiring all normative commands generated by a NM to pass through a SM who resolves conflicts and routes them. This feature is justified because SMs are the only agents who have a full representation of a scene and know the agents are participating in it and which role they are enacting. For example, if a prohibition for all painters to paint arrives at the work activity, a SM will forward this prohibition to the governors of the agents participating in that activity with the painter role and to the governors of all the new painters that join that activity while the prohibition is active. An alternative approach is to apply conflict resolution at the level of governor agents, curtailing some of the normative positions of its associated external agent. However, this type of conflict resolution is more limited since a governor only maintains the normative state of an agent. For example, a case that cannot be resolved with this approach is when all agents enacting a role are simultaneously prohibited and obliged to do something, i.e. when more than one agent is involved in the conflict.

Another approach would be if governors became the only managers of normative positions; in this case they would need to be aware of all normative positions that may affect its agent in the future, i.e. they would have to maintain all the normative positions affecting any of the roles that its agent may enact in every existing scene. For instance, a governor of an agent that is not yet enacting a painter role would also need to receive the normative positions that now applies to that role even if the agent is not in that scene or is enacting that role yet. This approach does not help with scalability since a large MAS with various scenes may generate a very large quantity of normative positions affecting agents in the future by the mere fact of their entering the MAS.

5 Related Work

The subject of norms has been studied widely in the literature (*e.g.*, [18,16,15]), and, more recently, much attention is being paid to more pragmatic and implementational aspects of norms, that is, how norms can be given a computational interpretation and how norms can be factored in the design and execution of MASs (e.g. [1,2,5,9,8]).

However, not much work has addressed the management of norms and reasoning about them in a distributed manner. Despite the fact that in [4,12] two languages are presented for the distributed enforcement of norms in MAS, in both works each agent has a local message interface that forwards legal messages according to a set of norms. Since these interfaces are local to each agent, norms can only be expressed in terms of actions of that agent. This is a serious disadvantage, *e.g.* when one needs to activate an obligation to one agent due to a certain message of another agent.

In [17] the authors propose a multi-agent architecture for policy monitoring, compliance checking and enforcement in virtual organisations (VOs). Their approach also uses a notion of hierarchical enforcement, i.e. the parent assimilates summarised event streams from multiple agents and may initiate further action on the subordinate agents. Depending on its policies, a parent can override the functioning of its children by changing their policies. Instead of considering any notion similar to our scene (multi-agent protocol where the number of participants may vary) and assigning an agent exclusively dedicated to the management of one scene, they assign another participant in the VO as parent of a set of agents. Although the parent would receive only the events it needs to monitor, it may receive them from *all* the interactions their children are engaging in. This can be a disadvantage when the number of interactions is large converting the parents in bottlenecks. Although they mention that conflict resolution may be accomplished with their architecture, they leave this feature to the VO agent thus centralising the conflict resolution in each VO. This can also be a disadvantage when the number of interactions is large since the VO agent has to resolve all the possible conflicts. This would require either all the events flowing through the VO agent or the VO agent monitoring the state of the whole VO in order to detect and resolve conflicts. The main theoretical restriction in their approach is that all the agents involved in a change in a policy must share a common parent in the hierarchy of the VO. In an e-commerce example, when a buyer accepts a deal an obligation to supply the purchased item should be added to the seller. However, as they are different parties, their only common parent is the VO agent converting the latter in a bottleneck in large e-commerce scenarios.

6 Conclusions and Future Work

We base the architecture presented in this paper in our proposal of normative structure and conflict resolution of [7]. The notion of normative structure is useful because it allows the separation of normative and procedural concerns. We notice that the algorithm presented in that paper is also amenable to the resolution of normative conflicts in a distributed manner.

The main contribution of this paper is an architecture for the management of norms in a distributed manner. As a result of the partial enactment of protocols in diverse scenes, normative positions generated in different scenes can be used to regulate the behaviour of agents not directly involved in previous interactions. Furthermore, conflict resolution is applied at a scene level meaning that resolution criteria involving more than one agent are now possible.

We want to extend normative structures [7], as we use them in our architecture, along several directions: (1) to handle constraints as part of the norm language, in particular constraints related with the notion of time; (2) to capture in the conflict resolution algorithm different semantics relating the deontic notions by supporting different axiomations (*e.g.*, relative strength of prohibition versus obligation, default deontic notions, deontic inconsistencies, etc.).

We also intend to use analysis techniques for Coloured Petri-Nets (CPNs) in order to characterise classes of CPNs (*e.g.*, acyclic, symmetric, etc.) corresponding to families of Normative Structures that are susceptible to tractable off-line conflict detection. The combination of these techniques along with our online conflict resolution mechanisms is intended to endow MAS designers with the ability to incorporate norms into their systems in a principled way.

Acknowledgements – This work was partially funded by the Spanish Education and Science Ministry as part of the projects TIN2006-15662-C02-01 and 2006-5-0I-099. García-Camino enjoys an I3P grant from the Spanish National Research Council (CSIC).

References

1. A. Artikis, L. Kamara, J. Pitt, and M. Sergot. A Protocol for Resource Sharing in Norm-Governed Ad Hoc Networks. In *Declarative Agent Languages and Technologies II*, volume 3476 (LNCS). Springer-Verlag, 2005.

2. S. Cranefield. A Rule Language for Modelling and Monitoring Social Expectations in Multi-Agent Systems. Technical Report 2005/01, University of Otago, 2005.
3. M. Esteva, B. Rosell, J. A. Rodríguez-Aguilar, and J. L. Arcos. AMELI: An agent-based middleware for electronic institutions. In *Procs of 3rd Int'l Conf on Autonomous Agents and Multiagent Systems (AAMAS'04)*, pages 236–243, 2004.
4. M. Esteva, W. Vasconcelos, C. Sierra, and J. A. Rodríguez-Aguilar. Norm consistency in electronic institutions. In *XVII Brazilian Symposium on Artificial Intelligence - SBIA'04*, volume 3171 (LNAI), pages 494–505. Springer-Verlag, 2004.
5. N. Fornara, F. Viganò, and M. Colombetti. An Event Driven Approach to Norms in Artificial Institutions. In *AAMAS05 Workshop: Agents, Norms and Institutions for Regulated Multiagent Systems (ANI@REM)*, Utrecht, 2005.
6. Foundation for Intelligent Physical Agents (FIPA). FIPA-ACL: Message Structure Specification, December 2002.
7. D. Gaertner, A. García-Camino, P. Noriega, J. A. Rodríguez-Aguilar, and W. Vasconcelos. Distributed Norm Management in Regulated Multi-agent Systems. In *Procs of 6th Int'l Conf on Autonomous Agents and Multiagent Systems (AAMAS'07)*, pages 624–631, Hawai'i, May 2007.
8. A. García-Camino, P. Noriega, and J. A. Rodríguez-Aguilar. Implementing Norms in Electronic Institutions. In *Procs of 4th Int'l Conf on Autonomous Agents and Multiagent Systems (AAMAS'05)*, pages 667–673, Utrecht, July 2005.
9. A. García-Camino, J.-A. Rodríguez-Aguilar, C. Sierra, and W. Vasconcelos. A Distributed Architecture for Norm-Aware Agent Societies. In *Decl. Agent Languages and Technologies III*, volume 3904 (LNAI), pages 89–105. Springer, 2006.
10. J. Habermas. *The Theory of Communication Action, Volume One, Reason and the Rationalization of Society.* Beacon Press, 1984.
11. M. J. Kollingbaum, W. W. Vasconcelos, A. García-Camino, and T. J. Norman. Conflict resolution in norm-regulated environments via uni cation and constraints. In *Fifth International Workshop on Declarative Agent Languages and Technologies (DALT 2007)*, Hawai'i, May 2007.
12. N. Minsky. Law Governed Interaction (LGI): A Distributed Coordination and Control Mechanism (An Introduction, and a Reference Manual). Technical report, Rutgers University, 2005.
13. A. Ricci and M. Viroli. Coordination Artifacts: A Unifying Abstraction for Engineering Environment-Mediated Coordination in MAS. *Informatica*, 29:433–443, 2005.
14. J. Searle. *Speech Acts, An Essay in the Philosophy of Language.* Cambridge University Press, 1969.
15. M. Sergot. A Computational Theory of Normative Positions. *ACM Trans. Comput. Logic*, 2(4):581–622, 2001.
16. Y. Shoham and M. Tennenholtz. On Social Laws for Artificial Agent Societies: Off-line Design. *Artificial Intelligence*, 73(1-2):231–252, 1995.
17. Y. B. Udupi and M. P. Singh. Multiagent policy architecture for virtual bussiness organizations. In *Proceedings of the IEEE International Conference on Services Computing (SCC)*, September 2006.
18. G. H. von Wright. *Norm and Action: A Logical Inquiry.* Routledge and Kegan Paul, London, 1963.

Dynamic Institutions for Teamwork

Mario Gómez and Enric Plaza
mgomez@csd.abdn.ac.uk, enric@iiia.csic.es

[1] Department of Computing Science, University of Aberdeen
[2] Artificial Intelligence Research Institute, Spanish National Research Council

Abstract. We present a dynamic electronic institutions approach for teamwork. In this model, agent teams are designed and deployed on-the-fly so as to met the requirements of the task at hand. The result is a new form of electronic institution that is created dynamically out of existing components. We also introduce a case-based learning mechanism to form new agent teams by reusing complete or partial team-designs used in the past for similar problems.

1 Introduction

Cooperative problem solving (CPS) is a form of social interaction in which a group of agents work together to achieve a common goal. Several models have been proposed to account for this form of interaction from different perspectives: distributed artificial intelligence, economics, philosophy, organization science and social sciences. From the artificial intelligence perspective there are two main approaches to cooperation: a micro-level –agent-centered– view, which is focused on the internal architecture or the decision-making model of individual agents, and a macro-level –social– view, which is focused on the societal and organizational aspects of cooperation. Most of the models and theories of cooperation proposed for MAS have adopted the agent-centered view, typically based on some refinement of the beliefs, desires and intentions (BDI) paradigm, such as the *Joint Intentions* model [12] and the *SharedPlans* theory [9].

Some of the most challenging issues faced by the MAS community are related to the creation of open MAS [11]. Closed systems are typically designed by one team for one homogeneous environment, while in open MAS the participants (both human and software agents) are unknown beforehand, may change over time and may be developed by different parties. Therefore, those infrastructures that adopt a social view on cooperation seem more appropriate that those adopting a micro-level view, for the former do not enforce a particular agent architecture.

In addition, some aspects of complex system development become more difficult by adopting an agent-centered approach: since agents are autonomous, the patterns and the effects of their interactions are uncertain, and it is extremely difficult to predict the behavior of the overall system based on its constituent components, because of the strong possibility of emergent behavior [10]. These problems can be circumvented by restraining interactions and imposing preset organizational structures, which are characteristic of the social view. The Civil Agent Societies framework [2] and the electronic institutions formalism [13, 15, 5] are good examples of this approach; many others can be found in the COIN international workshop series on coordination, organizations, institutions, and norms [1].

An electronic institution (or e-Institution) refers to a sort of "virtual place" that is designed to support and facilitate certain goals to the human and software agents concurring to that place by establishing explicit conventions. Since these goals are achieved by means of the interaction of agents, an e-institution provides the social mediation layer required by agents to achieve a successful interaction: interaction protocols, shared ontologies, communication languages and social behavior rules.

All in all, a main issue arises when trying to use preset organizational structures to operationalize CPS: the need for different team structures to deal with different problem types. The e-institutions formalism was originally conceived to formalize and implement static organizations of agents; therefore, at first glance it seems inadequate to use such a formalism for flexible teamwork. In this paper we introduce a proposal that uses the e-institution formalism in a novel way: dynamic institutions for teamwork. These institutions are created on-the-fly out of existing components that capture the communication and coordination aspects of teamwork.

The paper is structured as follows: Section §2 puts our institutional model of teamwork in context by introducing the framework this model is part of, §3 describes our proposal to model teamwork using the e-Institutions formalism [4], §4 describes a technique to improve team design by using case-based reasoning, and finally, §5 summarizes our contributions.

2 The ORCAS framework

In this paper we present an institutional approach to CPS that is part of the ORCAS framework for developing and deploying cooperative MAS [6]. The main contributions of this framework are:

- An *agent capability description language* (ACDL) that supports all the activities required to cooperate in open environments, from the discovery and invocation of capabilities, to their composition and coordination.
- A model of CPS that is driven by the specification of requirements for every instance of a problem to be solved
- An agent platform for developing and deploying cooperative MAS in open environments

Figure 1 depicts the main components of the ORCAS ACDL, and the activities enabled by each component. An agent provides one or more capabilities. There are two types of capability: *skill* and *task-decomposer*. Skills are primitive, non decomposable capabilities, while task-decomposers decompose a problem (a task) into more elementary problems (subtasks), so as to solve complex problems that primitive capabilities cannot accomplish alone. The *knowledge-level description* of a capability specifies features such as the input, output, preconditions, and postconditions, which can be used by middle agents to discover and compose capabilities. However, in order to interact with the provider of a given capability (to invoke the capability, pass input data and get the results back), the requester agent must use an interaction protocol that is compatible with the capability of interest and is supported by its provider. In ORCAS this interaction protocol is referred to as the *communication* of a capability (take note that the same capability could be invoked using different protocols). Finally, the information required to coordinate multiple agents that are cooperating to solve a problem together is specified by the *operational description* of a task decomposer, which describes the control flow among subtasks (sequencing, parallelism, choices, etc.) in terms of agent roles.

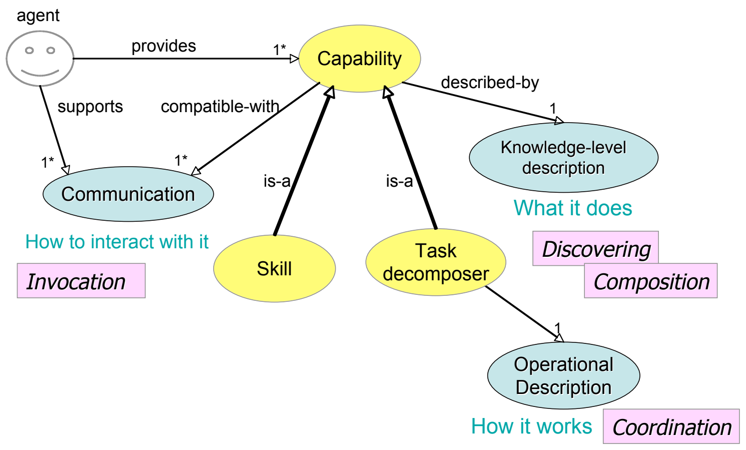


Fig. 1. Overview of the ORCAS ACDL

The ORCAS platform provides all the infrastructure required by agents to successfully cooperate according to the ORCAS model for CPS, which is depicted in Figure 2. The *problem specification* process produces a specification of problem requirements to be met by a team, including a description of the application domain (a collection of domain models) and the problem data to be used during teamwork. The *team design* process uses the problem requirements to build a *task-configuration*, which is a knowledge-level specification of: (1) the tasks to solve, (2) the capabilities to apply, and (3) the domain knowledge required by a team of agents in order to solve a given problem according to its specific requirements. The resulting task-configuration is used during *team formation* to allocate tasks and subtasks to agents, and to instruct agents on how solve the problem cooperatively. Finally, during *teamwork*, team members try to solve the problem together by following the instructions received during team formation, thus complying with the specific requirements of the problem at hand. To note that the ORCAS model for CPS should not be understood as a fixed sequence of steps, instead, we have implemented strategies that interleave team design and team formation with teamwork. These strategies enable the reconfiguration of agent teams dynamically so as to react to agent failure and other changes in the environment.

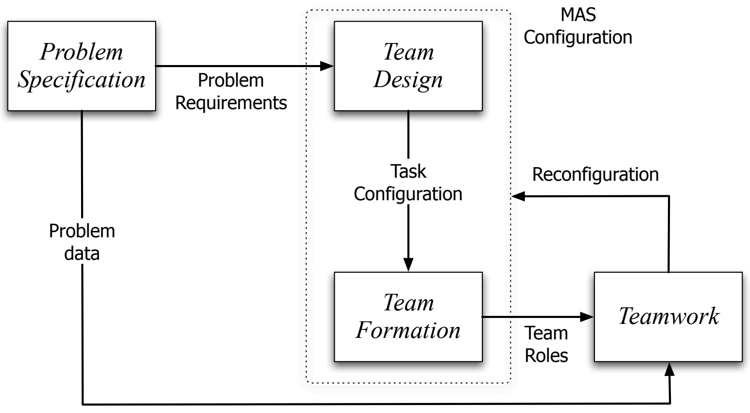


Fig. 2. The ORCAS model for the cooperative problem solving process.

It should be remarked that, within the ORCAS framework, the e-institutions formalism is used in two ways: on the one hand, we use concepts adapted from the ISLANDER e-institutions formalism [4, 3] for specifying some elements of the ORCAS ACDL (the communication and the operational description); on the other hand, the ORCAS agent platform is itself an e-institution that provides mediation services for both providers and requesters of problem solving capabilities to successfully cooperate.

The knowledge-level description of a capability and the mechanisms used in ORCAS to discover and compose capabilities (which are part of the team design process) have been described elsewhere [7]. The ORCAS agent platform is described in [8]. In this paper we focus on those aspects of the ORCAS ACDL that are based on the e-institutions formalism, namely the communication and the operational description, and how are these elements used to represent the interaction and coordination requirements of teamwork. These are the subjects of the following section.

3 Dynamic institutions for hierarchical teamwork

The ORCAS ACDL specifies the communication and operational description of capabilities using elements from the ISLANDER formalism in a novel way, so it seems appropriate to briefly review the main concepts of this formalism before describing their use in ORCAS:

1. *Agent roles:* agents are the players in an e-institution, interacting by the exchange of speech acts, whereas roles are standardized patterns of behavior required by agents playing part in given functional relationships.

2. *Dialogic framework:* determines the valid illocutions that can be exchanged among agents, including the vocabulary (ontology) and the agent communication language.
3. *Scenes:* a scene defines an interaction protocol among a set of agent roles, using the illocutions allowed by a given dialogic framework.
4. *Performative structure:* a network of connected scenes that captures the relationships among scenes; a performative structure constrains the paths agents can traverse to move from one scene to another, depending on the roles they are playing.

In ORCAS the specification of capabilities at the knowledge level enables the automated discovery and composition of capabilities, without taking into account neither the communication aspects required to invoke a capability, nor the operational aspects required to coordinate the behavior of several agents. These features are specified in the ORCAS ACDL adapting concepts from ISLANDER, as follows:

Communication: specifies one or several interaction protocols that can be used to interact with agent to invoke a given capability and get back the result of applying it. This feature is specified using the notion of *scene* taken from the e-institutions formalism.

Operational Description: specifies the control flow among the subtasks introduced by a task-decomposer, using a modified version of the *performative structure* concept from the e-institutions formalism.

A team in ORCAS is designed to solve a problem represented by a knowledge-level structure referred to as a *task-configuration* (the reader is referred to [7] for a more detailed description). Figure 3 shows an example of a task-configuration for a task called *Information-Search*. This task is decomposed into four tasks by the *Meta-search* task-decomposer: *Elaborate-query*, *Customize-query*, *Retrieve* and *Aggregate*, which is further decomposed by the *Aggregation* capability into two subtasks: *Elaborate-items* and *Aggregate-items*. The example includes some skills requiring domain knowledge: the *Query-expansion-with-thesaurus* requires a thesaurus (e.g. *MeSH*, a medical thesaurus), and the *Retrieval* and *Query-customization* skills require a description of information sources.

Any ORCAS team follows the hierarchical structure of a task-configuration, with one team-role per task. In particular, each team role includes the following elements: a team-role identifier (the same task could appear multiple times in the same task-configuration, so a unique team-role identifier is required), the identifier of a task to be solved, the identifier of a capability to apply, the domain knowledge to be used by the selected capability (if needed), and optionally, if the capability is a task decomposer, the information required to delegate subtasks to other team-members, which includes, for each subtask: the team member selected to play the task (or several agents in the case of tasks to be performed multiple times in parallel), a collection of reserve agents to use in case that the selected team member fails, and a communication protocol that is compatible with the selected capability and shared by both the agent assigned to the parent

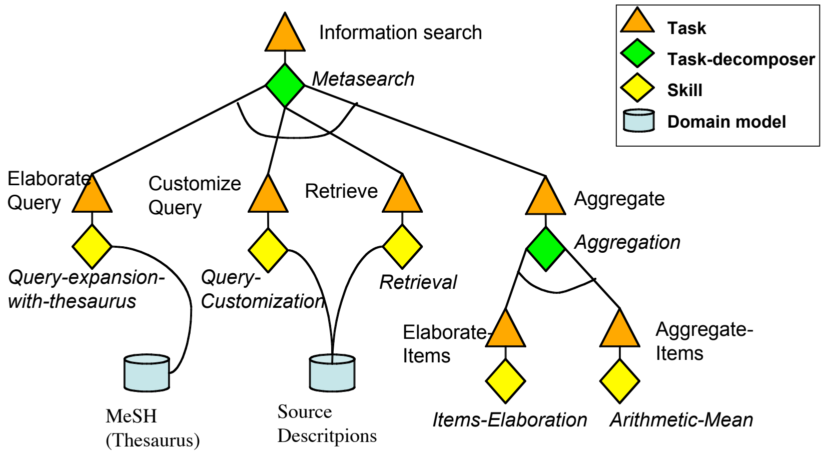
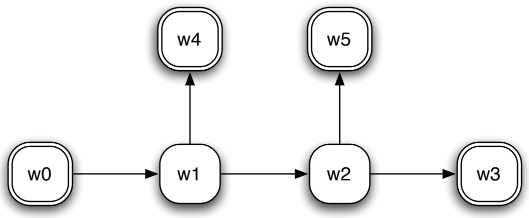


Fig. 3. Task-configuration example

task, and the agent or agents assigned to the subtask. Next subsections address, respectively, the specification of the communication and operational description of a capability in ORCAS.

3.1 Communication

Agent capabilities should be specified independently of other agents in order to maximize their reuse and facilitate their specification by third party agent developers. In the general case, agent developers do not know a priori the tasks that could be achieved by a particular capability, neither the domains they could be applied to. As a consequence, the team roles an agent could play using a capability are not known in advance, thus the scenes used to specify the communication requirements of an agent over certain capability cannot be specified in terms of specific team-roles, but in terms of abstract, generic problem solving roles. Since ORCAS teams are designed in terms of a hierarchical decomposition of tasks into subtasks, teamwork is organized as a hierarchy of team-roles.



1. request (?x Coordinator) (?y Operator) (perform ?team-role ?input)
2. agree !y !x (!team-role !Input)
3. inform !y !x (?team-role ?output)
4. refuse !y !x (!team-role !Input)
5. error !y !x! (team-role !Input)

Fig. 4. Example of a communication scene

Some positions within a team (team-roles) are bound to a task-decomposer, thus the agents playing those team-roles are responsible of delegating subtasks to other agents, receiving the results, and performing intermediate data processing between subtasks. In such an scenario, we establish an abstract communication model with two basic roles: *coordinator*, which is adopted by an agent willing to decompose a task into subtasks, and *operator*, which is adopted by the agent having to perform a task on demand, using the data provided by another agent that acts as coordinator of a top-level task

Figure 4 shows a scene depicting the communication requirements of an agent over a capability by using a typical request-inform protocol in terms of our two generic roles: *Coordinator* and *Operator*. Symbol *?* denotes a new bind for a variable, while *!* denotes a variable that has been already bound to a value.

3.2 Operational description

The operational description of a task decomposer is used to specify the coordination among agents in terms of the role-flow policy and the control flow among subtasks. Figure 5 depicts some of the control flow constructions allowed by a performative structure: (a) tasks performed consecutively, in sequence; (b) choice between alternative courses of action; (c) tasks performed in parallel; and (d) tasks that can be executed multiple times.

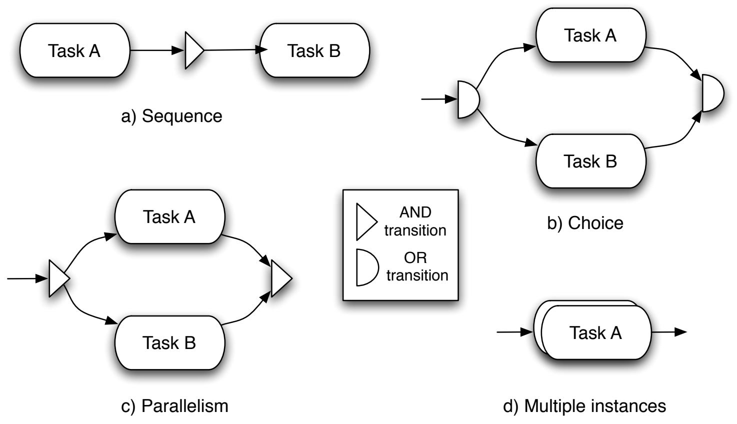


Fig. 5. Control flow among subtasks used in operational descriptions

In ORCAS the operational description of a task-decomposer is based on performative structures, with some distinctive features: as in the e-institutions formalism, each ORCAS scene within a performative structure must be instantiated by a communication protocol (except the *Start* and *End* scenes). However, in ORCAS the scenes within a performative structure are not instantiated beforehand; that is to say, they are not bound to a specific communication protocol. Instead, the scenes of an operational description are instantiated during team formation, using as a source the set of communication protocols shared by the agents having to interact.

After instantiation, each scene in an operational description corresponds to the communication required to solve a subtask, which implies an agent acting as coordinator invoking the capability provided by another agent acting as operator (or several operators in the case of multiple-instantiated tasks). The coordinator and the operators must use the same communication protocol in order to successfully communicate. Consequently, the instantiation of the scenes in an operational description is done using only those communication protocols shared by the agents involved in a scene. To note that team members are selected during team formation, and thus the set of shared communication protocols is not known until the team members are decided.

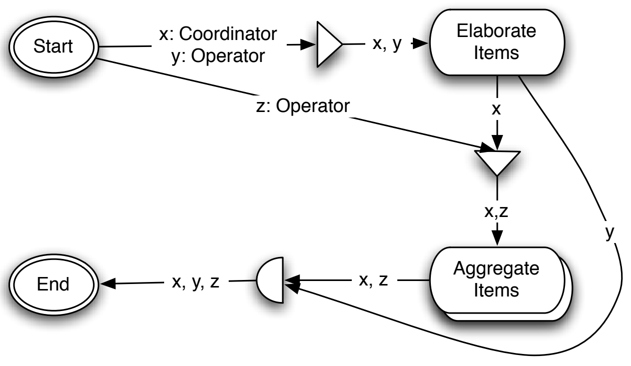


Fig. 6. Example of an operational description

Figure 6 shows an example of an operational description for a task-decomposer called Aggregation. This task-decomposer introduces two subtasks: Elaborate-items (EI) and Aggregate-items (AI). Thus, the operational description has two main scenes, one for each subtask, and three role variables: x is a coordinator role, to be played by the agent applying the task-decomposer; y and z are both operator roles; y participates in EI, and z participates AI. Notice that the coordinator (x) is the same in both scenes; it enters EI first and moves to AI only after EI ends.

Since each task-decomposer has an operational description, and the ORCAS organization of a team follows the hierarchical decomposition of tasks into subtasks that results of applying task-decomposers, we can model the operational description of a complete team as nested structure of operational descriptions.

Figure 7 depicts the operational description of a team. The top team-role, associated to the Information Search task, is bound to a task-decomposer (Meta-Serach) that introduces three subtasks: Customize Query, Retrieve and Aggregate. Therefore, the top team-role will follow an operational description that contains three scenes, one for each subtask. In addition, the last of these subtasks is bound to another task-decomposer, Aggregation, which in turn introduces a new

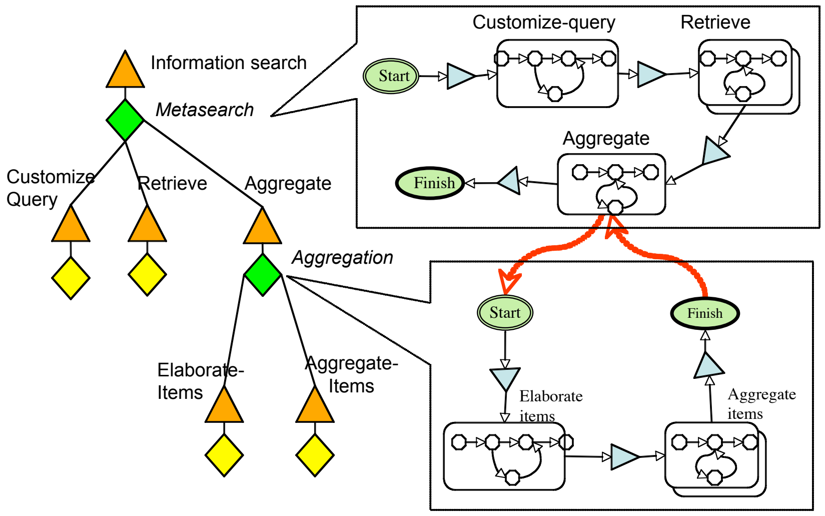


Fig. 7. Teamwork as a nested structure of operational descriptions

operational description. The new operational description is nested to the team leader's operational description, and has scenes, one for every subtask: Elaborate-Items and Aggregate-Items.

Teamwork follows the control flow and the communication scenes established by the nested structure of operational descriptions associated to task-decomposers (already instantiated during team formation). Each scene within an operational description refers to a communication protocol to be played by two agents, one applying a task-decomposer and playing the coordinator role, and one assigned to the corresponding subtask playing the operator role. When an agent playing an operator role has to apply itself a task-decomposer, it will follow the associated operational description playing itself the coordinator role. The execution of an operational description does not finish until all the nested operational descriptions are executed.

Each time a new team is formed according to a task-configuration, a new structure of nested operational descriptions is composed and their scenes instantiated. We regard this structure as a dynamic institution, since it is configured on-the-fly, out of the communication protocols and the operational descriptions supported by the selected team members.

4 Learning Team Designs

The ORCAS description of *team designs* allows that reasoning and learning processes may be applied to them by an agent capable of convening a team. For this purpose, an agent-oriented case-based reasoning (CBR) technique called CoopCA has been developed based on the notion of *compositional cases* [14]. In team design, a case is a pair (P, S), where the problem P is the specification of the *task* a team should be able to achieve and the solution S is the *task-configuration* of the team; a case is a compositional case when the solution is a configuration of components —as the ORCAS ACDL components in team design. Notice that a task basically specifies two conditions: the *assumptions* (those properties that are assumed to be true in the world) and *goals* (those properties that are to be achieved by the team).

Learning in CBR allows agents to solve much faster routine and easy tasks. An agent using CoopCA will store in its case base those team designs the agent has convened in the past. Since in a real environment regularities are common, an agent will encounter both routine tasks and novel tasks. When a new task that needs a team to be convened is equal or similar to tasks the agent has solved in the past using specific team designs, case-based reasoning will reuse the solutions of the past to fit the current situation: a previous team design will be used, possibly with a few alterations to adapt the new team to the differences (in assumptions and/or goals) between the old task and the current task.

When a new task is novel, in the sense that it is rather different from any other task previously solved by the agent, CoopCA is capable of achieving a new team design from scratch. However, CoopCA derives a new task-configuration in a search process that is guided by the past cases, and is able to find, for instance, that a specific *subtask* was solved in the past by a particular team, and will incorporate it as a *subteam* for that subtask in the overall team design. Thus, the agent playing the role of convener can learn from experience about the particular *team institutions* that achieve certain tasks. Notice that this learning is developed at the institutional level: the agent learns that a specific team institution is able to achieve a certain task — does not learn about the performance of a specific team with concrete agents performing the institution's roles.

Finally, CBR is used for *dynamic reconfiguration* when some event precludes the usability of the current team institution. For instance, imagine that and agent that was supposed to perform team-role R_j goes offline or refuses to satisfy its previous commitment; and imagine there is no other available agent capable of satisfying the requirements of that role: under this conditions the task associated to R_j could not be achieved and the task-configuration that shaped the current team is no longer viable. The CBR process however can continue its search process to find another task-configuration (if it exists) that achieves the same overall task. There is no need to stick to a fixed design when several possible solutions are available.

5 Conclusions

In this paper, we have presented a novel approach to teamwork specification using concepts adapted from the e-Institutions formalism. In this approach the communication and coordination aspects required for teamwork are reusable components that are used by agents to specify their problem solving capabilities. By doing so, middle agents such as brokers and matchmakers can reason about the communication and coordination aspects of individual agents to dynamically build an e-Institution that supports flexible teamwork.

We adapt the electronic formalism to handle the dynamics of teamwork. While e-institutions are supposed to be static structures characterized by a predefined network of scenes (a performative structure), we conceive teamwork as a dynamic institution that is build on the fly out of existing components: operational descriptions and communication protocols. The operational description of a task-decomposer describes the control flow among subtasks using a specific kind of performative structure in which the communication scenes are not instantiated beforehand. The instantiation of those scenes is done at runtime by selecting communication protocols that are shared by the agents involved in a given scene. The result is a hierarchical model of teamwork represented by nested performative structures instantiated and composed on-the-fly during team formation.

By adapting the e-Institutions formalism for teamwork, we expect to bring in some of the benefits of the social-approach in general, and the e-institutions approach in particular: promoting the development of agents by third parties by avoiding the imposition of a specific agent architecture (favors openness); increasing the degree of control over the global system behavior; and making the system more predictable, which in turn fosters trustiness.

Finally, by introducing a case-based reasoning approach to team design we have enabled a learning process that speeds up the configuration of new teams by reusing previous team designs for solving new problems

Acknowledgements

This research was sponsored by the Spanish Council for Scientific Research under the MID-CBR (TIN 2006-15140-C03-01) project.

References

1. O. Boissier, J. Padget, V. Dignum, G. Lindemann, E. Matson, S. Ossowski, J. Sichman, and J. Vaźquez-Salceda, editors. *Coordination, Organizations, Institutions, and Norms in Multi-Agent Systems*, volume 3913 of *Lecture Notes in Artificial Intelligence*. Springer, December 2006.
2. C. Dellarocas. Contractual Agent Societies. negotiated shared connote and social control in open multi-agent systems. In *Proceedings of the Workshop on Norms and Institutions in Multi-Agent Systems, ICMAS'02*, 2000.
3. M. Esteva. *Electronic Institutions: From Specification to Development*, volume 14 of *Monografies de l'Institut d'Investigació en Intel.ligència Artificial*. Spanish National Research Council, 2003.
4. M. Esteva, J. Padget, and C. Sierra. Formalizing a language for institutions and norms. In *Intelligent Agents VIII: Lecture Notes in Artificial Intelligence*, volume 2333 of *Lecture Notes in Artificial Intelligence*, pages 348–366. Springer-Verlag, 2002.
5. M. Esteva, J. A. Rodriguez, C. Sierra, P. Garcia, and J. L. Arcos. On the formal specifications of electronic institutions. In *Agent-mediated Electronic commerce. The European AgentLink Perspective*, volume 1991 of *Lecture Notes in Artificial Intelligence*, pages 126–147, 2001.

6. M. Gómez. *Open, Reusable and Configurable Multi-Agent Systems: A Knowledge-Modelling Approach*, volume 23 of *Monografies de l'Institut d'Investigació en Intel.ligència Artificial*. Spanish National Research Council, 2004.
7. M. Gómez and E. Plaza. Extending matchmaking to maximize capability reuse. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multi Agent Systems*, volume 1, 2004.
8. M. Gómez and E. Plaza. The ORCAS e-Institution: a Platform to Develop Open, Reusable and Configurable Multi-Agent Systems. *International Journal on Intelligent Control and Systems. Special Issue on Distributed Intelligent Systems*, juny 2007.
9. B. J. Grosz and S. Kraus. Collaborative plans for complex group action. *Artificial Intelligence*, 86(2):269–357, 1996.
10. N. R. Jennings. On-agent-based software engineering. *Artificial Intelligence*, 117:227–296, 2000.
11. M. Klein. The Challenge: Enabling Robust Open Multi-Agent Systems, 2000.
12. H. J. Levesque. On acting together. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, pages 94–99, 1990.
13. P. Noriega. *Agent-Mediated Auctions: The Fish-Market Metaphor.* PhD thesis, Universitat Autònoma de Barcelona, 1997.
14. E. Plaza. Cooperative reuse for compositional cases in multi-agent systems. *Lecture Notes in Computer Science*, 3620:382–396, 2005.
15. J. A. Rodríguez-Aguilar. *On the Design and Construction of Agent-mediated Electronic Institutions.* PhD thesis, Universitat Autnoma de Barcelona, 1997.

Coordination and sociability for intelligent virtual agents

Francisco Grimaldo, Miguel Lozano, Fernando Barber

Computer Science Dept., University of Valencia
Dr. Moliner 50, 46100 Burjassot (Valencia)
{francisco.grimaldo, miguel.lozano, fernando.barber}@uv.es

Abstract. This paper presents a multi-agent framework designed to simulate synthetic humans that properly balance task oriented and social behaviors. The work presented in this paper focuses on the social library integrated in BDI agents to provide socially acceptable decisions. We propose the use of ontologies to define the social relations within an artificial society and the use of a market based mechanism to reach sociability by means of task exchanges. The social model balances rationality, to control the global coordination of the group, and sociability, to simulate relations (e.g. friendliness) and reciprocity among agents. The multi-agent framework has been tested successfully in dynamic environments while simulating a virtual bar, where groups of waiters and customers can interact and finally display complex social behaviors (e.g. task passing, reciprocity, planned meetings).

1 Introduction

Multi-agent systems are sometimes referred to as societies of agents and provide an elegant and formal framework to animate synthetic humans. When designing such agents, the main concern has normally been with the decision-making mechanism, as it is the responsible for the actions that will be finally animated. Virtual actors normally operate in dynamic resource bounded contexts; thus, multi-agent simulations require group coordination, as self-interested agents easily come into conflicts due to the competition for the use of shared resources (i.e. objects in a virtual environment). These obstructions produce low quality animations where characters do not act realistically. Moreover, virtual humans represent roles in the scenario (e.g. a virtual guide, waiter, customer, etc.) and the social network formed by the relations among the members of the society should also be considered when animating their behaviors.

This paper presents a multi-agent simulation framework to produce good quality animations where the behavior of socially intelligent agents better imitates that of real humans. We aim at incorporating human style social reasoning in virtual characters. Therefore, we have developed a market based social model [15] which coordinates the activities of groups of virtual characters and incorporates social actions in the agent decision-making. Our approach is inspired in reciprocal task exchanges between agents [17] and uses ontologies to define the social relations within an artificial society. According with the main parameter of the model, that is *sociability*, the agents can balance their task-oriented behaviors (e.g. a virtual waiter should serve customers) and their social skills (e.g. negotiate with other waiters to gain access to a resource, assume external actions/favors, or simple chats).

The structure of the paper is as follows: in section 2 we describe briefly some previous literature on the field. In section 3 we present the multi-agent simulation framework and the main components of the social model. Section 4 describes an illustrative example modeled to test our framework. Lastly, section 5 summarizes the first results extracted and analyzes them.

2 Related work

Many interactive games and virtual communities put human users together with synthetic characters. In this context, some research has been done on the believability issues of virtual actors, usually centred on the interactions either between a human user and a single character [1] or among the synthetic characters themselves [21]. These interactive scenarios often present tasks to the participants that must be solved collaboratively [18]. Therefore, behavioral animation has broadly been tackled from the field of coordinated multi-agent systems (e.g. Generalized Partial Global Planning (GPGP) [8], the TAEMS framework [7] or the RETSINA system [10]). Moreover, task coordination has been applied to HSP-based (Heuristic Search Planning) virtual humans in [4] and [12] to adapt better to the dynamism of shared environments.

Social reasoning has also been extensively studied in multi-agent systems in order to incorporate social actions to cognitive agents [6]. As a result of these works, agent interaction models have evolved to social networks that try to imitate the social structures found in reality [14]. Social dependence networks in [20] allow agents to cooperate or to perform social exchanges attending to their dependence relations (i.e. social dependence and social power). Trust networks in [9] are used to define better delegation strategies by means of a contract net protocol and fuzzy cognitive representations of the other agents as well as of the dynamic environment. In preference networks, such as the one presented in this paper, agents express their preferences using utility functions and their attitude towards another agent is represented by the differential utilitarian importance they place on that agent's utility.

Semantic information can be of great value to the agents inhabiting a virtual world. As demonstrated in [13], the use of semantics associated to objects can enhance the interaction of virtual humans in complex environments. Environment-based approaches are also emerging to provide semantic interoperability among intelligent agents through the use of *coordination artifacts* [22]. Furthermore, ontologies are very useful to model the social relations between the agents involved in graphical and interactive simulations [16]. In MOISE+ [11], ontological concepts join roles with plans in a coherent organizational specification. Another example can be found in [5] where a functional ontology for reputation is proposed.

Although the results obtained by the previous approaches show realistic simulations for many task-oriented behaviors, synthetic characters should also display pure social behaviors (e.g. interchanging information with their partners or grouping and chatting with their friends). MAS-SOC [2] aims at creating a platform for multi-agent based social simulations with BDI agents, which is also our purpose. In this context, work is ongoing in order to incorporate social-reasoning mechanisms based on *exchange values* [19]. The multi-agent framework presented here is oriented to simulate socially intelligent agents able to balance their rationality and sociability, a key point to finally display high quality behavioral animations.

3 Multi-agent simulation framework

The multi-agent simulation framework presented in figure 1 has been developed over Jason [3], which allows the definition of BDI agents using an extended version of AgentSpeak(L). The animation system (virtual characters, motion tables, etc) is located at the 3D engine, which can run separately. The environment is handled by the *Semantic Layer*, which acts as an interface between the agent and the world. It is in charge of perceiving the state of the world and executing the actions requested by the agents, while ensuring the consistency of the *World Model*. Ontologies define the world knowledge base using two levels of representation: the *SVE Core Ontology* is a unique base ontology suitable for all virtual environments and it is extended by different *Domain Specific Ontologies* in order to model application-specific knowledge.[1]

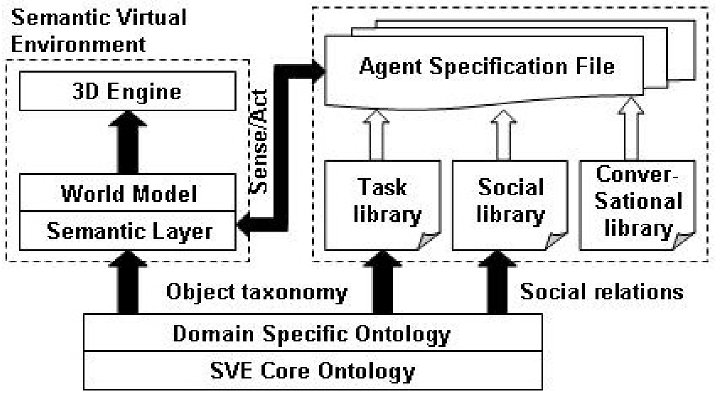


Fig. 1. Multi-agent simulation framework.

The agent decision-making is defined in the *Agent Specification File*. This file contains the initial beliefs as well as the set of plans that make up the agent's finite state machine. The *Task Library* contains the set of plans that sequence the actions needed to animate a task. For instance, a virtual waiter serving a coffee will go to the coffee machine to get the coffee and will give it to the customer afterwards. Here, modularity is guaranteed since the *Task library* can be changed depending on the environment and the roles being simulated. As stated above, only rational behaviors are not enough to simulate agent societies. Therefore, we have extended the ontologies to define the possible social relations among the agents of a society and we have included a *Social*

[1] See [13] for details on ontologies and their use to enhance agent-object interaction.

library to manage different types of situations. This library is based on an auction model and uses social welfare concepts to avoid conflicts and allow the agents to behave in a coordinated way. The *Social library* also incorporates a reciprocity mechanism to promote egalitarian social interactions. Finally, the *Conversational library* contains the set of plans that handle the animation of the interactions between characters (e.g. ask someone a favor, planned meetings, chats between friends...).

3.1 Social Ontology

The set of possible social relations among the agents within an artificial society can be ontologically represented in the form of interrelations between classes of agents. Figure 2 shows the extensions made to the object ontology previously presented in [13] in order to hold agent relations. We distinguish two basic levels of social relations: the level of individuals (i.e. *agentSocialRelations)* and the institutional level (i.e. *groupSocialRelations)*. When one agent is related with another single agent, an *agentSocialRelation* will link them. Different application domains can need specific relations; thus, Domain Specific Ontologies are used to inherit particular relations from the core ontology. For instance, the property *workFriend* is used by the waiters in the virtual bar presented in section 4 to model the characteristic of being a friend of a workmate. Other examples of individual relations are familiy relations such as to be parent of or to be married with another agent. In this case, there is not only semantic but also structural difference, since *parent* is a unidirectional relation whereas *marriedWith* is bidirectional.

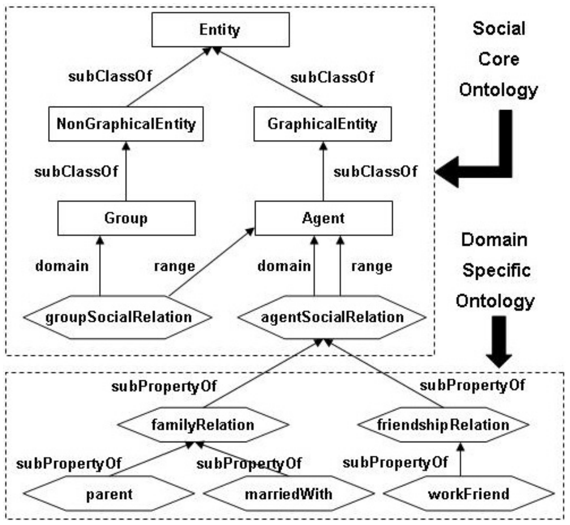


Fig. 2. Social ontology.

On the other hand, *groupSocialRelations* can be used to represent an agent belonging to a group. The social network created by this type of relation can be explored to get the rest of the agents of the same group, thus modeling a one-to-many relation. The *Group* class is an abstraction of any kind of aggregation. Therefore, we can model from physical groups such as the players of a football team to more sophisticated mental aggregations such as individuals of a certain social class or people of the same religious ideology. Although not considered in this paper, many-to-many relations between groups could also be created using this ontological approach. The dynamics of how these relations are created, modified and terminated falls out of the scope of this paper. Thus, at the moment relations are set off-line and do not change during the simulation.

3.2 Social library

The simulation of worlds inhabited by interactive virtual actors normally involves facing a set of problems related to the use of shared limited resources and the need to animate pure social behaviors. Both types of problems are managed by the *Social library* by using a Multi-agent Resource Allocation approach [15]. This library allows the auctioning of tasks by any agent in order to reallocate them so that the global social welfare can be increased. Tasks are exchanged between agents using a first-price sealed-bid (FPSB) auction model where the agents express their preferences using *performance* and *social utility functions*.

The performance utility function $U^{i}_{perf}(<i \leftarrow t>)$ of a bidder agent i reflects the efficiency achieved when he performs the task t. There can be many reasons for an agent to be more efficient: he might perform the task faster than others because of his know-how or it might be using a resource that allows several tasks to be performed simultaneously (e.g. a coffee machine in a virtual bar can be used by a waiter to make more than one coffee at the same time). The utility function has to favor the performance of the agents, but high performances can also be unrealistic for the animation of artificial human societies. For example, if all agents work as much as they can, they will display unethical or robotic behaviors. Furthermore, agents should also show pure social behaviors to animate the normal relations between the members of a society.

Whereas the performance utility function modeled the interest of an agent to exchange a task from an efficiency point of view, we introduce two additional social utilities to represent the social interest in exchanging a task. The aim of social utilities is to promote task allocations that lead the agents to perform social interactions with other agents (e.g. planned meetings with their friends). Therefore, these functions take into account the social relations established between the agents and defined in the ontology to compute a value that expresses their social preferences. Negotiation of long sequences of actions is not very interesting for interactive characters, as plans will probably be broken due to the dynamism of the environment and to other unpredictable events. Thus, we define the following social utility functions:

- Internal social utility ($U^{i}_{int}(<i \leftarrow t, j \leftarrow t_{next}>)$): is the utility that a bidder agent i assigns to a situation where i commits to do the auctioned task t so that the auctioneer agent j can execute his next task t_{next}.
- External social utility ($U^{i}_{ext}(<j \leftarrow t>)$): is the utility that a bidder agent i assigns to a situation where the auctioneer agent j executes the auctioned task t while i continues his current action.

The winner determination problem has two possible candidates coming from performance and sociability. In equation 1 the welfare of a society is related to performance, hence, the winner of an auction will be the agent that bid the maximum performance utility. On the other hand, equation 2 defines the social winner based on the maximum social utility received to pass the task to a bidder (see $U^{*}_{int}(t)$ in equation 3) and the maximum social utility given by all bidders to the situation where the task is not exchanged but performed by the auctioneer *j* (see $U^{*}_{ext}(t)$ in equation 4). To balance task exchange, social utilities are weighted with a reciprocity matrix. Equation 5 defines the reciprocity factor w_{ij} for two agents *i* and *j*, as the ratio between the number of favors (i.e. tasks) that *j* has made to *i*.

$$winner_{perf}(t) = \left\{ k \in Agents \mid U^{k}_{perf}(t) = \max_{i \in Agents} \left\{ U^{i}_{perf}(< i \leftarrow t >) \right\} \right. \quad \textbf{(1)}$$

$$winner_{soc}(t) = \begin{cases} j & U^{*}_{ext}(t) >= U^{*}_{\text{int}}(t) \\ i & U^{*}_{ext}(t) < U^{*}_{\text{int}}(t) \text{ and } U^{i}_{\text{int}}(t) = U^{*}_{\text{int}}(t) \end{cases} \quad \textbf{(2)}$$

$$U^{*}_{\text{int}}(t) = \max_{i \in Agents} \left\{ U^{i}_{\text{int}}(< i \leftarrow t, j \leftarrow t_{next} >) * w_{ji} \right\} \quad \textbf{(3)}$$

$$U^{*}_{ext}(t) = \max_{i \in Agents} \left\{ U^{i}_{ext}(< j \leftarrow t >) * w_{ij} \right\} \quad \textbf{(4)}$$

$$w_{ij} = Favors_{ji} \big/ Favors_{ij} \quad \textbf{(5)}$$

At this point, agents can decide whether to adopt this kind of social allocations or to be only rational as explained previously. They choose between them in accordance with their *Sociability* factor, which is the probability to select the social winner instead of the rational winner. *Sociability* can be adjusted in the range [0,1] to model intermediate behaviors between efficiency and total reciprocity. This can provide great flexibility when animating characters, since *Sociability* can be dynamically changed thus producing different behaviors depending on the world state.

4 Application example

In order to test the presented social multi-agent framework, we have created a virtual university bar where waiters take orders placed by customers (see figure 3a). The typical locations in a bar (e.g. a juice machine) behave like resources that have an associated time of use to supply their products (e.g. 2 minutes to make an orange juice) and they can only be occupied by one agent at a time. Agents can be socially linked using the concepts defined in the *Social Ontology*. According to them, all waiters are related through a *groupSocialRelation* to *Waiters*, a group representing their role (see figure 3b). Moreover, they can be individually related with other waiters through *workFriend*. This relation semantically means that the agents are friends at work and, in this application, it has been modeled as bidirectional but not transitive. For example, in figure 3b, Albert is friend of Dough and John but these later ones are not friends of each other. Moreover, we have also specified three possible groups of customers: teachers, undergraduates and graduates. The social network specified by them is used to promote social meetings among customers in the university bar.

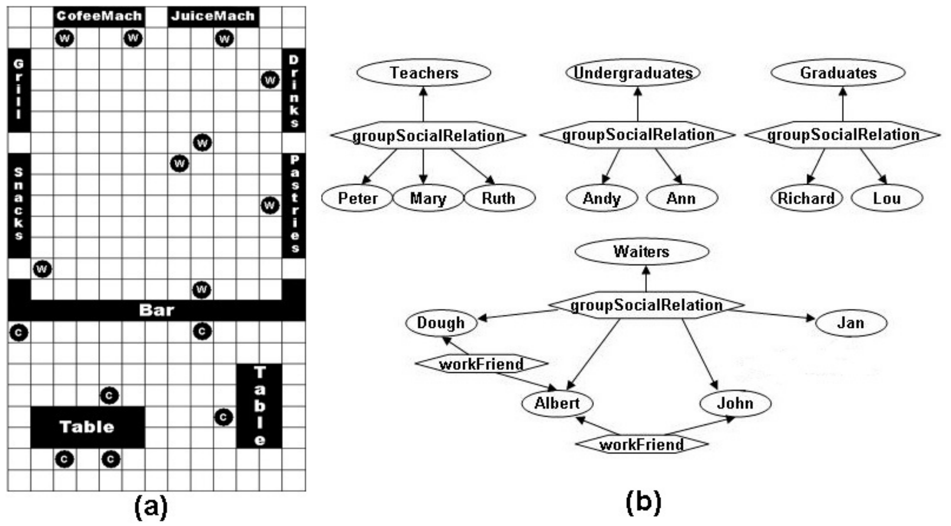


Fig. 3. (a) Virtual university bar environment (b) Social relations between agents.

The waiters are governed by the finite state machine[2] shown in figure 4a, where orders are served basically in two steps: first, using the corresponding resource (e.g. the grill to produce a sandwich) and second, giving the product to the customer. Tasks are always auctioned before their execution in order to find good social allocations. Equations 6 and 7 define the utility values returned by the performance utility function for these tasks. This function aims at maximizing the number of parallel tasks being performed and represents the waiters' willingness to serve orders as fast as possible. Social behaviors defined for a waiter are oriented to animate chats between his friends at work. Therefore, waiters implement the internal and external social utility functions detailed in equations 8 and 9, where *Near* computes the distance between the agents while they are executing a pair of tasks. These functions evaluate social interest as the chance to meet a *workFriend* in the near future (i.e. a planned meeting).

$$U^{i}_{perf}(i \leftarrow 'Use') = \begin{cases} 1 & \begin{array}{l}\text{if } [(\text{i} = \text{Auctioneer}) \text{ and } (\text{IsFree(Resource)}] \text{ or} \\ \quad [\text{IsUsing(i, Resource) and not(IsComplete(Resource)}]\end{array} \\ 0 & \qquad\qquad \text{Otherwise} \end{cases} \cdot \quad \textbf{(6)}$$

$$U^{i}_{perf}(i \leftarrow 'Give') = \begin{cases} 1 & \begin{array}{l}\text{if } [(\text{i} = \text{Auctioneer}) \text{ and } (\text{nextAction} = \text{NULL})] \text{ or} \\ \quad [\text{currentTask} = '\text{Give}' \text{ and not(handsBusy} < 2)]\end{array} \\ 0 & \qquad\qquad \text{Otherwise} \end{cases} \cdot \quad \textbf{(7)}$$

[2] Specified by means of plans in Jason's extended version of AgentSpeak(L)

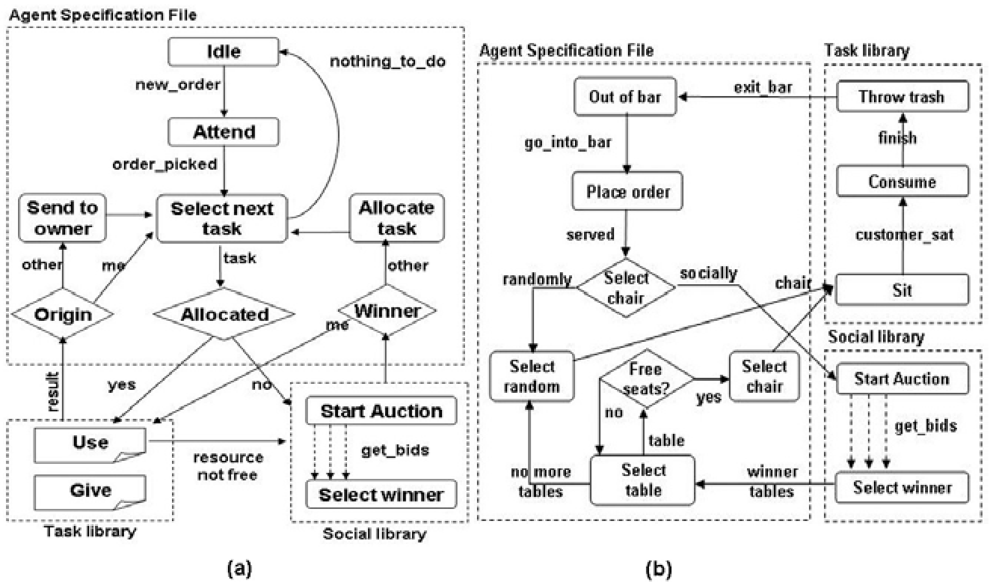


Fig. 4. (a) Waiter specification, (b) Customer specification.

$$U^i_{\text{int}}(<i \leftarrow t, j \leftarrow t_{next}>) = \begin{cases} 1 & \text{if IsWorkFriend(i, j) and Near(t, t}_{\text{next}}\text{) and} \\ & \text{ExecTime(t}_{\text{next}}\text{) > RemainTime(currenTask)} \\ 0 & \text{Otherwise} \end{cases} \cdot \quad \textbf{(8)}$$

$$U^i_{ext}(j \leftarrow t) = \begin{cases} 1 & \text{if IsWorkFriend(i, j) and Near(currentTask, t)} \\ 0 & \text{Otherwise} \end{cases} . \quad \textbf{(9)}$$

On the other hand, customers place orders and consume them when served. Now, we are not interested in improving customer performance but in animating interactions between the members of a social group (i.e. teachers, undergraduates and graduates). The finite state machine in figure 4b governs the actuation of customers that use auctions to solve the problem of *where to sit*. Depending on his or her sociability factor, a customer can randomly choose a chair or start an auction to decide where to sit and consume. This auction is received by all customers in the bar, which use the external social utility function defined in equation 10 to promote social meetings. This function uses the *groupSocialRelations* to determine if two individuals belong to the same group. We define the performance and the internal social utility functions as 0 since task passing is not possible in this case (i.e. no-one can sit instead of another customer). Finally, when a social meeting emerges, both waiters and customers use the plans in the *Conversational Library* to sequence the speech-acts needed to animate commitments, greetings or simple conversations.

$$U^i_{ext}(j \leftarrow \text{'}Sit\text{'}) = \begin{cases} 1 & \text{if IsSameGroup(i, j) and IsConsuming(i, auctionedTable)} \\ 0 & \text{Otherwise} \end{cases} . \quad \textbf{(10)}$$

5 Results

To illustrate the effects of the social techniques previously defined we have animated the virtual bar example with up to 10 waiters serving 100 customers, both with different sociability configurations. We estimate the social welfare of our society using two metrics explained along this section: *Throughput* and *Animation*. *Throughput* is an indicator in the range *[0,1]* that estimates how close a simulation is to the ideal situation in which the workload can be distributed among the agents and no collisions arise. Thus, equation 11 defines *Throughput* as the ratio between this ideal simulation time (T^{*}_{sim}) and the real simulation time (T_{sim}), where N_{tasks} and N_{agents} are the number of tasks and agents respectively and $\overline{T_{task}}$ is the mean time to execute a task.

$$Throughput = \frac{T^{*}_{sim}}{T_{sim}} = \frac{N_{tasks} * \overline{T_{task}} / N_{agents}}{T_{sim}} . \qquad \textbf{(11)}$$

Figure 5a shows the *Throughput* obtained by different types of waiters versus self-interested agents (i.e. agents with no social mechanisms included). In this first social configuration, all waiters are friends and customers are automatically assigned a group (teacher, undergraduate or graduate) when they come into the scenario. Self-interested agents collide as they compete for the use of the shared resources and these collisions produce high waiting times as the number of agents grows. We can enhance this low performance with elitist agents (*Sociability* = 0) which coordinately exchange tasks with others that can carry them out in parallel thus reducing the waiting times for resources. Nevertheless, they produce unrealistic outcomes since they are continuously working if they have the chance, leaving aside their social relationships (e.g. chats between friends). The *Sociability* factor can be used to balance rationality and sociability. Therefore, the *Throughput* for the sort of animations we are pursuing should be placed somewhere in between elitist and fully reciprocal social agents (*Sociability*=1). On the other hand, figure 5b demonstrates that the higher the *Sociability* factor is, the larger the number of social meetings that will be performed by the customers when they sit at a table.

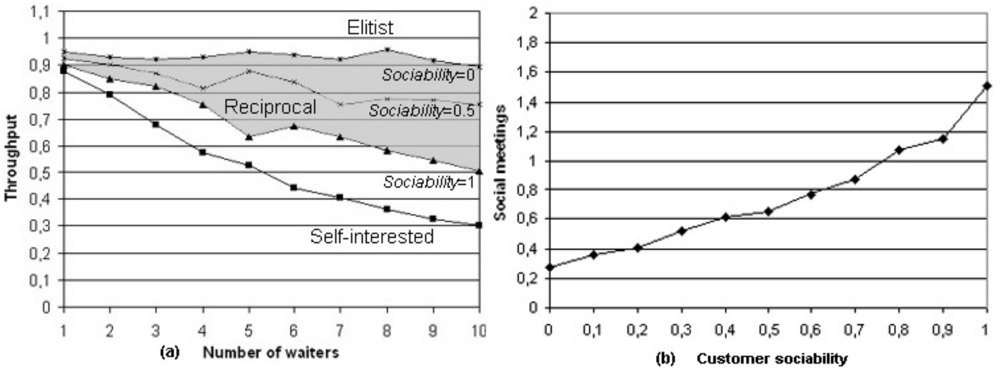


Fig. 5. (a) Waiter *Throughput*, (b) Customer social meetings.

Throughput is an estimator for the behavioral performance but, despite being a basic requirement when simulating groups of virtual characters, it is not the only criterion to evaluate when we try to create high quality simulations. Therefore, we have defined another estimator that takes into account the amount of time that the designer of the simulation wants to be spent in social interactions. According to this, we define the following simulation estimator:

$$Animation = \frac{T^{*}_{sim} + T_{social}}{T_{sim}} . \qquad \textbf{(12)}$$

, where T_{social} represents the time devoted to chat and to animate social agreements between friends. In our virtual bar we have chosen T_{social} as the 35% of T^{*}_{sim}. Figure 6 shows the animation values for 10 reciprocal social waiters with 4 degrees of friendship: all friends, 75% of the agents are friends, half of the agents are friends and only 25% of the agents are friends. As we have already mentioned, low values of *Sociability* produce low quality simulations since the values obtained for the animation function are greater than the reference value (*Animation=1*). On the other hand, high values of *Sociability* also lead to low quality simulations, especially when the degree of friendship is high. In these cases, the number of social conversations being animated is too high to be realistic and animation is far from the reference value. The animation function can be used to extract the adequate range of values for the *Sociability* factor, depending on the situation being simulated. For example, in our virtual bar we consider as good quality simulations those which fall inside ±10% of the reference value (see shared zone in figure 6). Hence, when all the waiters are friends, good simulations emerge when *Sociability* $\in$ [0.1,0.3].

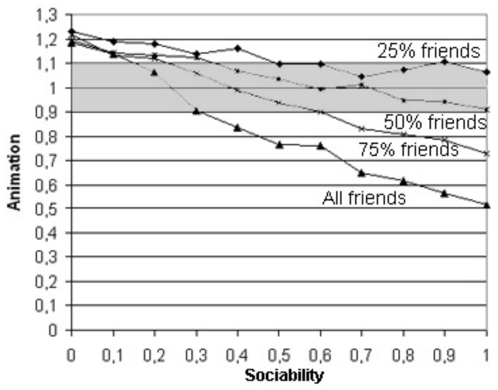


Fig. 6. Animation results obtained for waiters.

Finally, table 1 compares the amount of time devoted to execute each type of task in executions with 10 elitist waiters (*Sociability*=0) and 10 fully reciprocal social waiters (*Sociability*=1). The irregular values in the columns T_{use} and T_{give} on the left side of the table demonstrate how some agents have specialized in certain tasks. For instance, agents 2, 5, 9 and 10 spend most of their time giving products to the customers while agents 3 and 7 are mainly devoted to using the resources of the bar (e.g. coffee machine, etc). Although specialization is a desirable outcome in many multi-agent systems, egalitarian human societies need also to balance the workload assigned to each agent. On the right side of the table, fully reciprocal social waiters achieve equilibrium between the time they are giving products and the time they are using the resources of the environment (see columns T_{use} and T_{give}). Furthermore, the reciprocity factor balances the number of favors exchanged among the agents (compare *Balance* columns). A collateral effect of this equilibrium is the increase in the waiting times, since social agents will sometimes prefer to meet his friends in a resource than to reallocate the task (compare columns T_{wait}).

Table 1. Time distribution for 10 waiters in the bar (time values are in seconds).

	Sociability = 0				*Sociability* = 1			
Agent	T_{wait}	T_{use}	T_{give}	*Balance*	T_{wait}	T_{use}	T_{give}	*Balance*
1	0	32	19	-6	16	69	34	-2
2	3	4	26	-3	18	58	24	-2
3	14	52	1	28	41	45	16	0
4	3	16	28	-3	48	60	27	3
5	0	7	30	-16	34	58	12	-1
6	3	37	17	-1	48	64	14	-2
7	0	67	4	21	18	48	24	1
8	0	45	17	1	33	45	24	4
9	7	5	23	-11	46	36	21	0
10	1	6	41	-10	27	56	20	-1

6. Conclusions and Future Work

The animation of groups of intelligent characters is a current research topic with a great number of behavioral problems to be tackled. We aim at incorporating human style social reasoning in character animation. Therefore, this paper presents a technique to properly balance social with task-oriented plans in order to produce realistic social animations. We propose the use of ontologies to define the social relations within an artificial society and the use of a market based mechanism to reach sociability by means of task exchanges. The multi-agent animation framework presented allows for the definition of different types of social agents: from elitist agents (that only use their interactions to increase the global performance of the group) to fully reciprocal agents. These latter agents extend the theory of social welfare with a reciprocity model that allows the agents to control the emergence of social interactions among the members of a society. Work is ongoing to provide the agents with mechanisms to self-regulate their *Sociability* factor depending on their social relations and on their previous intervention. Thus, agents will be able to dynamically adjust to the situation in order to stay within the boundaries of good quality animations at all times.

Acknowledgments. This work has been jointly supported by the Spanish MEC and European Commission FEDER funds under grants Consolider Ingenio-2010 CSD2006-00046 and TIN2006-15516-C04-04.

References

1. T. Bickmore and J. Cassell. Relational agents: A model and implementation of building user trust. In *Proceedings of the Conference on Human Factors in Computing Systems - CHI'2001*, Seattle, USA, 2001. ACM Press.
2. R.H. Bordini, A.C. da Rocha, J.F. Hübner, A.F. Moreira, F.Y. Okuyama and R. Vieira. A Social Simulation Platform Based on Agent-Oriented Programming. *JASSS,* vol.8, 2005.
3. R.H. Bordini and J.F. Hübner. Jason. *Available at http://jason.sourceforge.net/* March 2007
4. J. Ciger. Collaboration with agents in VR environments. *PhD Thesis* 2005.
5. S. Casare and J. Sichman. Towards a Functional Ontology of Reputation. In In *AAMAS'05: Autonomous Agents and Multi-agent Systems*. ACM, 2005.
6. R. Conte and C. Castelfranchi. Cognitive and Social Action. *UCL Press*, London, 1995.
7. K.S. Decker. Environment Centered Analysis And Design of Coordination Mechanisms. *PhD thesis*. University of Massachusetts Amherst, May 1995.
8. K.S. Decker and V.R. Lesser. Designing a family of coordination algorithms. *Readings in Agents*. Huhns and Singh editors, 1997.
9. R. Falcone, G. Pezzulo, C. Castelfranchi and G. Calvi. Why a cognitive trustier performs better: Simulating trust-based Contract Nets. In *AAMAS'04: Autonomous Agents and Multi-agent Systems*. ACM, 1392-1393, 2004.
10. J. A. Giampapa and K. Sycara. Team-Oriented Agent Coordination in the RETSINA Multi-Agent System. On *Tech. Report CMU-RI-TR-02-34*, Robotics Institute-Carnegie Mellon University, 2002.
11. G.A. Giménez-Lugo, J.S. Sichman and J.F. Hübner. *"Addressing the social components of knowledge to foster communitary exchanges"*. In *International Journal on Web Based Communities*, 1(2), pages 176-194, 2005.
12. F.Grimaldo, M.Lozano and F.Barber. Integrating social skills in task-oriented 3D IVA. In *IVA'05: International Conference on Intelligent Virtual Agents*. Springer, 2005.
13. F.Grimaldo, F.Barber and M. Lozano. An ontology-based approach for IVE+VA. In *IVEVA International Conference*. 2006.
14. H. Hexmoor. From Inter-Agents to Groups. In *ISAI'01: International Symposium on Artificial Intelligence*. 2001
15. L.M. Hogg and N.Jennings. Socially intelligent reasoning for autonomous agents. *IEEE Transactions on System Man and Cybernetics*, 31(5), 2001.
16. E. C-C. Kao, P. H-M. Chang, Y-H. Chien and V-W. Soo. Using Ontology to Establish Social Context and Support Social Reasoning. In *IVA'05: International Conference on Intelligent Virtual Agents*. Springer, 2005.
17. J. Piaget. Sociological Studies. Routlege, London, 1995.
18. R. Prada, and A. Paiva. Believable groups of Synthetic Characters. In *AAMAS'05: Autonomous Agents and Multi-agent Systems*. ACM, 2005.
19. M.Ribeiro, A.C. da Rocha and R.H. Bordini. A System of Exchange Values to Support Social Interactions in Artificial Societies. In *AAMAS'03: Autonomous Agents and Multi-agent Systems*. ACM, 2003.
20. J.S. Sichman and Y. Demazeau. On Social Reasoning in Multi-Agent Systems. *Revista Ibero-Americana de Inteligencia Artificial*, 13, 68-84. AEPIA, 2001.
21. B. Tomlinson and B. Blumberg. Social synthetic characters. *Computer Graphics*, 26(2), May 2002.
22. M. Viroli, A. Ricci and A. Omicini. Operating instructions for intelligent agent coordination. *The Knowledge Engineering Review*. Vol. 21:1, 49-69. 2006.

The examination of an information-based approach to trust

Maaike Harbers[1], Rineke Verbrugge[2], Carles Sierra[3], and John Debenham[4]

[1] Institute of Information and Computing Sciences, Utrecht University, P.O.Box 80.089, 3508 TB Utrecht, The Netherlands
maaike@cs.uu.nl

[2] Institute of Artificial Intelligence, University of Groningen, Grote Kruisstraat 2/1, 9712 TS Groningen, The Netherlands
rineke@ai.rug.nl

[3] IIIA-CSIC, Campus UAB, 08193 Cerdanyola, Catalonia, Spain
sierra@iiia.csic.es

[4] Faculty of Information Technology, University of Technology, Sydney, PO Box 123, Broadway, NSW 2007, Australia
debenham@it.uts.edu.au

Abstract. This article presents the results of experiments performed with agents based on an operalization of an information-theoretic model for trust. Experiments have been performed with the ART test-bed, a test domain for trust and reputation aiming to provide transparent and recognizable standards. An agent architecture based on information theory is described in the paper. According to a set of experimental results, information theory is shown to be appropriate for the modelling of trust in multi-agent systems.

1 Introduction

In negotiation, one tries to obtain a profitable outcome. But what is a profitable outcome: to pay little money for many goods of high quality? Although this seems to be a good deal, it might not always provide the most profitable outcome in the long run. If negotiation partners meet again in the future, it could be more rational to focus on the relationship with the other agents, to make them trust you and to build up a good reputation.

In computer science and especially in distributed artificial intelligence, many models of trust and reputation have been developed over the last years. This relatively young field of research is still rapidly growing and gaining popularity. The aim of trust and reputation models in multi-agent systems is to support decision making in uncertain situations. A computational model derives trust or reputation values from the agent's past interactions with its environment and possible extra information. These values influence the agent's decision-making process, in order to facilitate dealing with uncertain information.

Big differences can be found among current models of trust and reputation, which indicates the broadness of the research area. Several articles providing an overview of the field conclude that the research activity is not very coherent and needs to be more unified [1–4]. In order to achieve that, test-beds and frameworks to evaluate and compare the models are needed.

Most present models of trust and reputation make use of game-theoretical concepts [1, 5]. The trust and reputation values in these models are the result of utility functions and numerical aggregation of past interactions. Some other approaches use a cognitive model of reference, in which trust and reputation are made up of underlying beliefs. Castelfranchi and Falcone [6] developed such a cognitive model of trust, based on beliefs about competence, dependence, disposition, willingness and persistence of others. Most existing models of trust and reputation do not differentiate between trust and reputation, and if they do, the relation between trust and reputation is often not explicit [1, 3]. The ReGreT system [7] is one of the few models of trust and reputation that does combine the two concepts. Applications of computational trust and reputation systems are mainly found in electronic markets. Several research reports have found that seller reputation has significant influences on on-line auction prices, especially for high-valued items [3]. An example is eBay, an online market place with a community of over 50 million registered users [2].

Sierra and Debenham [8] introduced an approach using information theory for the modeling of trust, which has been further developed in [9], [10]. The present article presents an examination of Sierra and Debenham's information-based approach to trust. Experiments have been performed with the ART test-bed [4], a test domain for trust and reputation. Section 2 introduces the trust model, section 3 describes the ART test-bed, and section 4 describes how the model has been translated into an agent able to participate in the ART test-bed. The remainder of the article gives an overview of the experiments (section 5) and the results (section 6), followed by a discussion (section 7). The article ends with conclusions and recommendations for further research (section 8).

2 The information-based model of trust

In Sierra and Debenham's information-based model, trust is defined as the measure of how uncertain the outcome of a contract is [8]. All possible outcomes are modelled and a probability is ascribed to each of them. More formally, agent α can negotiate with agent β and together they aim to strike a deal δ. In the expression $\delta = (a, b)$, a represents agent α's commitments and b represents β's commitments in deal δ. All agents have two languages, language C for communication and language L for internal representation. The language for communication consists of five illocutionary acts (Offer, Accept, Reject, Withdraw, Inform), which are actions that can succeed or fail. With an agent's internal language L, many different worlds can be constructed. A possible world represents, for example, a specific deal for a specific price with a specific agent.

To be able to make grounded decisions in a negotiation under conditions of uncertainty, the information-theoretic method denotes a probability distribution over all possible worlds. If an agent would not have any beliefs or knowledge, it would ascribe to all worlds the same probability to be the actual world. Often however, agents do have knowledge and beliefs which put constraints on the probability distribution. The agent's knowledge set K restricts *all worlds* to all *possible worlds*: that is, worlds that are consistent with its knowledge. Formally, a world v corresponds to a valuation function on the positive ground literals in the language, and is an element of the set of all possible worlds V. Worlds inconsistent with the agent's knowledge are not considered.

An agent's set of beliefs B determines its opinion on the probability of possible worlds: according to its beliefs some worlds are more probable to be the actual world than others. In a probability distribution over all possible worlds, W, a probability p_i expresses the degree of belief an agent attaches to a world v_i to be the actual world. From a probability distribution over all possible worlds, the probability of a certain sentence or expression in language L can be derived. For example the probability $P(executed \mid accepted)$ of whether a deal, once accepted, is going to be executed can be calculated. This derived sentence probability is considered with respect to a particular probability distribution over all possible worlds. The probability of a sentence σ is calculated by taking the sum of the probabilities of the possible worlds in which the sentence is true. For every possible sentence σ that can be constructed in language L the following holds: $P_{\{W|K\}}(\sigma) \equiv \Sigma_n \{p_n : \sigma \text{ is true in } v_n\}$. An agent has attached given *sentence probabilities* to every possible statement φ in its set of beliefs B.

A probability distribution over all possible worlds is consistent with the agent's beliefs if for all statements in the set of beliefs, the probabilities attached to the sentences are the same as the derived sentence probability. Expressed in a formula, for all beliefs φ in B the following holds: $B(\varphi) = P_{\{W|K\}}(\varphi)$. Thus, the agent's beliefs impose linear constraints on the probability distribution. To find the best probability distribution consistent with the knowledge and beliefs of the agent, *maximum entropy inference* (see [11]) uses the probability distribution that is maximally non-committal with respect to missing information. This distribution has maximum entropy and is consistent with the knowledge and beliefs. It is used for further processing when a decision has to be made.

When the agent obtains new beliefs, the probability distribution has to be updated. This happens according to the principle of *minimum relative entropy*. Given a prior probability distribution $\underline{q} = (q_i)_{i=1}^n$ and a set of constraints, the *principle of minimum relative entropy* chooses the posterior probability distribution $\underline{p} = (p_i)_{i=1}^n$ that has the least relative entropy with respect to $\underline{q}$, and that satisfies the constraints. In general, the relative entropy between probability distribution p and q is calculated as follows: $D_{RL}(p \parallel q) = \Sigma_{i=1}^n p_i \log_2 \frac{p_i}{q_i}$. The principle of minimum relative entropy is a generalization of the principle of maximum entropy. If the prior distribution $\underline{q}$ is uniform, the relative entropy of $\underline{p}$ with respect to $\underline{q}$ differs from the maximum entropy $H(\underline{p})$ only by a constant. So the principle of maximum entropy is equivalent to the principle of minimum relative entropy with a uniform prior distribution (see also [8]).

While an agent is interacting with other agents, it obtains new information. Sierra and Debenham [8] mention the following types of information from which the probability distribution can be updated:

- *Updating from decay and experience*. This type of updating takes place when the agent derives information from its direct experiences with other agents. It is taken into account that negotiating people or agents may forget about the behavior of a past negotiation partner.
- *Updating from preferences*. This updating is based on past utterances of a partner. If agent α prefers a deal with property Q_1 to a deal with Q_2, he will be more likely to accept deals with property Q_1 than with Q_2.

– *Updating from social information*. Social relationships, social roles and positions held by agents influence the probability of accepting a deal.

Once the probability distribution is constructed and up to date, it can be used to derive trust values. From an actual probability distribution, the trust of agent α in agent β at the current time, with respect to deal δ or in general, can be calculated. The trust calculation of α in β is based on the idea that the more the actual executions of a contract go in the direction of the agent α's preferences, the higher its level of trust. The relative entropy between the probability distribution of acceptance and the distribution of the observation of actual contract execution models this idea. For $T(\alpha,\beta,b)$, the trust of agent α in agent β with respect to the fulfillment of contract (a,b), the following holds:

$$T(\alpha,\beta,b) = 1 - \sum_{b' \in B(b)^+} P^t(b') \log \frac{P^t(b')}{P^t(b'|b)}$$

Here, $B(b)^+$ is the set of contract executions that agent α prefers to b. $T(\alpha,\beta)$, the trust of α in β in general, is the average over all possible situations. After making observations, updating the probability distribution and calculating the trust, the probability of the actual outcomes for a specific contract can be derived from the trust value and an agent can decide about the acceptance of a deal.

3 The ART Test-bed

Participants in the ART test-bed [4] act as appraisers who can be hired by clients to deliver appraisals about paintings, each for a fixed client fee. Initially, a fixed number of clients is evenly distributed among appraisers. When a session proceeds, appraisers whose final appraisals were most accurate are rewarded with a larger share of the client base. Each painting in the test-bed has a fixed value, unknown to the participating agents. All agents have varying levels of expertise in different artistic eras (e.g. classical, impressionist, post-modern), which are only known to the agents themselves and which will not change during a game. To produce more accurate appraisals, appraisers may sell and buy opinions from each other. If an appraiser accepts an opinion request, it has to decide about how much time it wants to invest in creating an opinion. The more time (thus money) it spends in studying a painting, the more accurate the opinion.

However, agents might (on purpose) provide bad opinions or not provide promised opinions at all. Then without spending time on creating an opinion, the seller receives payment. So to prevent paying money for a useless opinion, the test-bed agents have to learn which agents to trust. To facilitate this process, agents can buy information about other agents' reputations from each other. Here again agents do not always tell the truth or provide valuable information.

Appraisers produce final appraisals by using their own opinion and the opinions received from other appraisers. An agent's final appraisal is calculated by the simulation, to ensure that appraisers do not strategize for selecting opinions after receiving all purchased opinions. The final appraisal $p*$ is calculated as a weighted average of received opinions: $p* = \frac{\sum_i (w_i \cdot p_i)}{\sum_i w_i}$. In the formula, p_i is the opinion p received from provider i and w_i is the appraiser's weight for provider i: the better α trusts an agent i, the higher the weight w_i attached to that agent and the more importance will be given to its opinion.

Agent α determines its final appraisal by using all the opinions it received plus its own opinion. The true painting value t and the calculated final appraisal $p*$ are revealed by the simulation to the agent. The agent can use this information to revise its trust models of other participants.

4 An information-based test-bed agent

The implemented test-bed agent ascribes probabilities to the accuracy of the opinions other agents provide. The agent maintains a probability distribution for each era of expertise with respect to each agent. The different possible worlds in a probability distribution represent the possible grades of the opinions an agent might provide in a specific era. An opinion of high grade means that the appraised value of a painting is close to the real value of the painting. A low grade means that the agent provides very bad opinions in the corresponding era or that the agent does not provide opinions at all. The quality of an opinion actually is a continuous variable, but to fit the model all possible opinions are grouped into ten levels of quality. The act of promising but not sending an opinion is classified in the lowest quality level.

The probability distributions are updated during the course of a session each time the agent receives new information, which can be of three types:

- Updating from direct experiences;
- Updating from reputation information;
- Updating from the evaporation of beliefs (forgetting).

Updating from reputation information corresponds to *Updating from social information* in Sierra and Debenham's model [8]. The other two types of updating are derived from *Updating from decay and experience* in the model.

Updating from direct experiences takes place when the agent receives the true values of paintings. The value of a constraint is obtained by taking the relative error of an opinion: the real value of a painting and an agent's estimated value of a painting are compared to each other. *Updating from reputation information* takes place when the agent receives witness information. The value of a constraint is derived by taking the average of the reputation values in all messages received at a specific time from trusted agents about a specific agent and era. *Updating from forgetting* is performed each time when a probability distribution is updated either from direct experiences or from reputation information.

Direct experiences and reputation information are translated into the same type of constraints. Such a constraint is for example: agent α will provide opinions with a quality of at least 7 in era e with a certainty of 0.6. This constraint is put to the probability distribution of agent α and era e. After updating from this constraint, the probabilities of the worlds 7, 8, 9 and 10 should together be 0.6. Constraints are always of the type opinions of *at least* quality x.

The value of a constraint (the quality grade) derived from a direct experience is obtained by comparing the real value of a painting to an agent's estimated value according to the equation: $constraintValue = 10 \cdot (1 - \frac{|appraisedValue - trueValue|}{trueValue})$. The outcome represents the quality of the opinion and a new constraint can be added to the set of beliefs. If a value lower than one is found, a constraint with the value of one is added to the set of beliefs. Reputation information is translated into a constraint by taking the average of the reputation values in all messages received at a specific time from trusted agents about a specific agent and era multiplied by ten: $constraintValue = 10 \cdot \Sigma_{r \in reps} \frac{r}{n_1}$, where r is a reputation value, $reps$ is the set of useful reputation values and n_1 is the size of $reps$.

With a set of constraints and the principle of maximum entropy, an actual probability distribution can be calculated. Therefore one general constraint is derived from all the stored constraints for calculating the probability distribution. The general constraint is a weighted average of all the constraints stored so far, calculated according to the following equation: $generalconstraintValue =$
$\frac{1}{n_2} \cdot \Sigma_{c \in C} \frac{1}{(c(t_{obtained}) - t_{current}) + 1} \cdot c(value)$, where constraint c is an element of the set C of stored constraints and n_2 the total amount of constraints. Each constraint c consists of the time it was obtained $c(t_{obtained})$ and a quality grade $c(value)$, calculated with one of the formulas $constraintValue$ above. The outcome is rounded to get an integer value.

The constraints are weighted with a factor of one divided by their age plus one (to avoid fractions with a zero in the denominator). Forgetting is modelled by giving younger constraints more influence on the probability distribution than older constraints. In this calculation, constraints obtained from reputation information are weighted with a factor which determines their importance in relation to constraints obtained from direct information. A ratio of 0.3:1, respectively, was taken because reputation info is assumed to have less influence than info from direct experiences. With the principle of maximum entropy, a new and updated probability distribution can be found.

Finally, when all information available has been processed and the probability distributions are up to date, trust values can be derived from the probability distributions. There are two types of trust, the trust of a particular agent in a specific era and the trust of a particular agent in general. The trust value of an agent in a specific era is calculated from the probability distribution of the corresponding agent and era. In an *ideal probability distribution*, the probability of getting opinions of the highest quality is very high and the probability of getting opinions with qualities lower than that is very low. Now trust can be calculated by taking one minus the relative entropy between the ideal and the actual probability distribution, as follows: $trust(agent, era) = 1 - \Sigma_{i=1}^{n_3} (P_{actual}(i) \cdot \log \frac{P_{actual}(i)}{P_{ideal}(i)})$, where n_3 is the number of probabilities. The trust of an agent in general is calculated by taking the average of the trust values of that agent in all the eras. At each moment of the game, the agent can consult its model to determine the trust value of an agent in general or the trust value of an agent with respect to a specific era. These trust values guide the behavior of the agent.

At the beginning of a new session the agent trusts all agents, so the probability distributions are initialized with all derived trust values (for each agent in each era) at 1.0. During the game the model is updated with new constraints and trust values change. The general behavior of the information-based agent is honest and cooperative towards the agents it trusts. The agent buys relevant opinions and reputation messages from all agents it trusts (with trust value 0.5 or higher). The agent only accepts and invests in requests from trusted agents, and if the agent accepts a request it provides the best possible requested information. If the agent does not trust a requesting agent, it informs the other agent by sending a decline message. If a trusted agent requests for reputation information, the agent provides the trust value its model attaches to the subject agent.

If the agent trusts an agent requesting for opinions, it always highly invests in ordering opinions from the simulator for that agent. Finally, the agent uses the model for generating weights for calculating the final opinions. It weights each agent (including itself) according to the trust in that agent in that era.

5 Set-up of the experiments

To test the influences of the use of different types of information, four variations of an information-based agent have been made. The suffixes in the names of the agents indicate the information types they use for updating: *de* corresponds to direct experiences, *rep* to reputation information and *time* to forgetting.

- Agent *Info-de* only updates from direct experiences;
- Agent *Info-de-time* updates from direct experiences and from forgetting;
- Agent *Info-rep-time* updates from reputation information and forgetting;
- Agent *Info-de-rep-time* updates from all three types of information.

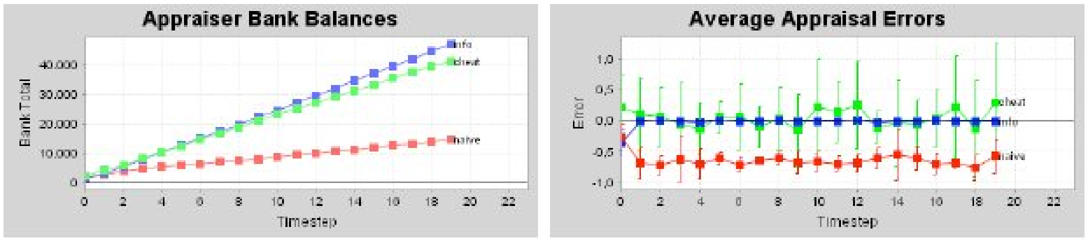
The performances of these agents in the ART test-bed are in the first place measured by their ability to make accurate appraisals, which is indicated by their client shares after the last game round. Besides, information about the agents' bank account balances will be presented. The use of each of the information types is expected to increase the average appraisal accuracy of an information-based test-bed agent. Moreover, the use of the combination of all three information types is expected to deliver the best results. In order to verify the correctness of these expectations, three test conditions have been designed and four extra agents have been implemented.

The **first condition** tests an agent's ability to distinguish between a cooperating and a non-cooperating agent. In this first part of the experiment, the agents *Info-de*, *Info-de-time* and *Info-de-rep-time* each participated in a game together with the test-agents *Cheat* and *Naive*. The test-agent *Cheat* never makes reputation or opinion requests itself, but when it receives requests it always promises to provide the requested reputation information or opinions. As its name suggests, the agent cheats on the other agents and it never sends any promised information. Its final appraisals are just based on its own expertise. The agent *Naive* bases its behavior on the idea that all agents it encounters are trustworthy and *Naive* keeps on trusting others during the whole course of a game. This agent always requests every other agent for reputation information and opinions, it accepts all requests from other agents and it highly invests in creating the requested opinions. Its final appraisals are based on its own expertise and on the (promised but sometimes not received) opinions of all other agents.

For the **second condition**, a third test-agent was developed to investigate other agents' ability to adapt to new situations. This agent *Changing* shows the same behavior as *Naive* during the first ten rounds of a game. Then it suddenly changes its strategy and from the eleventh game round till the end of the game it behaves exactly the same as the agent *Cheat*. The performances of the agents *Info-de* and *Info-de-time* in reaction to *Changing* have been examined.

The **third condition** was designed to examine the updating from reputation information. This type of updating is only of use if there are agents in the game that provide reputation information, so a reputation information providing agent *Providing* has been

Fig. 1. Bank account balances and average appraisal errors of agents *Info-de-time* (black), *Cheat* (light grey) and *Naive* (dark grey) in the first test conditions.



	Cheat		**Naive**		**Agent**	
	Bank	*Client*	*Bank*	*Client*	*Bank*	*Client*
info-de	45957	24.5	14361	8.8	40700	26.4
info-de-time	47975	25.9	13552	8.8	40262	25.0
info-de-rep-time	46097	24.7	14073	8.2	41461	26.7

Table 1. Averages for three information-based agents in conditions of type one.

implemented. The only difference with *Info-de-time* is that the *Providing* agent always accepts reputation requests and provides the wished reputation information, whereas the agent *Info-de-time* only provides reputation to agents it trusts. The agents *Info-de-time*, *Info-rep-time* and *Info-de-rep-time* each participated in a game with *Providing*, *Cheat* and *Naive*.

6 Results

In the first experiment, each of the agents *Info-de*, *Info-de-time* and *Info-de-rep-time* participated in a test-bed game together with the agents *Cheat* and *Naive*. The graphics in Figure 1 show an example of a session with the agents *Info-de-time*, *Cheat* and *Naive*. Left the development of the agents' bank account balance during the whole game is shown. All agents have increasing balances, but *Info-de-time* ends the game with the most and *Naive* with the least money. The right part of the figure shows the average appraisal errors of the agents in each round. The appraisals of *Naive* are obviously less accurate than those of the other two agents. This can be explained by *Naive*'s behavior to keep on trusting the cheating agent during the whole game. *Info-de-time* provides its least accurate appraisals the first game round; there it still has to learn that it cannot trust the agent *Cheat*. After that, its appraisals are the most accurate: the errors are close to the zero line and show the least deviation. This can be explained by *Info-de-time* using the expertise of two agents (itself and *Naive*), whereas *Cheat* only uses its own expertise.

Table 1 shows the averages of 30 sessions for the three information-based agents in condition one. In the tables, Client refers to the final number of clients of an agent at the end of a session and Bank means its final bank account balance. The first row shows the average final bank account balance and average final number of clients of respectively, *Cheat*, *Naive* and *Info-de*, for the sessions in which the three of them participated together in the game. The second row displays the results of the sessions with *Cheat*, *Naive* and *Info-de-time*. Applying Student T-test (two-tailed, homoscedastic distribution) showed that with a significance level of 5% one can only conclude that *Info-de-rep-time* gathers a significantly bigger client share than *Info-de-time*. The differences in bank account balances between the different agents are not significant.

In the second condition *Info-de* and *Info-de-time* participate in a game with the agent *Changing*, which starts to cheat from the tenth round of the game. In contrast to *Info-de*, the agent *Info-de-time* does take forgetting into account. As time goes by, information gathered in the past becomes less and less important. The difference is clear: after a first big decrease in appraisal accuracy when the agent *Changing* starts cheating, *Info-de-time* learns from *Changing*'s new behavior and adjusts its trust values. Its past beliefs about a seemingly trustworthy agent *Changing* do not overrule the new information it gathers and it ends with higher scores. The averages of all the sessions with the agent *Changing* are presented in Table 2. Both client share and bank account balance of the two information-based agents are significantly different on a 5% level of significance according to the Student T-test. The results of the third condition, testing the update from reputation information, are shown in Table 3. A Student T-test demonstrates that all differences in client shares between the three tested agents are significant.

	Changing		**Agent**	
	Bank	*Client*	*Bank*	*Client*
info-de	44189	33.4	25817	6.6
info-de-time	36211	21.2	33864	18.8

Table 2. Averages for the agent *Changing*.

7 Discussion

It was expected that the experiments would show that each of the three types of updating would contribute to appraisal accuracy. Condition one shows that, except for *Info-de-time*, all agents updating from direct experiences provide more accurate appraisals than *Cheat* and *Naive*, which do not update from past experiences. The third condition of the experiment is even more convincing regarding the usefulness of information from experiences. Two information-based agents, one with and one without updating from direct experiences, were tested in the same condition. The agent that updated from direct experiences had a significantly larger final client share and therefore must have produced more accurate appraisals. Thus, the expectation that updating from direct experiences improves the appraisal accuracy is supported by the experimental results.

For evaluating updating from forgetting, the first two test conditions can be examined. Here two information-based agents updating from direct experiences, one of them also updating from forgetting, were tested in the same condition. In the condition with the agents *Cheat* and *Naive*, the agent *Info-de* scored better than *Info-de-time*, but the difference is not significant. In the condition with the agent *Changing*, the agent *Info-de-time* updating from forgetting, has a significant larger client share than *Info-de*. This supports the expectation that updating from forgetting would contribute to more accurate appraisals.

The last type of information, updating from reputation information, has been examined in the third condition. The participating agents are the information-based agent to be evaluated, combined with the three test-agents *Cheat*, *Naive*, and *Providing* which

	Cheat		**Naive**		**Providing**		**Agent**	
	Bank	*Client*	*Bank*	*Client*	*Bank*	*Client*	*Bank*	*Client*
info-de-time	43252	23.1	12986	10.6	34889	23.3	34245	22.7
info-rep-time	45337	22.3	15363	12.7	35337	23.5	28713	21.1
info-de-rep-time	41076	21.3	14089	10.8	34988	23.4	35099	24.5

Table 3. Averages for three information-based agents in the third set of conditions.

provides reputation information. The agent *Providing* performs very well, so the reputation information it provides is supposed to be useful. Agent *Info-rep-time* does not update from any of its own experiences, so its performance only depends on updating from reputation information. *Info-rep-time* ended with much larger client shares than *Naive*, so it seems to use *Providing*'s reputation information profitably. This observation supports the expectation that the use of reputation information would increase the average appraisal accuracy of an information-based test-bed agent. Of course this conclusion only holds when there is at least one agent in the game that is able and willing to provide useful reputation information.

The results show that all three types of updating contribute to appraisal accuracy, but do they also work well in combination? Updating from forgetting can be used in combination with the other two types of updating without hindering them. However, updating from information from direct experiences and from reputation information cannot be added to each other. When more reputation information is used, less information from direct experiences can be used and vice versa. The results show that in both condition one and three, the use of all available types of information yields the most accurate appraisals.

However, in the first condition *Naive* is the only agent providing reputation information and it assumes that each agent is trustworthy, so it always provides reputations with the value 1. So the good performance of the agent using reputation information in this condition cannot be due to its updating from reputation information. In the third condition however, useful reputation information is provided and the agent *Info-de-rep-time* seems to make good use of it. So the results support the expectation that all three types of updating contribute to providing more accurate appraisals, and the information-based agent using all three types of updating provides the most accurate appraisals.

The experiments performed are not exhaustive and when interpreting the results, some remarks should be kept in mind. First, an agent's performance depends a lot on the other participants in a test-bed game. For example, an agent with a very sophisticated model for dealing with reputation information only profits when other agents are prepared to provide reputation information. A cooperative agent functions very well with other cooperative participants, but it might perform very badly with non-cooperative participants. In the experiments, four test-agents were used, *Naive*, *Cheat*, *Changing* and *Providing*, which show quite simple and obvious behavior. The use of more complex test-agents would provide more information. Moreover, conditions with larger numbers of participants would create new situations and might yield extra information.

A second consideration is the choice of the ART test-bed. A general problem of all test-beds is *validity*: does the system test what it is supposed to test? Especially when complicated concepts are involved, it is difficult to prove that a test-bed just examines the performance of a model on that particular concept. The aim of the ART test-bed is to compare and evaluate trust- and reputation-modeling algorithms [4]. But what do the developers exactly understand by trust and reputation? The ART test-bed is quite complicated and allows so many variables that it is sometimes difficult to explain why something happened.

A final remark about the experiments is that in the translation of the trust model to a test-bed agent some adjustments and adaptations had to be made. Not every part of the model can be used in the ART test-bed. Sierra and Debenham's model [8] allows updating from preferences and different power relations between agents; these facets cannot be tested by the ART test-bed. On the other hand, the trust model lacks theory for some topics needed in the ART test-bed. The updating from reputation was not very elaborated in the model [8] and had to be extended. Besides, the information-based trust model does not provide a negotiation strategy: it is a system to maintain values of trust. The strategy used might have influenced the test results.

8 Conclusion and further research

The goal of this article is to examine Sierra and Debenham's information-based model for trust [8]. Therefore, an agent based on the model has been implemented and several experiments in the ART test-bed have been performed. The experiments showed that the information-based agent learned about its opponents during a game session and could distinguish between cooperating and non-cooperating agents. They also demonstrated that the three examined types of updating (from direct experiences, from reputation information and from the evaporation of beliefs as time goes by), all improved the agent. So in general expectations have been met: the results are promising and the information-based approach seems to be appropriate for the modeling of trust.

The diversity and the amount of the experiments could be extended. The information-based agent could be tested in more conditions with different test agents and with larger amounts of participating agents. It would also be interesting to pay more attention to the agent's strategy. Besides, the implementation of the agent could be improved. Some aspects of the trust model could be translated more literally to the implementation of the information-based agent. Even another test-bed could be used, as the ART test-bed is not able to evaluate all aspects of the theory. All these suggestions would deliver new information about the model and would justify making stronger statements about it.

As to Sierra and Debenham's trust model itself [8, 9], its core seems to be robust and clear: they use a clear definition of trust and probability distributions are updated from a set of beliefs with the principle of minimum relative entropy. The experiments support the model. To further improve it, more work could be done on other concepts related to trust. For example, now it provides some initial ideas about how to deal with reputation and other types of social information. But social aspects are becoming more and more central in the field of multi-agent systems lately, so a contemporary model of trust should give a complete account of it. So, it can be said conclusively that the core of the model seems to be a good approach, but for a fully developed approach to trust and reputation more work should be done. This should not be a problem, because the model is flexible and provides ample space for extensions.

Acknowledgements. Carles Sierra's research is partially supported by the OpenKnowledge STREP project, sponsored by the European Commission under contract number

FP6-027253, and partially by the Spanish project “Agreement Technologies” (CONSOLIDER CSD2007-0022, INGENIO 2010).

References

1. Sabater, J., Sierra, C.: Review on computational trust and reputation models. Artificial Intelligence Review **24** (2005) 33–60
2. Jøsang, A., Ismail, R., Boyd, C.: A survey of trust and reputation systems for online service provision. Decision Support Systems **43** (2007) 618–644
3. Mui, L., Mohtashemi, M., Halberstadt, A.: Notions of reputation in multi-agents systems: a review. In: AAMAS ’02: Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems, New York, NY, USA, ACM Press (2002) 280–287
4. Fullam, K., Klos, T., Muller, G., Sabater, J., Topol, Z., Barber, K.S., Rosenschein, J.: A specification of the agent reputation and trust (ART) testbed: experimentation and competition for trust in agent societies. In et al., F.D., ed.: Fifth International Conference on Autonomous Agents and Multiagent systems (AAMAS-05), Utrecht, The Netherlands (2005) 512–518
5. Ramchurn, S.D., Huynh, D., Jennings, N.R.: Trust in multiagent systems. Knowledge Engineering Review **19** (2004) 1–25
6. Castelfranchi, C., Falcone, R.: Principles of trust for MAS: Cognitive anatomy, social importance, and quantification. In Demazeau, Y., ed.: Proceedings of the Third International Conference of Multi-agent Systems (ICMAS98). (1998) 72–79
7. Sabater, J., Sierra, C.: REGRET: reputation in gregarious societies. In: AGENTS’01: Proceedings of the Fifth International Conference on Autonomous Agents, New York, NY, USA, ACM Press (2001) 194–195
8. Sierra, C., Debenham, J.: An information-based model for trust. In et al., F.D., ed.: Fifth International Conference on Autonomous Agents and Multiagent systems (AAMAS-05), Utrecht, The Netherlands (2005) 497–504
9. Sierra, C., Debenham, J.: Trust and honour in information-based agency. In Stone, P., Weiss, G., eds.: Proceedings Fifth International Conference on Autonomous Agents and Multi Agent Systems AAMAS-2006, Hakodate, Japan, ACM Press, New York (2006) 1225 – 1232
10. Sierra, C., Debenham, J.: Information-based agency. In: Proceedings of Twentieth International Joint Conference on Artificial Intelligence IJCAI-07, Hyderabad, India (2007)
11. MacKay, D.: Information Theory, Inference and Learning Algorithms. Cambridge University Press (2003)

A Coherence Based Framework for Institutional Agents

Sindhu Joseph, Carles Sierra, and Marco Schorlemmer

Artificial Intelligence Research Institute, IIIA Spanish National Research Council, CSIC
Bellaterra (Barcelona), Catalonia, Spain
{joseph,sierra,marco}@iiia.csic.es

Abstract. We introduce in this paper an agent model based on coherence theory. We give a formalization of Thagard's theory on coherence and use it to explain the reasoning process of an intentional agent that permits the agent to drop beliefs or to violate norms in order to keep a maximal state of coherence. The architecture is illustrated in the paper and a discussion on the possible use of this approach in the design of institutional agents is presented.

1 Introduction

Electronic institutions are multiagent system models inspired by human institutions [10] and used to create technological extensions of human societies [12]. These devices are designed to help agents cope with the uncertainty on the environment and in some cases to increase their individual utility. They are important due to the bounded nature of human and software rationality (global maximization of individual utility cannot be guaranteed in a complex society). If two or more persons exchange goods with one another, then the result for each one will depend in general not merely upon his own actions but on those of the others as well [8]. Therefore, to make these exchanges possible, behavioral rules that govern the way in which individuals can cooperate and compete are required [7]. Behavioral rules translate the social objectives into executable permissions, prohibitions, and obligations. These modalities are collectively called *norms*. Thus, institutions are role based normative systems representing a collective intention[1]. This is the case in general, but we do acknowledge the fact that institutions need not always represent a collective intention. But such institutions almost always undergo periodic revolutions as an attempt to reinforce collective intention.

Human institutions tend to adapt when the group conscience shifts or is in conflict with the current institutional definition. It is thus important to know and be able to verify at any point in time, that the institutional definition do not have inconsistencies between its norms and the social objectives, among its norms, and that the definition is in agreement with the current values and objectives of the individuals in the group. Thus an institution to be sustainable almost always needs to continuously strive to achieve this consistent state, here we call it *equilibrium*. We say an institution is in a state of equilibrium when it has no incentive to change the institutional definition. When an inconstistency or a deviation from equilibrium is detected, it is also important to identify

[1] Collective intention here refers to the explicit expression of the intention and do not refer to the mental state.

the candidates that cause or are minimally consistent with the rest, to be able to bring the institution back into equilibrium with the minimum possible change.

An autonomous agent is motivated to join an institution when it believes that the individual goals of the agent can be satisfied within the institution. And that happens in our opinion when the beliefs or goals of the agent are *coherent* with the institutional objectives. For simplicity, here we assume that all institutional objectives are realized through norms. Thus being incoherent with a norm is equivalent to being incoherent with a corresponding institutional objective. An agent will hence need to continuously re-evaluate the alignment of its beliefs and goals with that of the norms of the institution. (The same applies to a group.) Thus, it is important for an agent to know whether there is an incoherence among the beliefs and the norms, and how the decision is made on what needs to be changed to bring the coherence back. This incoherence among other things drives the agent to violate a norm, revise a belief or both. The individual state of equilibrium is achieved when the coherence between individual beliefs and goals, those of the group and those of the institution is maximized.

We use the theory of coherence and the theory of cognitive dissonance to ground our framework. The *theory of coherence* [11] has been well studied in the field of cognitive science and as a general theory to describe the world. Coherence theory is about how different pieces fit together to make a whole. The acceptance of a new piece is based on its coherence with the rest available. That does not mean that there is a primary set that is given as accepted, but each time a system enters a new state, the pieces that contribute to coherence maximization in the new state are accepted.

The *theory of dissonance* [5] in social psychology is closely related to the theory of coherence. Leon Festinger calls dissonance as the distressing mental state in which people feel they “find themselves doing things that don’t fit with what they know, or having opinions that do not fit with other opinions they hold.” The tension of dissonance motivates us to change either our behavior or our belief in an effort to avoid a distressing feeling. The more important the issue and the greater the discrepancy between behavior and belief, the higher the magnitude of dissonance that we will feel.

In this paper we propose an institutional agent architecture based on the theory of coherence. This architecture permits us to talk about the coherence of the individual beliefs, desires and intentions[2], coherence among these cognitions, and the coherence among the cognitions and institutional norms or social commitments. In particular when there is an incoherence between any of these elements, the agent often needs to choose between a norm violation or a belief revision to maximize its internal coherence. That is, the theory of incoherence helps us to model autonomous agents who can reason about obeying or violating institutional norms. From the institutional point of view, the same tools can be used to reason about an institution, coherence of an institution with respect to the conscience of the group and how to evolve norms to stay in alignment with the objectives. While coherence theory helps to find the maximally coherent state, dissonance theory helps to decide how much of incoherence an agent or an institution can tolerate and which of the actions to chose from to reduce incoherence.

In Sections 2 and 3 we introduce our coherence-based framework and the reasoning of a coherence-maximizing agent. In Section 4 we illustrate with the help of an example,

[2] In the paper we discuss beliefs, the extension to desires and intentions is straight-forward.

how this framework can be used to reason about norm violations. We conclude with related work in Section 5 and discussion future work in Section 6. We use the example of a car agent in a traffic control institution. Here we give the intuitive summary of the example, for the reader to follow the coherence framework introduced in Section 2. In Section 4, we detail the example further. The car agent in our example has personal beliefs and intentions. Where-as the traffic control institution has a set of objectives which it implements through a number of norms. The car agent initially starts with the belief that the traffic control is efficient, and has a maximally coherent graph with his beliefs, intentions and institutional norms in it. But when the car agent reaches a situation where, he is made to stop at a traffic signal, where as the other lane has no cars waiting to go, he builds up a certain incoherence. This then leads to a norm violation as the agent encounters a high incoherence to maintain the intention to obey the traffic norms to restore maximum coherence.

2 Coherence framework

In this section we introduce a number of definitions to build the coherence framework. Our primary interest is to put the theory in relation to an institutional agent context and to provide a formal representation and some computing tools. We do this for the belief cognition of an agent and for the norms of an institution.

2.1 Coherence Graph

To determine the coherence of a set of elements, we need to explore their associations. We shall use a graph to model these associations in order to compute coherence of various partitions of a given set of elements, and to determine its maximally coherent partition as well as study other related aspects of coherency.

We shall define a coherence graph over an underlying logic. Given a set of propositional formulae PL, a *logic* over PL is a tuple $\mathcal{K} = \langle \mathcal{L}, A, \vdash \rangle$, with language $\mathcal{L} \subseteq PL \times [0,1]$, i.e., a set of pairs formed by a proposition and a confidence value between 0 and 1, a set of *axioms* $A \subseteq \mathcal{L}$, and a *consequence relation* $\vdash \subseteq 2^{\mathcal{L}} \times \mathcal{L}$.

The nodes of a coherence graph are always elements of $\mathcal{L}$. The consequence relation $\vdash$ determines the relationship between these elements, and thus puts contraints on the edges that are allowed in a coherence graph. Furthermore, propositions that are assumed to be true belong to the axioms A of the logic.

A coherence graph is therefore a set $(\in V)$ of nodes taken from $\mathcal{L}$ and a set E of edges connecting them. The edges are associated with a number called the *strength of the connection* which gives an estimate of how coherent the two elements are[3]. The strength value of an edge (φ, γ), noted $\sigma(\varphi, \gamma)$, respects the strength values that it has with other connected edges. It is important to note that a coherence graph is a *fully connected graph* with a restriction that for every node φ[4] $\in \mathcal{L}$, $\sigma(\varphi, \varphi) = 1$ and if there

[3] This value is fuzzy and is determined by the type of relation between the edges. For an *incoherence* relation, tends toward -1, for *coherence* a positive value tending toward 1.

[4] This should be understood as $\langle \varphi, d \rangle$, whenever it is understood from the context, we omit the d part of the element for better readability.

are two nodes φ and, ψ that are not related, then $\sigma(\varphi, \psi) = 0$. Further α is a projection function defined from the set V to $[0, 1]$ which projects the confidence degrees associated with elements of $\mathcal{L}$. The role of this function is to make the confidence degrees explicit in the graph for ease of explanation.

Definition 1. *Given a logic $\mathcal{K} = \langle \mathcal{L}, A, \vdash \rangle$ over a propositional language PL, a* coherence graph $\langle V, E, \sigma, \alpha \rangle$ *over $\mathcal{K}$ is a graph for which*

– $V \subseteq \mathcal{L}$
– $E = V \times V$
– $\sigma : E \rightarrow [-1, 1]$
– $\alpha : V \rightarrow [0, 1]$

and which satisfies the following constraints:

– $A \subseteq V$
– $\forall v \in V, \sigma(v, v) = 1$
– $\sigma(v, w) = \sigma(w, v)$

We write $\mathcal{G}(\mathcal{K})$ for the set of all coherence graphs over $\mathcal{K}$.

Given this general definition of a *coherence graph*, we can instantiate two specific families of coherence graphs namely the *belief coherence graphs $\mathcal{BG}$* and the *norm coherence graphs $\mathcal{NG}$*, which are of interest to us. $\mathcal{BG}$ represents graphs where the nodes are beliefs of an agent and the edges are association between beliefs. And $\mathcal{NG}$ represents nodes which are the possible norms defined in an institution. In this paper, we do not discuss the desire and the intention cognitions, but these can be defined similarly. And when defining the norm logic, we only talk about permissions and obligations, whereas norms may include prohibitions, too. Also for clarity we have kept the structure of the norms simple, but we intend to include objectives and values associated with a norm. The work by Atkinson and Bench-Capon [1] is indicative. We now define the belief and the norm logic to express the nodes of these graphs and their interconnections.

In our representation, beliefs are propositional formulas φ which are closed under negation and union with an associated confidence degree d. We may borrow the axioms and the consequence relation $\vdash$ from an appropriate belief logic. Then for example we have the following definition for the belief logic.

Definition 2. *Given the propositional language PL, we define the* belief logic $\mathcal{K}_B = \langle \mathcal{L}_B, A_B, \vdash_B \rangle$ *where*

– *the belief language $\mathcal{L}_B$ is defined as follows:*
  • *Given $\varphi \in PL$* and $d \in [0, 1]$, $\langle B\varphi, d \rangle \in \mathcal{L}_B$
  • *Given $\langle \theta, d \rangle, \langle \psi, e \rangle \in \mathcal{L}_B$, $\langle \neg\theta, f(d) \rangle \in \mathcal{L}_B$* and $\langle \theta \wedge \psi, g(d, e) \rangle \in \mathcal{L}_B$ where f and g are functions for example as in [3]
– *A_B as axioms of an appropriate belief logic.*
– *$\vdash_B$ is a consequence relation of an appropriate belief logic.*

We need a number of additional constraints that we want the Belief coherence graphs to satisfy. They are constraints on how the strength values have to be assigned. A constraint that we impose on this number is that if two elements are related by a $\vdash$, then the value should be positive and if two elements contradicts then then there is a negative strength[5]. And here we define α more concretely as the projection function over the belief degree. Then we have

Given the belief logic $\mathcal{K}_B$, the *set of all belief coherence Graphs is* $\mathcal{G}(\mathcal{K}_\mathcal{B})$ satisfying the additional constraints:

– Given $\varphi, \psi \in$ V *and* $\Gamma \subseteq$ V *and* $\Gamma \vdash \varphi$
  – $\forall \gamma \in \Gamma, \sigma(\varphi, \gamma) > 0$
  – $\forall \gamma \in \Gamma$ *and* $\psi = \neg\varphi, \sigma(\psi, \gamma) < 0$
– $\forall \langle B\varphi, d\rangle \in V, \alpha(\langle B\varphi, d\rangle) = d$

We can similarly derive the set of all norm coherence graphs $\mathcal{G}(\mathcal{K}_N)$ corresponding to norms. In our definition, norms define obligations and permissions associated with a role. We use *deontic logic* to represent the norms, with the difference that we use modalities subscripted with roles. Thus O_r and P_r represent deontic obligations and deontic permissions associated with a role $r \in R$, the set of all roles. In this paper we assume the confidence degrees associated with norms to be 1. Thus we have the following definition for a norm logic $\mathcal{K}_N$.

Definition 3. *Given the propositional language PL and the set of roles$R$, we define the* Norm logic $\mathcal{K}_N = \langle \mathcal{L}_N, A_N, \vdash_N\rangle$ *where*

– $\mathcal{L}_N$ *is defined as:*
  • Given $\varphi \in PL$ *and* $r \in R$*, then* $\langle O_r\varphi, 1\rangle, \langle P_r\varphi, 1\rangle \in \mathcal{L}_N$
  • Given $\langle\varphi, d\rangle$ and $\langle\psi, e\rangle \in \mathcal{L}_N$ then $\langle\neg\varphi, f_1(d)\rangle$ and $\langle\varphi \wedge \psi, g_1(d, e)\rangle \in \mathcal{L}_N$
– A_N *following the standard axioms of deontic logic.*
– $\vdash_N$ *using the standard deduction of deontic logic*[6]

Given the norm logic $\mathcal{K}_N$ *the set of all norm coherence graphs* is $\mathcal{G}(\mathcal{K}_N)$ satisfying the additional constraints:

– Given $\varphi, \psi \in \mathcal{L}$ *and* $\Gamma \subseteq \mathcal{L}$ *and* $\Gamma \vdash \varphi$
  – $\forall \gamma \in \Gamma, \sigma(\varphi, \gamma) > 0$
  – $\forall \gamma \in \Gamma$ *and* $\psi = \neg\varphi, \sigma(\psi, \gamma) < 0$
– $\forall \langle\varphi, d\rangle \in V, \alpha(\langle\varphi, d\rangle) = 1$

2.2 Calculating Coherence

We can now define the coherence value of a graph, the partition that maximizes coherence and the coherence of an element with respect to the graph. These values will help an agent to determine whether to keep a belief or drop it, whether to obey a norm or

[5] This relates to Thagard's *deductive coherence*, though in this paper, we limit our discussion to the general coherence relation.
[6] For an introduction to deontic logic, see [13] and in the context of institutions see [6]

violate it to increase coherence and which of the beliefs or norms need to be dropped to maximize coherence. This will also help an institution decide whether to accept a proposed norm change and to determine the gain in coherence when accepting or rejecting a change.

We use the notion of coherence as maximizing constraint satisfaction as defined by Thagard [11]. The intuition behind this idea is that there are various degrees of coherence/incoherence relations between nodes of a coherence graph. And if there is a strong negative association between two nodes, then the graph will be more coherent if we decide to accept one of the nodes and reject the other. Similarly when there is a strong positive association, coherence will be increased when either both the nodes are accepted or both are rejected. Thus we can construct a partition of the set of nodes, with one set of nodes in the partition being accepted and the other rejected in such a way to maximize the coherence of the entire graph. Such accepted sets are denoted by $\mathcal{A}$ and the rejected sets by $\mathcal{R}$. The coherence value is calculated by considering positive associations within nodes of $\mathcal{A}$ and within nodes of $\mathcal{R}$ and negative associations between nodes of $\mathcal{A}$ and $\mathcal{R}$. This criteria is called *satisfaction of constraints*. More formally we have the following definition:

Definition 4. *Given a coherence graph $g \in \mathcal{G}(\mathcal{K})$ and a partition $(\mathcal{A}, \mathcal{R})$ of V, we define the* set of satisfied associations $C^+ \subseteq E$ *as*

$$C^+ = \left\{ \forall (v_i, v_j) \in E \;\middle|\; \begin{array}{l} v_j \in \mathcal{A} \leftrightarrow v_i \in \mathcal{A} (\text{or } v_j \in \mathcal{R} \leftrightarrow v_i \in \mathcal{R}) \text{ when } \sigma(v_i, v_j) \geq 0 \\ v_j \in \mathcal{A} \leftrightarrow v_i \in \mathcal{R} \text{ when } \sigma(v_i, v_j) < 0 \end{array} \right\}$$

In all other cases the association is said to be unsatisfied.

To define coherence, we first define the total strength of a partition. The total strength of a partition is the sum of the strengths of all the satisfied constraints multiplied by the degrees (the α values) of the nodes connected by the edge. Then the coherence of a graph is defined to be the maximum among the total strengths when calculated over all its partitions. We have the following definitions:

Definition 5. *Given a coherence graph $g \in \mathcal{G}(\mathcal{K})$, we define* the total strength of a partition $\{\mathcal{A}, \mathcal{R}\}$ *as*

$$S(g, \mathcal{A}, \mathcal{R}) = \sum_{(v_i, v_j) \in C^+} \mid \sigma(v_i, v_j) \mid \cdot \, \alpha(v_i) \cdot \alpha(v_j) \tag{1}$$

Definition 6. *Given a coherence graph $g = \langle V, E, \sigma, \alpha \rangle \in \mathcal{G}(\mathcal{K})$ and given the total strength $S(g', \mathcal{A}, \mathcal{R})$ for all partitions of V ($\mathcal{P}(V)$), we define* the coherence of g *as*

$$C(g) = max\{S(g', \mathcal{A}, \mathcal{R}) \mid \mathcal{A}, \mathcal{R} \in \mathcal{P}(V)\} \tag{2}$$

and we say that the partition with the maximal value divides the set of nodes into an accepted set $\mathcal{A}$ and a rejected set $\mathcal{R}$.

Given the coherence $C(g)$ of a graph, *the coherence of an element $C(\varphi)$* is the ratio of coherence when φ is in the accepted set with respect to φ not being in the accepted set. That is if the acceptance of the element improves the overall coherence of the set considered, than when it is rejected, then the element is said to be coherent with the set. Then we have the definition:

Definition 7. *Given a coherence graph $g \in \mathcal{G}(\mathcal{K})$, we define the* coherence of an element $\varphi \in V$ *as*

$$C(\varphi) = \frac{\max\{S(g, \mathcal{A}, \mathcal{R}) \mid (\mathcal{A}, \mathcal{R}) \in \mathcal{P}(V) \wedge \varphi \in \mathcal{A}\}}{\max\{S(g, \mathcal{A}, \mathcal{R}) \mid (\mathcal{A}, \mathcal{R}) \in \mathcal{P}(V) \wedge \varphi \in \mathcal{R}\}} \tag{3}$$

Similar to the coherence definitions of a graph, we now define the dissonance of a graph. We define dissonance as the measure of incoherence that exists in the graph. Deducing from the theory of dissonance [5] an increase in dissonance increases in an agent the need to take a coherence maximizing action. We use the dissonance as a criteria to chose among the number of alternative actions an agent can perform such as belief revision, norm violation or commitment modification for example. The dissonance of a graph is computed as the difference between the total strength of the graph and the coherence of the graph. Thus we have the following definition:

Definition 8. *Given a coherence graph $g \in \mathcal{G}(\mathcal{K})$, we define the dissonance of g as*

$$D(g)^7 = \begin{cases} 0 \text{ if } C(g) = 0 \\ \frac{C(g) - S(g)}{C(g)} \text{ otherwise} \end{cases} \tag{4}$$

2.3 Graph Composition

For an agent that is part of an institution and has social relations, it not only needs to maximize the internal coherence between its beliefs, but also needs to maximize the *social coherence* which is the coherence between the beliefs and the commitments made in the context of his social relations. Similarly, an agent who belongs to an institution, needs to maximize the *institutional role coherence*, that is the coherence between the projection of the norms onto the role he plays in the institution and his beliefs. This leads naturally the notion of graph composition, which will allow us to explore the coherence or incoherence that might exist between nodes of one graph and those of the other.

The nodes of a composite graph are always the disjoint union of the nodes of the individual graphs. The set of edges contains at least those edges that existed in the individual graphs. In addition a composite graph may have new edges between nodes of one graph to the nodes of the other graph.

Definition 9. *Let $\mathcal{K}_1 = \langle \mathcal{L}_1, A_1, \vdash_1 \rangle$ and $\mathcal{K}_2 = \langle \mathcal{L}_2, A_2, \vdash_2 \rangle$ be logics over propositional language PL_1 and PL_2. Let $g_1 = \langle V_1, E_1, \sigma_1, \alpha_1 \rangle \in \mathcal{G}(\mathcal{K}_1)$ and $g_2 = \langle V_2, E_2, \sigma_2, \alpha_2 \rangle \in \mathcal{G}(\mathcal{K}_2)$. The* set of composite graphs $g_1 \odot g_2 \subset \mathcal{G}(\mathcal{K})$ *is the set of those coherence graphs $\langle V, E, \sigma, \alpha \rangle \in \mathcal{G}(\mathcal{K})$ over logic $\mathcal{K} = \langle \mathcal{L}, A, \vdash \rangle$—where $\mathcal{L}$ is the disjoint union of $\mathcal{L}_1$ and $\mathcal{L}_2$, A is the disjoint union of A_1 and A_2, and $\vdash$ is the smallest consequence relation containing both $\vdash_1$ and $\vdash_2$[8]— such that*

[7] When $C(g) = 0$, $S(g) = 0$ and hence the dissonance $D(g) = 0$

[8] For the moment we assume that the properties that make $\vdash_1$ and $\vdash_2$ a consequence relation as the same.

- $\mathrm{V} = \{\mathcal{L}_1/\varphi \mid \varphi \in \mathrm{V}_1\} \cup \{\mathcal{L}_2/\varphi \mid \varphi \in \mathrm{V}_2\}$[9]
- $\mathrm{E} = \mathrm{V} \times \mathrm{V}$ such that
  - if$(\varphi, \psi) \in \mathrm{E}_1$ then $(\mathcal{L}_1/\varphi, \mathcal{L}_1/\psi) \in \mathrm{E}$
  - if$(\varphi, \psi) \in \mathrm{E}_2$ then $(\mathcal{L}_2/\varphi, \mathcal{L}_2/\psi) \in \mathrm{E}$
- $\sigma : \mathrm{E} \longrightarrow [-1, 1]$ such that
  - $\sigma(\mathcal{L}_1/\varphi, \mathcal{L}_1/\gamma) = \sigma_1(\varphi, \gamma)$
  - $\sigma(\mathcal{L}_2/\varphi, \mathcal{L}_2/\gamma) = \sigma_2(\varphi, \gamma)$

These properties state that the nodes of the composite graph are the disjoint union of the original graphs. When making the composition, the existing edges and strength values are preserved.

3 A coherence maximizing agent

In this section we describe some of the reasoning performed by a coherence maximizing agent. Consider an agent a having a belief coherence graph b, intention coherence graph i and role coherence graph n_r. At any moment in time the agent aims at coherence maximization. When the coherence cannot be further maximized, a does nothing, or has no incentive to act. For an agent who has no social commitments, nor is part of any institution, nor has any unfulfilled intentions, the accepted set $\mathcal{A}$ is the entire belief set, as he is not likely to have an incoherence.

We consider an agent that is part of an institution, has social commitments and is in the state of equilibrium. When a new belief is created (either communicated to the agent by others, by observation, or internally deduced), a executes the sequence in Figure 1.

```
1:  while $S(g, V, \emptyset) \equiv C(g) \& \ new\langle\varphi, d\rangle$ do
2:      $v_{new} \leftarrow \langle\varphi, d\rangle$
3:      $V \leftarrow V \cup \{v_{new}\}$
4:      for $v_i \in V$ do
5:          if $v_i \in \Gamma$ and $\Gamma \vdash v_{new}$ then
6:              $\sigma(v_i, v_{new}) = 1$
7:          end if
8:          if $v_{new} \in \Gamma$ and $\Gamma \vdash v_i$ then
9:              $\sigma(v_i, v_{new}) = 1$
10:         end if
11:         if $v_i, v_{new} \vdash$ then
12:             $\sigma(v_i, v_{new}) = -1$
13:         end if
14:     end for
15:     $S \leftarrow S(b \odot i \odot n_r)$ using eq(1)
16:     $C \leftarrow C(b \odot i \odot n_r)$ using eq(2)
17:     $D \leftarrow D(b \odot i \odot n_r)$ using eq(4)
18:     if $D \geq threshold$ then
19:         update $\mathcal{A}, \mathcal{R}$
20:     end if
21: end while
```

Fig. 1. Reasoning in a coherence maximizing agent

The lines from 1 to 14 put certain constraints on how new edges are created and how their strength values are determined. Here we assume that a human user will provide

[9] We write $\mathcal{L}_i/\varphi$ for those elements of $\mathcal{L}$ that come form $\mathcal{L}_i$ in the disjoint union, with $i = 1, 2$.

them while respecting the constraints though we envision many semi automatic methods worth exploring (see section 6). The lines from 15 to 17 recalculate the strength, coherence and dissonance values of the new graph. Lines 18 and 19 check whether the dissonance value exceeds the threshold and if it does, the agent acts by removing the nodes causing the incoherence from the accepted set. To keep the discussion simple in this algorithm, we have simply removed the nodes. But in reality, the reaction to an incoherence can vary greatly. For instance a mildly distressed agent may choose to ignore the incoherence, may be satisfied with lowering the degree associated with a particular belief, may still choose to follow a norm. Where as a heavily distressed agent may not only chose to violate a norm, but initiate a dialogue to campaign for a norm change.

4 An Example

The main entities in our example are a car agent a having the role c in a traffic control institution and the institution itself T. We take a very simplified version of the objectives of T as

– minimizing the probability of collisions
– increasing the traffic handling capacity

To meet these objectives, the traffic control system has a signal at the crossing of the lanes along with specific norms of use. The norms of the traffic control system for the car agents belong to the set $\mathrm{N_c}$.

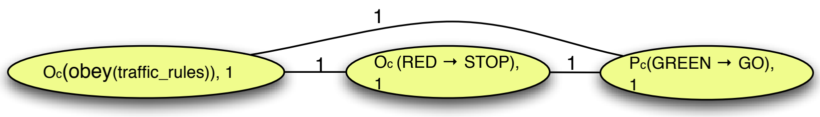


Fig. 2. Norm Coherence graph of the traffic control institution

The traffic is controlled using the norms given below and the corresponding norm coherence graph is shown in the Figure 2. Note that all the coherence graphs in this example have additional self loops which are not drawn for the sake of readability. But it is included in the coherence calculations.

- $\mathrm{O_c(RED \rightarrow STOP), 1} \rightarrow$ *It is obligatory to STOP, when the signal is RED*
- $\mathrm{P_c(GREEN \rightarrow GO), 1} \rightarrow$ *It is permitted to GO , when the signal is GREEN*

Here we illustrate the model with one of the most simple cases, namely the crossing between a major and a minor lane. The major lane has more traffic than minor lane. Due to the fixed time control, and due to ignoring to assign priority to the lanes, the signal durations are the same for both major and minor lanes. Thus there are situations when there are no cars waiting to cross at the minor lane and there is a “RED” light at the major lane. So the car agents at the major lane sometimes experience an incoherence when trying to follow the traffic norms. We now show the evolution of the coherence of an agent situated at the major lane with the help of the some figures.

A car agent a of role c at the major lane has the intention to reach destination X at time T. He holds a number of beliefs which support this intention. A few relevant beliefs of a for this intention are *can reach destination X in time t* and *traffic control is efficient* and a generic belief that *It is good to reduce pollution*. The composite graph $b \odot i$ is shown in Figure 3.

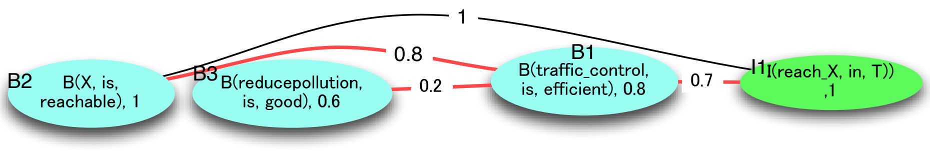

1
0.8
0.2
0.7
B2 B(X, is, reachable), 1
B3 B(reducepollution, is, good), 0.6
B1 B(traffic_control, is, efficient), 0.8
I1 I(reach_X, in, T)) ,1

Fig. 3. $b \odot i$ Coherence graph of the car agent

We use Equations 1, 2, 4 of Section 2 for calculating the various coherence values of all the graphs of the example[10].

The coherence of the graph is $C(b \odot i) = 5.296$ with $\mathcal{A} = \{B1, B2, B3, I1\}$ and $D(b \odot i) = 0$. As a is part of the traffic control system, having a role c, the projection of the norms n_c to the beliefs graph of a with an additional intention *to stop at RED signal* is as given in Figure 4. This additional intention is due to the fact that a intends to follow the norms of the institution. Now the coherence of the composite graph is $C(b \odot i \odot n_c) = 17.716$ with $\mathcal{A} = \{B1, B2, B3, I1, I2, N1, N2\}$ and dissonance $D(b \odot i \odot n_c) = 0$, still staying 0.

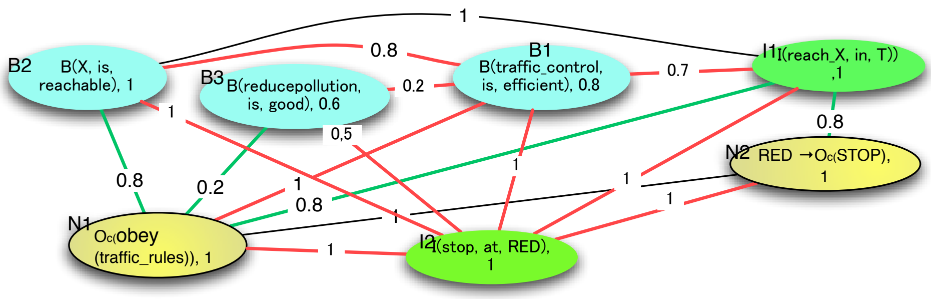
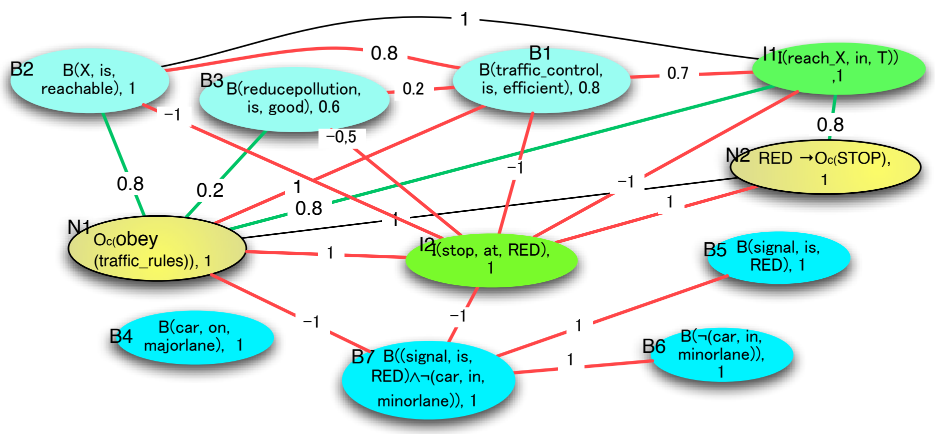

1
0.8
0.2
0.7
B2 B(X, is, reachable), 1
B3 B(reducepollution, is, good), 0.6
B1 B(traffic_control, is, efficient), 0.8
I1 I(reach_X, in, T)) ,1
0.8
N2 RED →Oc(STOP), 1
N1 Oc(obey (traffic_rules)), 1
I2 I(stop, at, RED), 1
0,5
1
0.8
0.2
1
0.8
1
1
1
1
1

Fig. 4. Belief Coherence graph of the car agent with projected norms

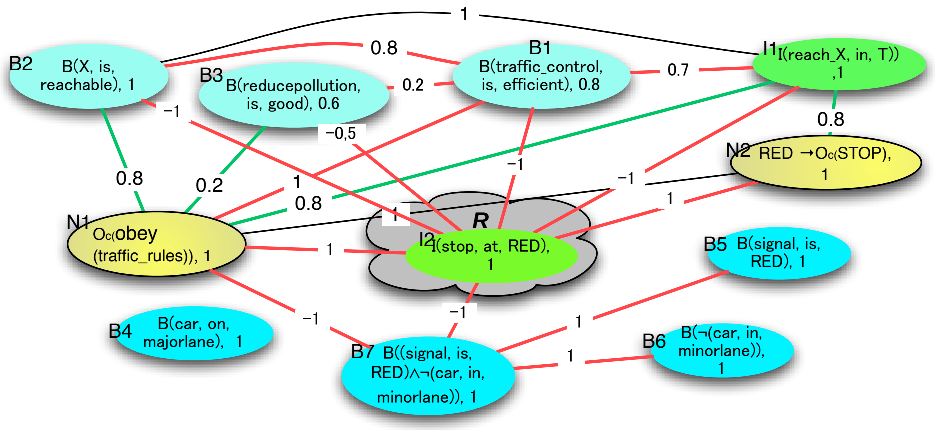
When a encounters the "RED" signal, and observes the traffic, its belief graph gets enriched with new information, and due to this addition of new beliefs, the strengths get modified. The new beliefs added to b are *a is at the Major lane*, *The signal is "RED"* and that *there are no cars on the minor lane*. The modified coherence graph is shown in Figure 5.

[10] The strength values and the degrees on beliefs and intentions are given manually respecting the constraints on the graph definition.



1
0.8
B2 B(X, is, reachable), 1
B3 B(reducepollution, is, good), 0.6
0.2
B1 B(traffic_control, is, efficient), 0.8
0.7
I1 I(reach_X, in, T)) ,1
0.8
−1
−0,5
N2 RED →Oc(STOP), 1
0.8
0.2
1
0.8
−1
−1
1
N1 Oc(obey (traffic_rules)), 1
1
1
I2 I(stop, at, RED), 1
B5 B(signal, is, RED), 1
−1
−1
1
B4 B(car, on, majorlane), 1
B7 B((signal, is, RED)∧¬(car, in, minorlane)), 1
1
B6 B(¬(car, in, minorlane)), 1

Fig. 5. Modified coherence graph



1
0.8
B2 B(X, is, reachable), 1
B3 B(reducepollution, is, good), 0.6
0.2
B1 B(traffic_control, is, efficient), 0.8
0.7
I1 I(reach_X, in, T)) ,1
0.8
−1
−0,5
N2 RED →Oc(STOP), 1
0.8
0.2
1
0.8
−1
−1
1
R
N1 Oc(obey (traffic_rules)), 1
1
1
I2 I(stop, at, RED), 1
B5 B(signal, is, RED), 1
−1
−1
1
B4 B(car, on, majorlane), 1
B7 B((signal, is, RED)∧¬(car, in, minorlane)), 1
1
B6 B(¬(car, in, minorlane)), 1

Fig. 6. maximizing coherence - $\mathcal{A} = b \odot i \odot n \setminus \{I2\}$

Now when trying to maximize the coherence, a discovers that if it removes the intention $I2 \rightarrow$ *to stop at RED signal* from the accepted set, he is able to maximize the coherence as in Figure 6. The total strength is $S(b{\odot}i{\odot}n_r, V, \emptyset) = 15.516$, Coherence of the graph is $C(b{\odot}i{\odot}n_r) = 23.716$ with $\mathcal{A} = \{B1, B2, B3, B4, B5, B6, B7, I1, N1, N2\}$ and dissonance $D(b \odot i \odot n_r) = .35$. Here the agent has a high enough dissonance to reject the intention $I2$, that is violate the norm. This example though simple, illustrates how an agent can act based on coherence maximization.

5 Related work

BDI theory is the most popular of the existing agent architectures. This architecture concentrates on the deliberative nature of the agent. There are several add ons to BDI architecture considering the recent developments in social and institutional agency, where the traditional cognitive model seems inadequate. They primarily include the addition of $norms$ to the cognitive concepts of *belief, desire, and intention*. The BOID architecture with the addition of obligation [2], and the work on deliberative normative agents [4] are the most prominent among them. In the BOID architecture the main problem is conflict resolution between and within the modules belief, desire, intention and obligation. Their focus is on architecture, while they do not specify any means to identify or resolve the conflict arising from the various interactions. They also do not have a structure of the cognitive modules, where the associations can be explored. The work by Castelfranchi in [4] again concentrates on the architecture. Their main contribution is the emphasis on agent autonomy. While most literature assume the strict adherence to the norms, they insist that it is an agent's decision whether to obey norms or not. As in the BOID architecture, they do not provide any mechanism by which an agent can violate a norm or reason about a norm violation. Another work by Lopez et al. [14] discusses how norm compliance can be ensured while allowing autonomy, using rewards and sanctions. Such mechanisms, while certainly complimenting our approach, only handle the issue at a superficial level and do not give the power to an agent to understand what it means to obey or violate a norm with respect to its cognitions. On the other hand, the work of Pasquier et al [9] is the first to our knowledge that attempts to unify the theory of coherence with the BDI architecture. The authors propose the theory as a reasoning mechanism associated with agent interaction such as when to dialogue based on the associated utility. In their work, the details of how coherence is calculated is not clear, nor do they provide a formalism based on coherence theory, but rather use the BDI framework.

Thagard, who proposed the coherence theory as constraint satisfaction [11] has applied his theory to explain many of the natural phenomena. But so far has not given a formal specification and integration into other theories. And finally, there is no work which gives a coherence framework to reason about agents and institutions, individually and together.

6 Discussion and Future work

In this paper, we have formally defined the basic coherence tools for building institutional agents. We aim to further develop this theory in the following directions.

An important question we have left unanswered in the paper is given the beliefs or norms how their corresponding coherence graphs can be created. Evaluating the association between two atomic beliefs looks more like a human task, yet we can use similarity measures extracted from other repositories like ontologies, Wordnet or search results. Whereas evaluating associations between complex beliefs, we can use the underlying logic. We plan to explore these ideas in more detail in our future work.

In the present work, we have provided the basic reasoning tools for a norm aware agent. We have shown when and how an autonomous agent could violate a norm. From the institutional perspective, a series of norm violations should trigger further actions, such as an analysis of why the norm is being violated. This could lead to a norm revision leading to an institutional redefinition. Our future work involves further exploration into questions related to norm violation from an institutional perspective.

We have simplified the representation of norms in the present work. In the future, we plan to have a more expressive representation of norms which includes the state of action when the norm is applicable, objectives behind the norm and the values promoted by the norm, borrowing the ideas developed in [1].

And finally, a coherence maximization may not only lead to a norm violation, but can also trigger a belief update, leading to the process of evolution of cognition. There are no widely accepted theories on how a cognitive agent can be evolved. The proposed theory helps to understand when a belief revision is profitable. In the future work, we propose to further explore cognitive revision in an institutional agent.

Acknowledgments. This work is supported under the OpenKnowledge[11] Specific Targeted Research Project (STREP), which is funded by the European Commission under contract number FP6-027253. Schorlemmer is supported by a *Ramon y Cajal* research fellowship from Spain's Ministry of Education and Science, which is partially funded by the European Social Fund. Special acknowledgments to all the reviewers of COIN@DURHAM07 for their detailed reviews and insightful comments.

References

[1] K. Atkinson. *What Should We Do?: Computational Representation of Persuasive Argument in Practical Reasoning.* PhD thesis, University of Liverpool, 2005.

[2] Jan Broersen, Mehdi Dastani, Joris Hulstijn, Zisheng Huang, and Leendert van der Torre. The BOID architecture: conflicts between beliefs, obligations, intentions and desires. In *AGENTS '01*, 2001.

[3] A. Casali, L. Godo, and C. Sierra. Graded BDI models for agent architectures. In *lecture notes in computer science*, volume 3487, 2005.

[4] Cristiano Castelfranchi, Frank Dignum, Catholijn M. Jonker, and Jan Treur. Deliberative normative agents: Principles and architecture. In *ATAL '99*, 2000.

[5] Leon Festinger. *A theory of cognitive dissonance*. Stanford University Press, 1957.

[6] Lou Goble and John-Jules Ch. Meyer. Deontic logic and artificial normative systems. In *DEON 2006*, 2006.

[7] Justin Yifu Lin. An economic theory of institutional change: induced and imposed change. *Cato Journal*, 9(1), 1989.

[8] John Von Neumann and Oskar Morgenstern. *Theory of Games and Economic Behavior*. Science Editions, J. Wiley, 1964.

[9] Philippe Pasquier and Brahim Chaib-draa. The cognitive coherence approach for agent communication pragmatics. In *AAMAS '03*, 2003.

[10] John R. Searle. *The Construction of Social Reality*. Free Press, 1997.

[11] Paul Thagard. *Coherence in Thought and Action*. MIT Press, 2002.

[12] Francesco Vigan, Nicoletta Fornara, and Marco Colombetti. An operational approach to norms in artificial institutions. In *AAMAS '05*, 2005.

[13] G. H. von Wright. *An Essay in Deontic Logic and the General Theory of Action : with a Bibliography of Deontic and Imperative Logic*. North-Holland Pub. Co, 1968.

[14] Fabiola López y López, Michael Luck, and Mark d'Inverno. Constraining autonomy through norms. In *AAMAS '02: Proceedings of the first international joint conference on Autonomous agents and multiagent systems*, 2002.

[11] http://www.openk.org

Organisational Artifacts and Agents For Open Multi-Agent Organisations: "Giving the power back to the agents"

Rosine Kitio[1], Olivier Boissier[1] *, Jomi Fred Hübner[2], and Alessandro Ricci[3]

[1] SMA/G2I/ENSM.SE, 158 Cours Fauriel
42023 Saint-Etienne Cedex, France
{kitio,boissier}@emse.fr

[2] GIA/DSC/FURB, Braz Wanka, 238
89035-160, Blumenau, Brazil
jomi@inf.furb.br

[3] DEIS, ALMA MATER STUDIORUM Università di Bologna
47023 Cesena (FC), Italy
a.ricci@unibo.it

Abstract. The social and organisational aspects of agency have have become nowadays a major focus of interest in the MAS community, and a good amount of theoretical work is available, in terms of formal models and theories. However, the conception and engineering of proper organisational infrastructures embodying such models and theories is still an open issue, in particular when open MAS are considered. Accordingly, in this paper we discuss a model for an organisational infrastructure called ORA4MAS that aims at addressing these issues. By being based on the A&A (Agents and Artifacts) meta-model, the key and novel aspect introduced with ORA4MAS is that organisations and the organisation infrastructure itself are conceived in terms of agents and artifacts, as first-class basic abstractions giving body to the MAS from design to runtime. This is in analogy with human organisation and organisation infrastructures, that are populated by humans (as participants and part of the organisation machinery), and by rich sets of artifacts and tools that humans use to support their activities inside the organisation and the organisation itself, encapsulating essential infrastructure services.

Keywords: Multi-agent Systems, MAS organisations, Open systems, Artifacts.

1 Introduction

Nowadays, current applications of IT show the interweaving of both human and technological *communities* (e.g. pervasive computing and ambient intelligence [15]), resulting in the construction of *connected* communities (ICities [30])

* Partially supported by USP-COFECUB.

in which software entities act on behalf of users and cooperate with infohabitants, taking into account issues like trust, security, flexibility, adaptation and *openness*. As shown in [20], current applications have led to an increase in number of agents, in the duration and repetitiveness of their activities, with a decision and action perimeter still enlarging. Moreover the number of agents' *designers* is also increasing, leading to a huge palette of heterogeneity in these systems. The complex system engineering's approach needed to build such applications highlights and stresses requirements on *openness* in terms of ability to take into account several kinds of changes and to adapt the system configuration while it keeps running.

In this paper, we are interested in social and organisational aspects of agency. They have always been a topic of study in the multiagent domain since the seminal work of [12, 7] and have become a major focus of interest in the MAS community (e.g. [21, 4]). However, most designers have doubts about how to put these concepts in practice, i.e., how to program them, while both addressing the openness and scalability issues and keeping agent's autonomy and decentralization which are essential features of MAS. Addressing the requirements stated above at the organisation level leads to a shift of design and programming paradigm. We denote it as a shift from closed to *open* organisations introducing the need for agents' systems that: (i) allow agents to arrive/leave dynamically into/from it, (ii) permit the dynamic change of both agents' individual behaviors in response to evolving environmental constraints and agents' *social* or *collective* behaviors due to changes of the systems' goals, and (iii) move from off-line to *on-line* adaptation, in the sense that designers should be replaced by the *agents* themselves to control and to realize the above issues.

Since it is a huge and complex work to develop systems with this kind of openness, in this paper we propose an organisational infrastructure referred as ORA4MAS which is meant to provide a conceptual and architectural step towards the simplification of this problem. Our proposal is based on the A&A approach where instead of a lot of different components and concepts (e.g., agents, services, proxies, objects, ...), only two types of entities are involved: agents and artifacts. Roughly, while agents model the decisions of the system, the artifacts model its functions. We especially demonstrate this approach showing how the organisational aspect of the MAS can be conceived and designed by only *organisational agents* and *organisational artifacts*. This is in analogy with human organisation and organisation infrastructures, that are populated by humans (as participants and part of the organisation machinery), and by rich sets of artifacts and tools that humans use to support their activities inside the organisation and the organisation itself, encapsulating essential infrastructure services.

In the first part of the paper (cf. sec. 2), we will have a look at the different approaches that have been developed in the field of multi-agent organisation, stressing what limitations we consider. This is complemented by a look at what has been done in the other dimensions of a MAS, i.e. environment and interaction. Then, we present the basic concepts underlying ORA4MAS infrastructure (cf. sec. 3), and we briefly describe the shapes of the organisational artifacts devised in ORA4MAS reifying the $\mathcal{M}$OISE$^+$ organisational model (cf. sec. 4). Finally, we provide concluding remarks and perspectives for the work in (cf. sec. 5)

2 Background

The recent developments in MAS domain, belonging to what we call *Organisation Oriented Programming (OOP)* [3], have provided many proposals of organisation-oriented middleware. In the different approaches related to OOP, we distinguish two important components: an declarative *organisation modeling language* (OML) and an *Organisation Implementation Architecture* (OIA). The OML *specifies* the organisation(s) of a MAS. It is used to collect and express specific constraints and cooperation patterns imposed on the agents by the designer (or the agents), resulting in an explicit representation that we call *organisation specification* (OS). A collective entity, called *Organisation Entity* (OE), instantiates this OS by assigning agents to roles. The OIA will then help these agents to properly "play" their roles as they are specified in the OS.

The OIA considers both an agent centered and a system centered point of view [4]. In the former, the focus lies on how to develop different *agent reasoning mechanisms* to interpret and reason on the OS and OE applying on the agents (e.g. [5, 6]). In the latter, the main concern is how to develop an infrastructure, that we call Organisation Infrastructure (OI), that ensures the satisfaction of the organisational constraints (e.g., agents playing the right roles, following the specified norms). This second point of view is important in heterogeneous and open systems where the agents that enter into the system may have unknown architectures. Of course, to develop the overall MAS, the former point of view is necessary since the agents probably need to have access to an organisational representation that enable them to reason about it.

The implementation of OI normally follows a common trend in multiagent platforms like JADE [2] and FIPA-OS [1]. These platforms have demonstrated the requirement and utility of the notion of "infrastructure" for MAS development [13]. Not only have they supported the implementation of the agents, but are being noticed as a provider of fundamental global generic services going further of only directory facilitator, agent management system or agent communications by also addressing coordination [24]. Therefore, agents related to the application domain operate on top of a middleware layer.

As shown in [3], many implementations of the OI follow the general layered architecture depicted in Fig. 1: (i) domain (or application) agents, responsible to achieve organisational goals and use an *organisational proxy* component to interact with the organisation, (ii) *organisational layer*, responsible to bind all agents in a coherent OS and OE and provides some services for them, and (iii) communication layer for connecting all components of the infrastructure

[4] We prefer here system-centred to organisation-centred in order to avoid confusion even if, as we have seen, the organisation is reified in OE. Let's notice that in [33] these points of view are called agent and institutional perspectives.

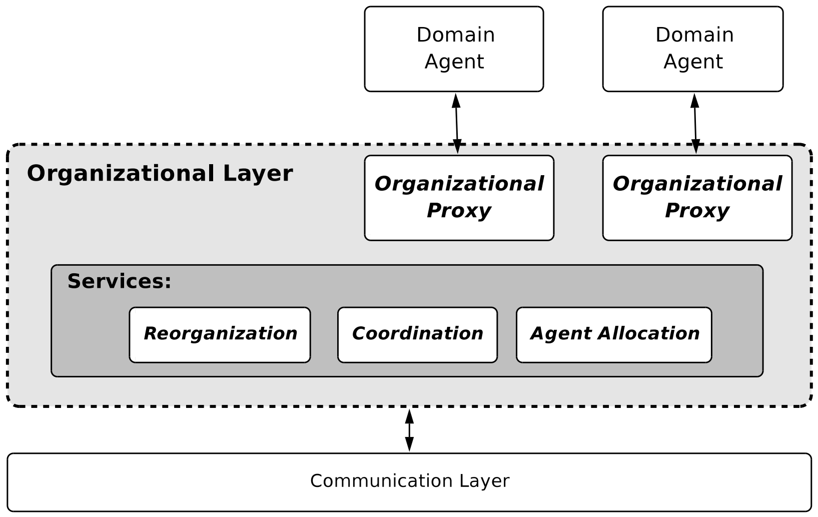


Fig. 1. Common Organisation Infrastructure (OI) for open MAS

in a distributed and heterogeneous applications. This layered structure results in an engineering approach where the MAS development is considered to be addressed by three kinds of designers: domain or application designers (for the agents and the specification of the OS using the OML), MAS or OI designers (for the organisational layer and OE management), and communication designers.

From the study of the different works considering this organisation layer, we can identify a set of specialized services and proxies (e.g., angels [8], governors [10], managers [18]). In order to stress their ability to manage organisational concepts and to develop dedicated reasoning/processing abilities on the organisation, let's call them respectively *organisational services* (OrgServices). One important point to notice is that all the access to the OE by the agents is mediated by these OrgServices in the OI.

This brief general introduction of OI designs allow us to point out some drawbacks:

1. In some proposals, like $\mathcal{S}$-$\mathcal{M}$OISE^{+} [18], OrgServices are implemented as agents. The problem is that, conceptually, services are not in the same abstraction level as agents.
2. In the proposals where OrgServices are not agents, whenever an application designer needs to customise some decisions of the system in the organisational dimension (e.g., a sanction system, a reorganisation strategy, the allocation of agents to roles), s/he has to develop/change an OrgService. It can be quite confusing to deal with both OrgServices and agents concepts while developing a system. It will be better to always use the same abstraction level when modelling and implementing the decision aspect of system.
3. The designer (and the agents) also have to deal with two kinds of environments: a virtual organisational environment (where the agents adopt roles, send messages) and the real environment (where the agents act). An unified view of the environment simplifies the concept of agent interaction.

4. In the general architecture of Fig. 1, the middleware has too much power. Most of the organisational "decisions" are performed at this layer. It is more suitable if the agents make decisions and not the OrgServices. For example, if some agent wants to perform some action or send a message that its organisation does not allow, it can not do it since the middleware (and its organisational proxy) will detect this violation tentative and deny it. The middleware is thus performing two functions: detection and decision. In some cases agents operating on the application layer should get their control power back in the sense that they could play some of the roles of the OrgServices. As another example, reorganisation requires that *agents* should be able to manage and access the creation of new organisations.
   The problems of existing approaches of organisations are consequence of some properties of the organisational managers design: (i) the enforcement of organisational functions and constraints and (ii) the inclusion of reasoning and decision aspects that can be managed by agents and thus should be in the agent layer.

It's worth noting that the issues stated in this section do not concern solely the implementation level, but also the conceptual and theoretical level: what is the nature of OrgServices in MAS where only agents are considered as first-class entities?

3 An Organisational Infrastructures based on Agents and Artifacts

The proposal presented in this paper draws its inspiration from human organisation infrastructures. Human organisation and organisation infrastructures, that are populated by humans (as participants and part of the organisation machinery), and by rich sets of artifacts and tools that humans use to support their activities inside the organisation and the organisation itself, encapsulating essential infrastructure services. According to psycho-sociological theories and studies such as Activity Theory and Distributed Cognition [22]—recently adopted in computer science fields such as CSCW, HCI and MAS [32, 31, 27]—the notion of artifact (and tool, taken here as a synonym) plays a key role for the overall sustainability of an organisation and the effectiveness and efficiency of activities taking place inside the organisation.

In particular, some of these artifacts—that we call here *organisational artifacts*—appear to be vital for supporting the coordination of organisation processes and management: for instance by making more effective the communication among the members of an organisation (e.g. the telephone, instant-messaging services, chat-rooms), by providing information useful for orienting the activities of organisation participants (e.g. signs inside a building), by coordinating participants (e.g. queue systems at the post-office), by controlling access to resources and enforcing norms (e.g. the badge used by members in a computer science department to access certains rooms or use some other artifacts, such as copiers). Human societies and organisations continuously improve

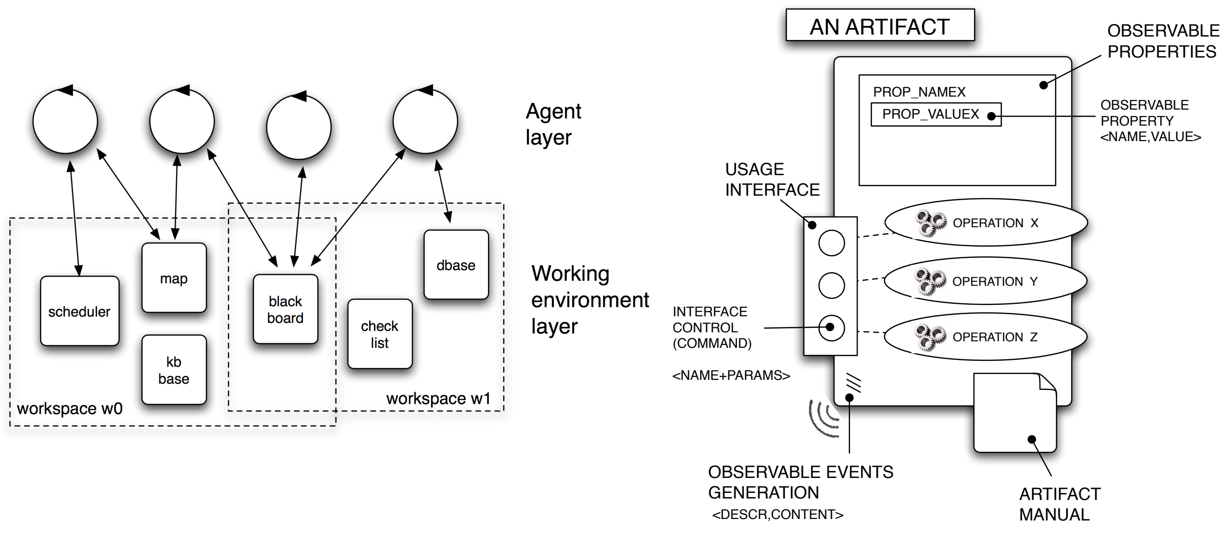


Fig. 2. *(Left)* abstract representation of workspaces, populated by agents—represented by circles—and artifacts—represented by squares. *(Right)* A representation of the main parts and properties of an artifact, with in evidence the usage interface, the observable properties and the manual.

their experience in designing artifacts more and more effective to support both organisation participation—helping members to cope with the complexity of social activities and work—and organisation management—helping managers to monitor and control the organisation behaviour as a whole.

Analogously, here we propose an organisational infrastructure called ORA4MAS in which both organisations and the organisation infrastructure itself are conceived and engineered in terms of a set of agents and *artifacts*. ORA4MAS in particular exploits the notion of artifact as introduced by the A&A meta-model [28] to encapsulate as first-class abstractions from design to runtime structures, and related functionalities / rules as defined theoretically by the $\mathcal{M}$OISE^{+} organisational model [17].

In the remainder of the section, first we recall the basic ideas provided by the A&A meta-model, and then describe how such concepts are exploited to shape the ORA4MAS infrastructure.

3.1 The Notion of Artifacts in MAS

The notion of *MAS environment*, as remarked by recent literatures, has gained a key role in the recent past, becoming a *mediating* entity, functioning as enabler but possibly also as a manager and constrainer of agent actions, perceptions, and interactions (see here [34] for comprehensive surveys). According to such a perspective, the environment is not a merely passive source of agent perceptions and target of agent actions—which is, actually, the dominant perspective in agency and in MAS—, but a first-class abstraction that can be suitably designed to *encapsulate* some fundamental functionalities and services, supporting MAS dimensions such as coordination and organisation, besides agent mobility, communications, security, etc.

Among the various approaches, the A&A in particular introduces a notion of *working environment*, representing such a part of the MAS explicitly designed on the one hand by MAS engineers to provide various kinds of functionality—including MAS coordination, organisation—and perceived as first-class entity on the other hand by agents of the MAS [29, 28, 25]. By drawing its inspiration from human society and theories such as Activity Theory and Distributed Cognition, A&A working environment are made of *artifacts*, introduced as first-class abstraction representing function-oriented dynamic entities and tools that agents can create and use to perform their individual and social activities.

Artifacts can be considered as a complimentary abstraction to agent populating a MAS: while agents are goal-oriented pro-active entities, artifacts are a general abstraction to model function-oriented passive entities, designed by MAS designers to encapsulate some kind of functionality, by representing (or wrapping existing) resources or instruments mediating agent activities. Passive here means that—differently from the agent case—they do not encapsulate any thread of control.

As artifacts and tools in human societies play a key role in mediating any not naive social activities, analogously artifacts in MAS are meant to play an important role in supporting the activities performed inside MAS organisations. Among the others, *coordination artifacts* have been introduced as an important class of artifacts for MAS organisations [26], as artifacts mediating agent interactions and encapsulating some kind of coordinating functionality— whiteboards, event services, shared task schedulers are examples.

Figure 2 shows an abstract representation of an artifact as defined in the A&A meta-model, exhibiting analogous parts and properties of artifacts as found in human society. The artifact *function*—and related artifact behaviour—is partitioned in a set of *operations*, which agents can trigger by acting on artifact *usage interface*. The usage interface provides all the controls that make it possible for an agent to interact with an artifact, that is to *use* and *observe* it. Agents can use an artifact by triggering the execution of operations through the usage interface and by perceiving *observable events* generated by the artifact itself, as a result of operation execution and evolution of its state. Besides the controls for triggering the execution of operation, an artifact can have some *observable properties*, i.e. properties whose value is made observable to agents, without necessarily executing operations on it. The interaction between agents and artifacts strictly mimics the way in which humans use their artifacts: let's consider a coffee machine, for a simple but effective analogy. The set of buttons of the coffee machines represents the usage interface, while the displays that are typically used to show the state of the machine represent artifact observable properties. The signals emitted by the coffee machine during its usage represent observable events generated by the artifact.

Analogously to the human case, in A&A each artifact type can be equipped by the artifact programmer with a *manual* composed essentially by the *function description*—as the formal description of the purpose intended by the designer—, the *usage interface description*—as the formal description of artifact usage interface and observable states—, and finally the *operating instructions*—as the formal description of how to properly use the artifact so as to exploit its functionalities. Such a manual is meant to be essential for creating open systems with intelligent agents that dynamically discover and select which kind of artifacts could be useful for their work, and then can use them effectively even if they have not pre-programmed by MAS programmers for the purpose.

Finally, artifacts can be composed together by means of *link interface*s, that are sets of input / output ports that can be (dynamically) linked together by means of suitable channels and through which artifacts can exchange data. Link interfaces serve two purposes: on the one side, to explicitly define a principle of composability for artifacts, enabling the ruled construction of articulated and complex artifacts by means of simpler ones; on the other side, to realise distributed (composed) artifacts: channels can connect link interfaces of artifacts possibly belonging to different workspaces.

3.2 ORA4MAS Infrastructure: The Basic Idea

The basic idea in ORA4MAS is to engineer the organisational infrastructure—and the organisations living upon it—in terms of agents and artifacts, following the basic A&A metamodel.

Here we use the terms *organisational agents* and *organisational artifacts* to identify those agents and artifacts of the MAS which are part of the organisational infrastructure, and that are responsible of activities and encapsulate functionalities concerning the management and enacting of the organisation. In particular, organisation agents—analogously to managers and administrators in human organisation—are responsible of management activities inside the organisation, concerning observing, monitoring, and reasoning about organisation dynamics, etc. Such activities take place almost by creating and managing organisational artifacts that are then used by member agents of the organisation. Organisation artifacts are those artifacts that agents of an organisation may want or have to use in order to participate in organisation activities and access to organisation resources, encapsulating organisation rules and functionalities, such as enabling and mediating (ruling) agent interaction, tracing and ruling resource access, and so on. The overall picture accounts for organisational agents that dynamically articulate, manage and adapt the organisation by creating, linking and manipulating the organisational artifacts, which are discovered and used by the member agents to work inside the organisation.

Even from this abstract characterisation, it is possible to identify some general properties that are of some importance to face the drawbacks listed at the end of Section 2:

– *Abstraction & encapsulation*—By using agents and artifacts to reify both the organisation and the organisation infrastructure—from design to runtime—, we raise the level of abstraction with respect to approaches in which organisation mechanisms are hidden at the implementation / middleware level. Such mechanisms become parts of the agent world, suitably encapsulated in proper entities that agents then can inspect, reason and manipulate, by adopting a uniform approach.

– *"Power back to agents"*—Decision functionalities that were embedded in the OrgServices in the OI go back to the agents' layer in organisational agents. Agents are autonomous with respect to decision of using or not a specific artifact—including the organisational artifacts—and keeps its autonomy—in terms of control of its actions—while using artifacts. Agents however can depend on the functionalities provided (encapsulated) by artifacts, which can concern, for instance, some kind of mediation with respect to the other agents co-using the same artifact. Then, by enforcing some kind of mediation policy an artifact can be both an enabler and a *constrainer* of agent interactions. However, such a constraining functioning can take place without compromising the autonomy of the agents, who are fully encapsulating their control.
– *Distributed management*—Distributing the management of the organisation into different organisational artifacts installs a distributed coordination (meaning here more particularily synchronization) of the different functions related to the management of the organisation. Completing this distribution of the coordination, the reasoning and decision processes which are encapsulated in the organisational agents may be also distributed among the different agents. Thanks to their respective autonomy, all the reasoning related to the management of the organisation (monitoring, reorganisation, control) may be decentralized into different loci of decision with a loosely coupled set of agents.
– *Openness*—Organisational artifacts can be created and added dynamically according to the need. They have a proper semantics description of both the functionalities and operating instructions, so conceptually agents can discover at runtime how to use them in the best way. Related to openness, the approach promotes heterogeneity of agent (societies): artifacts can be used by heterogeneous kinds of agents, with different kinds of reasoning capabilities. Extending the idea to multiple organisations, we can have the same agents playing different roles in different organisations, and then interacting with organisational artifacts belonging to different organisations.
– *Re-organisation and autonomic-properties*—The basic properties of organisational agents and artifacts can be effective in devising scenarios in which the MAS supports forms of self-organisation (and configuration, healing, protection). On the one side we have organisation artifacts that are inspectable—in terms of their manual (static) and observable state (dynamic)—and *malleable*, i.e. they can be designed so as to be manipulated and adapted at runtime. On the other side we have organisation agents that can have suitable reasoning capabilities so as to observe, reason and manipulate organisation artifacts according to the needs. This is particularly important when organisational artifacts mediating the interaction of groups of agents are considered: by observing and changing the mediating behaviour of such kinds of artifacts, agents are able to change the collective behaviour of overall groups of agents. In other words, here the re-organisation process

is modelled as an organisation process itself, in the same vein as proposed in [16].

After sketching the basic concepts underlying the ORA4MAS approach, in next section we finally describe how a full-fledged organisational model—$\mathcal{M}$OISE$^+$ in this case—can be abstractly implemented on top of agents and artifacts.

4 Shaping ORA4MAS Artifacts Upon $\mathcal{M}$oise$^+$

4.1 The $\mathcal{M}$oise$^+$ Model

$\mathcal{M}$OISE$^+$ (Model of Organisation for multI-agent SystEms) [17] is an OML that explicitly decomposes the organisation into structural, functional, and deontic dimensions. The structural dimension defines the *roles*, *groups*, and *links* of the organisation. The definition of roles states that when an agent decides to play some role in a group, it is accepting some behavioural constraints related to this role. The functional dimension describes how the *global collective goals* should be achieved, i.e., how these goals are decomposed (in global *plans*), grouped in coherent sets (by *missions*) to be distributed to the agents. The decomposition of global goals results in a goal-tree where the leafs-goals can by achieved individually by the agents. The deontic dimension is added in order to binds the structural dimension with the functional one by the specification of the roles' *permissions* and *obligations* for missions. Instead of being related to the agents' behavior space (what they can do), the deontic dimension is related to the agents' autonomy (what they should do).

$\mathcal{S}$-$\mathcal{M}$OISE$^+$ is an open source implementation of an OI that supports the $\mathcal{M}$OISE$^+$ OML [18]. The organisational proxy is called *OrgBox* and it consists of an API that agents use to access the OrgServices, provided by a special systen agent called *OrgManager*. The OrgManager receives and manages all the messages from the agents' OrgBox asking for changes in the OE state (e.g. role adoption, group creation, mission commitment). Those changes bring about the OrgManager to modify the OE only if they do not violate any organisational constraint. For instance, when some agent asks for a role adoption in a group g, the OrgManager ensures that: (1) the role belongs to a specified group g; (2) the number of players in g is lesser or equals than the maximum number of players defined in the group's compositional specification; (3) each role ρ_i that the agent already plays is specified as compatible with the new role in g. Besides the organisational compliance, the OrgManager also provides useful information for the agents' organisational reasoning and coordination, for example the missions they are forced to commit to and goals it can pursue.

An important feature of this architecture is that the OS may be interpreted at run-time by the agents because its specification is available to them. Thus agents can be developed as hardwired programmed for some particular OS or they can be programmed to interpret the current available OS. This last feature is not only useful in open systems, but also when one considers a reorganisation process, since the adaptation of the new OS may be dynamic. $\mathcal{S}$-$\mathcal{M}$OISE$^+$ does not require any specific type of agent architecture. Although agents normally use the OrgBox to interact with the system, an agent could even interact with the OrgManager directly using KQML.

4.2 Organisational Agents and Artifacts based on $\mathcal{M}$oise$^+$

We exploit here the $\mathcal{M}$OISE$^+$ model to identify and shape a basic set of organisational artifacts (kind) and agents that constitute the basic infrastructure building blocks of ORA4MAS, being a sort of "reification" of the SS, FS, DS specifications (see figure 3). This basic set accounts for:

– an OrgBoard artifact—used to keep track of the structure of organisation in the overall;
– a GroupBoard artifact—used to manage the life-cycle of a specific group;
– a SchemeBoard type—used to support and manage the execution of a social scheme.

Here we consider just a core set, skipping most details that would make heavy the overall understanding of the approach: the interested reader is forwarded on this technical report [19] to get further details.

In the following we briefly describe the basic characteristics of these kinds of artifact. In the description, the operations (commands) enlisted in artifact usage interface are abstractly described by a name with input parameters, followed (optionally) by a set of the observable events possibly generated by the operation execution (only events significant for artifact specific functionalities are considered, skipping those generated by default by the artifact). Observable properties are represented just by a name, which corresponds to the name of the property.

OrgBoard **Artifacts.** A simple abstract model for the OrgBoard artifact is depicted in figure 3 (left). The usage interface is composed by operations to:

– enter the organisation: enterOrg;
– leave the organisation: leaveOrg;
– register / de-register a new group: registerGroup(G,GB), removeGroup(G)—where G is an identifier for a group and GB is the identifier of the related group board artifact;
– register / de-register a new scheme: registerScheme(S,SB), removeScheme(S) where S is the identifier for a schema and SB is the identifier of the related scheme board.

Among the observable properties:

– list of current groups: current-groups property;
– list of current schemes: current-schemes property;
– organisation specification (including SS, FS, DS): org-spec property.

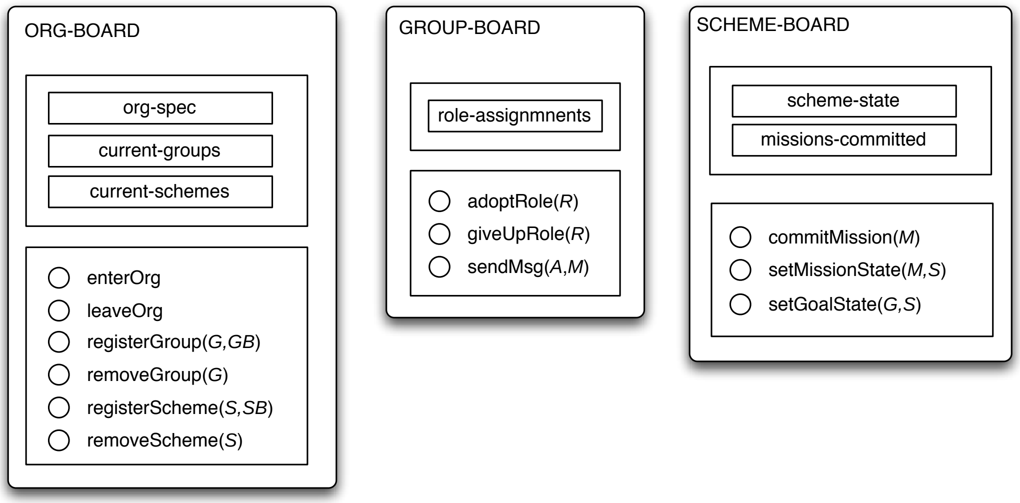


Fig. 3. Basic kinds of artifacts in ORA4MAS, with in evidence their usage interface, including operations and observable properties

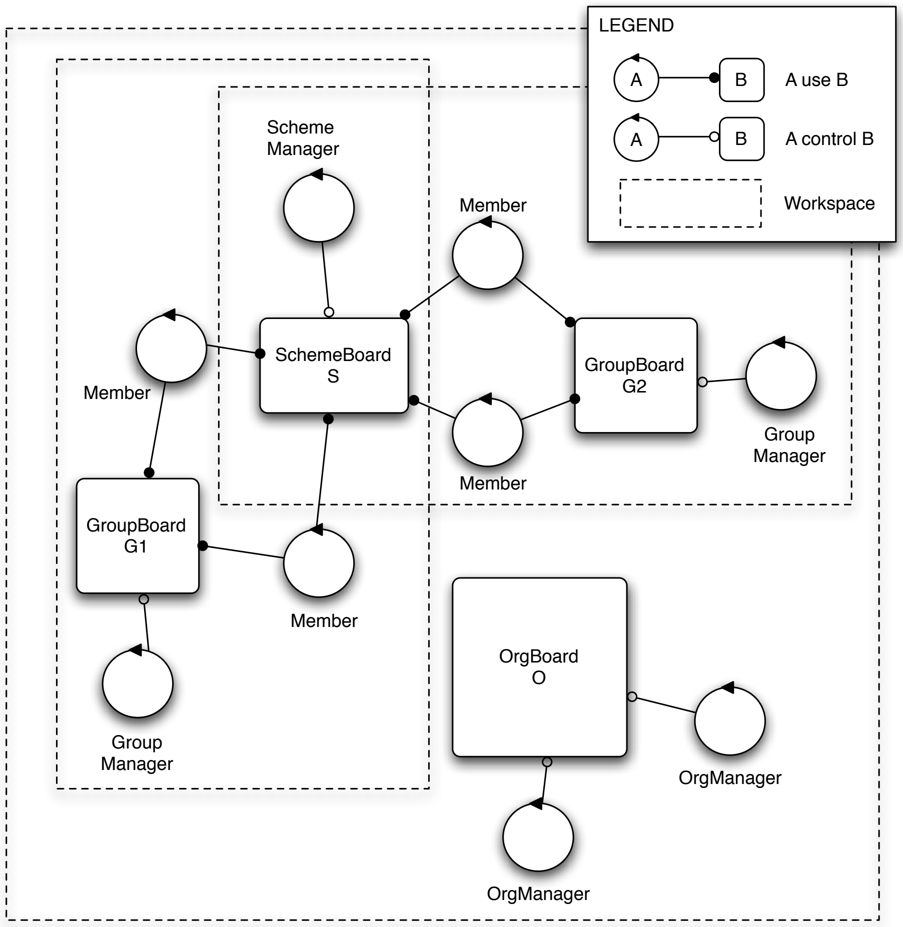


Fig. 4. A simple example of instance of an $\mathcal{M}$OISE$^+$ organisation with 2 groups and a running social scheme, with in evidence the artifacts used

Generally speaking, the observable properties of the artifact make it possible—for agents observing an OrgBoard —to monitor and be aware of which agents are actually participating to the organisation, and which are the schemes and groups instantiated. Also, this artifact can be inspected to know which are the SS, FS, DS currently adopted in the organisation.

GroupBoard **Artifacts.** The GroupBoard artifact type (see figure 3, center) is instantiated upon a specific instance of SS and DS, and provides functionalities to manage a group in terms of set of available roles and agents participation, according to the specific structure and strategy specified in SS and DS. For what concerns the DS, the artifact enforces those deontic rules that entail permissions and obligations in role adoptions and give up for the agents.

The usage interface accounts for the following operations:

- adopt a new role: adoptRole(R):{role_adoption_ok,role_adoption_failed}, where R is the identifier for a role;
- give up a role: giveUpRole(R):{role_giveup_ok,role_giveup_failed};
- sending a message to a specific agent or all the agents part of the group: sendMsg(A,M), sendMsg(M), where A is the identifier for the receiver agent, m is the message content.

Among the observable properties:

- role assignments: role-assignments property.

By observing a GroupBoard artifact, an agent can monitor and be aware of the role-agent assignments inside the group.

SchemeBoard **Artifacts.** The SchemeBoard artifact type (see figure 3, right) is instantiated upon a specific instance of FS and DS, and provides functionalities to manage the execution of a social scheme, coordinating the commitments to missions and goals, and their interaction. It function as a coordination artifact, automating the management of the dependencies between the missions and the goals as described by the social scheme, and embedding such part of the deontic specification concerning permissions and obligations for agents to commit to missions. The usage interface provides commands to:

- commit to a mission: commitMission(M):{commit_ok, commit_failed}, where M is the identifier for a mission;
- set mission state: setMissionState(M,S), where M is the identifier for a mission and S can be either *completed* or *failed*;
- set goal state: setGoalState(G,S), where G is the identifier for a goal and S can be either *satisfied* or *impossible.*

Among the observable properties:

- scheme dynamic state: scheme-state property, that includes all the goals of the scheme and their state;

– list of the current missions committed: missions-committed property.

By observing a SchemeBoard artifact, an agent can monitor then the overall dynamics concerning the scheme execution, and the be aware of which missions are assigned to which agents.

Organisational Agents. The organisational agents introduced are essentially managers responsible to create and manage the organisational artifacts described previously. Such activities typically include observing artifacts dynamics and possibly intervening, by changing / adapting artifacts or interacting directly with agents, so as to improve the overall (or specific) organisation processes or taking some kinds of decisions when detecting violations. As an example, one or multiple *scheme managers* agents can be introduced, responsible of monitoring the dynamics of the execution of a specific running scheme by observing a specific SchemeBoard instance. The SchemeBoard artifact and scheme manager agents can be designed so as that the artifact allows for violation of the deontic rules concerning the commitment of missions by agents playing some specific roles, and then the decision about what action to take—after detecting the violation—can be in charge of the manager agent.

4.3 Towards a Concrete Architecture

ORA4MAS concrete architecture is realised on top of CARTAGO infrastructure, embedding algorithms used in $\mathcal{S}$-$\mathcal{M}$OISE$^+$. CARTAGO (Common ARtifact Infrastructure for AGent Open environment) is a MAS infrastructure based on the A&A meta-model, providing the capability to define new artifacts types, suitable API for agents to work with artifacts and workspaces, and a runtime supporting the existence and dynamic management of working environments. CARTAGO is meant to be integrated with existing cognitive MAS architectures and models / languages / platforms, so as to extend them to create and work with artifact-based environments. A first example of integration with the *Jason* agent programming platform is briefly described here [28]. CARTAGO technology is based on Java and are available as open-source projects freely downloadable from the project web sites[5].

The engineering of the first prototype of the ORA4MAS infrastructure upon CARTAGO is still a work in progress.

5 Conclusion and Perspectives

In this paper, we have followed the A&A approach to give back the power to agents in an organisational approach. From this perspective, we have defined on the one hand the organisational artifacts which encapsulate the functional aspects of an organisation and organisation management, and on the other hand

[5] http://www.alice.unibo.it/cartago

the organisational agents, which encapsulated the decision and reasoning side of the management of organisations. We have thus designed the ORA4MAS model. With this proposal, we provide a decentralized management of an organisational entity.

Extensions to this work includes the instantiation of OrgArts to different OMLs such as ISLANDER [9] or $\mathcal{M}$OISE^{Inst} [14], or those like AGR [11]. Other extensions aim at taking benefit of the uniform concepts used to implement the environment and the organisation abstractions through the concept of artifacts. Such an homogeneous conceptual point of view will certainly help us to bind both concepts together in order to situate organisations in environment or to install the access to the environment into organisational models (in the same direction as proposed in [23]). Another point of investigation is the definition of a meta-organisation for the ORA4MAS, so that we have special roles for organisational agents that give them access to the organisational artifacts.

References

1. FIPA-OS. Technical report, Nortel Networks, 2000. (`http://www.nortelnetworks.com/products/announcements/fipa/`).
2. F. Bellifemine, F. Bergenti, G. Caire, and A. Poggi. JADE – a java agent development framework. In R. H. Bordini, M. Dastani, J. Dix, and A. El Fallah Seghrouchni, editors, *Multi-Agent Programming: Languages, Platforms, and Applications*, number 15 in Multiagent Systems, Artificial Societies, and Simulated Organizations, chapter 5. Springer, 2005.
3. O. Boissier, J. F. Hübner, and J. S. Sichman. Organization oriented programming from closed to open organizations. In G. O'Hare, M. O'Grady, O. Dikenelli, and A. Ricci, editors, *Engineering Societies in the Agents World VII*, volume 4457 of *LNCS*. Springer-Verlag, 2007.
4. O. Boissier, J. Padget, V. Dignum, G. Lindemann, E. Matson, S. Ossowski, J. Sichman, and J. Vázquez-Salceda, editors. *Coordination, Organizations, Institutions, and Norms in Multi-Agent Systems*, volume 3913 of *Lecture Notes in Artificial Intelligence*. Springer Verlag, 2006. AAMAS 2005 International Workshops on Agents, Norms, and Institutions for Regulated Multiagent Systems, ANIREM 2005 and on Organizations in Multi-Agent Systems, OOOP 2005, Utrecht, The Netherlands, July 25-26, 2005, Revised Selected Papers.
5. J. Broersen, M. Dastani, J. Hulstijn, Z. Huang, and L. der van Torre. The BOID architecture: conflicts between beliefs, obligations, intentions and desires. In J. P. Müller, E. Andre, S. Sen, and C. Frasson, editors, *Proceedings of the Fifth International Conference on Autonomous Agents*, pages 9–16, Montreal, Canada, 2001. ACM Press.
6. C. Castelfranchi, F. Dignum, C. M. Jonker, and J. Treur. Deliberate normative agents: Principles and architecture. In *Proceedings of The Sixth International Workshop on Agent Theories, Architectures, and Languages (ATAL-99)*, 1999.
7. D. D. Corkill. *A Frameworkfor Organizational Self-Design in Distributed Problem Solving Networks*. PhD thesis, University of Massachusetts, Amherst, 1983.
8. V. Dignum, J. Vazquez-Salceda, and F. Dignum. OMNI: Introducing social structure, norms and ontologies into agent organizations. In R. H. Bordini, M. Dastani, J. Dix, and A. El Fallah-Seghrouchni, editors, *Proceeding of the Programming Multi-Agent Systems (ProMAS 2004)*, LNAI 3346, Berlin, 2004. Springer.

9. M. Esteva, J. A. Rodriguez-Aguiar, C. Sierra, P. Garcia, and J. L. Arcos. On the formal specification of electronic institutions. In F. Dignum and C. Sierra, editors, *Proceedings of the Agent-mediated Electronic Commerce*, LNAI 1191, pages 126–147, Berlin, 2001. Springer.
10. M. Esteva, J. A. Rodríguez-Aguilar, B. Rosell, and J. L. AMELI: An agent-based middleware for electronic institutions. In N. R. Jennings, C. Sierra, L. Sonenberg, and M. Tambe, editors, *Proceedings of the Third International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'2004)*, pages 236–243, New York, 2004. ACM.
11. J. Ferber and O. Gutknecht. A meta-model for the analysis and design of organizations in multi-agents systems. In Y. Demazeau, editor, *Proceedings of the 3rd International Conference on Multi-Agent Systems (ICMAS'98)*, pages 128–135. IEEE Press, 1998.
12. M. S. Fox. An organizational view of distributed systems. *IEEE Transactions on Systems, Man, and Cybernetics*, 11(1):70–80, Jan 1981.
13. L. Gasser. Mas infrastructure: Definitions, needs and prospects. In *Revised Papers from the International Workshop on Infrastructure for Multi-Agent Systems*, pages 1–11, London, UK, 2001. Springer-Verlag.
14. B. Gâteau, O. Boissier, D. Khadraoui, and E. Dubois. Moiseinst: An organizational model for specifying rights and duties of autonomous agents. In *Third European Workshop on Multi-Agent Systems (EUMAS 2005)*, pages 484–485, Brussels Belgium, December 7-8 2005.
15. I. A. Group. Ambient intelligence: from vision to reality. Technical report, IST, 2003. ftp://ftp.cordis.europa.eu/pub/ist/docs/istag-ist2003_consolidated_report.pdf.
16. J. F. Hübner, O. Boissier, and J. S. Sichman. Programming MAS reorganisation with MOISE+. In J. Meyer, M. Dastani, and R. Bordini, editors, *Dagstuhl Seminar on Foundations and Practice of Programming Multi-Agent Systems*, volume 06261, 2006.
17. J. F. Hübner, J. S. Sichman, and O. Boissier. A model for the structural, functional, and deontic specification of organizations in multiagent systems. In G. Bittencourt and G. L. Ramalho, editors, *Proceedings of the 16th Brazilian Symposium on Artificial Intelligence (SBIA'02)*, volume 2507 of *LNAI*, pages 118–128, Berlin, 2002. Springer.
18. J. F. Hübner, J. S. Sichman, and O. Boissier. $\mathcal{S}$-$\mathcal{M}$OISE^{+}: A middleware for developing organised multi-agent systems. In O. Boissier, V. Dignum, E. Matson, and J. S. Sichman, editors, *Proceedings of the International Workshop on Organizations in Multi-Agent Systems, from Organizations to Organization Oriented Programming in MAS (OOOP'2005)*, volume 3913 of *LNCS*. Springer, 2006.
19. R. Kitio. Organizational artifacts and agents for open multi-agent systems, Jun 2007. Master Thesis report, Available at http://www.emse.fr/~boissier/kitio.
20. M. Luck, P. McBurney, O. Shehory, and S. Willmott. *Agent Technology: Computing as Interaction (A Roadmap for Agent Based Computing)*. AgentLink, 2005.
21. Modeling Autonomous Agents in a Multi-Agent World (MAAMAW'2001). *Pre-Proceeding of the 10th European Workshop on Modeling Autonomous Agents in a Multi-Agent World (MAAMAW'2001)*, 2001.
22. B. A. Nardi. *Context and Consciousness: Activity Theory and Human-Computer Interaction.* MIT Press, 1996.
23. F. Y. Okuyama, R. H. Bordini, and A. C. da Rocha Costa. Spatially distributed normative objects. In G. Boella, O. Boissier, E. Matson, and J. Vázquez-Salceda,

editors, *Proceedings of the Workshop on Coordination, Organization, Institutions and Norms in Agent Systems (COIN), held with ECAI 2006, 28th August, Riva del Garda, Italy.*, 2006.

24. A. Omicini, S. Ossowski, and A. Ricci. Coordination infrastructures in the engineering of multiagent systems. In F. Bergenti, M.-P. Gleizes, and F. Zambonelli, editors, *Methodologies and Software Engineering for Agent Systems: The Agent-Oriented Software Engineering Handbook*, volume 11 of *Multiagent Systems, Artificial Societies, and Simulated Organizations*, chapter 14, pages 273–296. Kluwer Academic Publishers, June 2004.
25. A. Omicini, A. Ricci, and M. Viroli. *Agens Faber*: Toward a theory of artefacts for MAS. *Electronic Notes in Theoretical Computer Sciences*, 150(3):21–36, 29 May 2006.
26. A. Omicini, A. Ricci, M. Viroli, C. Castelfranchi, and L. Tummolini. Coordination artifacts: Environment-based coordination for intelligent agents. In *AAMAS'04*, volume 1, pages 286–293, New York, USA, 19–23July 2004. ACM.
27. A. Ricci, A. Omicini, and E. Denti. Activity Theory as a framework for MAS coordination. In P. Petta, R. Tolksdorf, and F. Zambonelli, editors, *Engineering Societies in the Agents World III*, volume 2577 of *LNCS*, pages 96–110. Springer-Verlag, Apr. 2003.
28. A. Ricci, M. Viroli, and A. Omicini. A general purpose programming model & technology for developing working environments in MAS. In M. Dastani, A. El Fallah Seghrouchni, A. Ricci, and M. Winikoff, editors, *5th International Workshop "Programming Multi-Agent Systems" (PROMAS 2007)*, pages 54–69, AAMAS 2007, Honolulu, Hawaii, USA, 15 May 2007.
29. A. Ricci, M. Viroli, and A. Omicini. "Give agents their artifacts": The A&A approach for engineering working environments in MAS. In E. Durfee, M. Yokoo, M. Huhns, and O. Shehory, editors, *6th International Joint Conference "Autonomous Agents & Multi-Agent Systems" (AAMAS 2007)*, pages 601–603, Honolulu, Hawai'i, USA, 14–18 May 2007. IFAAMAS.
30. J. Sairamesh, A. Lee, and L. Anania. Introduction. *Commun. ACM*, 47(2):28–31, 2004.
31. K. Schmidt and C. Simone. Coordination mechanisms: Towards a conceptual foundation of CSCW systems design. *International Journal of Computer Supported Cooperative Work (CSCW)*, 5(2–3):155–200, 1996.
32. T. Susi and T. Ziemke. Social cognition, artefacts, and stigmergy: A comparative analysis of theoretical frameworks for the understanding of artefact-mediated collaborative activity. *Cognitive Systems Research*, 2(4):273–290, Dec. 2001.
33. J. Vázquez-Salceda, H. Aldewereld, and F. Dignum. Norms in multiagent systems: some implementation guidelines. In *Proceedings of the Second European Workshop on Multi-Agent Systems (EUMAS 2004)*, 2004.
34. D. Weyns and H. V. D. Parunak, editors. *Journal of Autonomous Agents and Multi-Agent Systems. Special Issue: Environment for Multi-Agent Systems*, volume 14(1). Springer Netherlands, 2007.

Knowledge Sharing Between Agents in a Transitioning Organization

Eric Matson[1][2] and Raj Bhatnagar[1]

[1] Wright State University
Department of Computer Science and Engineering
Dayton, OH, USA
[2] University of Cincinnati
Department of Computer Science
Cincinnati OH, USA
eric.matson@wright.edu

Abstract. People that interact within a cooperative organization must constantly exchange information on the details of the organization as well as the goals the organization exists to meet. Agent organizations must share knowledge if they are to cooperatively act in the solution of some set of defined goals. The manner in which they share and when they share information varies. In this paper, we present the process to share organization information during the process of transition from one organization state to the next. Some organization models choose to vary the information known between two agents, in relation to the organization. A key element of organization success is that all members operate with the same information so as not to cause divergence in action or purpose.

1 Introduction

Organizations exist in every facet of human existence. People join organizations for reasons such as fulfillment, position or learning. When a person joins an organization, they must learn, or at least be aware, of the others involved in the organization. They must understand the overall structure to fully comprehend their place within the organization. As an example, human organizations commonly use charts to describe where each person fits into the structure. These organization charts exhibit the relationships between positions and people. When a new person joins an organization they are shown where they fit as part of the orientation to the organization. As the organization transitions through changes, the knowledge required for continued understanding of place and position must be updated.

To learn about the organization, the person must exchange organization specific information with others. When they first join, others in the organization transfer information to them to facilitate their organizational learning. The organizational learning is not necessarily classical learning, but instead a process to share or transfer knowledge. Each agent is previously aware of the knowledge structure and process required to interact with other agents in the organization.

Modeling agent organizations using the inspiration of human organizations, as is commonly done, the designer must create the formalities and implementation to allow the transfer of information between agents. We look at interaction as a basic exchange of information between two agents, but can be extended to any number of agents belonging to an organization. The goal of the exchange is to maintain a state of perfect information between all agents. Perfect organization dictates that all agents must have identical organizational knowledge. The trick is that during transition, initial organization or reorganization, the information will change for at least one agent. That agent then has to insure that all other agents must receive the same knowledge changes. Differences in knowledge between agents will cause potential divergence in goals or roles played by the organization. The effect of bad information, in an agent organization, is much the same as if it were a human organization.

Our logic-based approach to this problem stems from some fundamental work, such as work by Su et al. [13]. Deitterich expressed the need to establish a useful level to approach knowledge, both for storing and learning or exchange [4]. Gordon and Subramanian augmented the approach to knowledge, by establishing the need for finer grain tuning of logic [8]. Baader provides a more general approach to the need for knowledge representation [2]. The basis of these works establishes the fit of logical representations for organization knowledge storage. In this work, the logical representation of organization will mirror the structural representation of organization.

In terms of knowledge sharing, Dignum and Dignum [5] indicate the shift from sharing to collaboration. That is key for this effort, although the basis of our model restricts the knowledge exchanged to a very specific set, lending to the strategy shown by Soller and Busetta [14] to develop a shared understanding between agents. While not strictly a default set of rules, as described by Rybinski and Ryzko [12], our logical structures are standardized, to simplify the body of knowledge exchanged.

An assumption, for this research, is agents are cooperatively participating in an organization where common goals are paramount. Individual agent goals and motivations are not above the needs of the organization. The difference is our approach reflect separating the constitution of the organization from a strictly structural concept. Organizations are normally perceived as structural, with components and relationships. Our approach considers an organization as a mental image of a structural entity. This approach allows better scalability and computation of new states.

In this paper, we describe and demonstrate the organization knowledge exchange between agents belonging to the same organization. In section 2, we describe model elements and processes for sharing of knowledge between organization agents. In section 3, the implementation of this system is described. Results of the implementation are described in section 4. Section 5 explains opportunities for further work.

2 Organization Knowledge Sharing

In this section, we describe the basic structure required to model organizational information to facilitate exchange of information. The foundation of exchange is an organization model [9, 3]. The agent structure is shown first followed by the structural, state and transitional elements of our organization model. Once an organization model is described we extend the model to include processes of exchange. Finally, the model and processes are integrated to show the overall formalities of knowledge transfer between agents.

2.1 Agent Core Composition

Before looking at the specific elements of organization, we must first show the overall structure of an agent. An agent is comprised of several knowledge cores, as shown in Fig. 1. An agent has three knowledge cores which are the *organization core*, *communications core* and *task core*. Each core represents the knowledge held by an agent in an area. For example, the communications core represents all knowledge required to communicate with all other agents to which the agent has access. The task core represents it knowledge of each of the capabilities possessed by the agent. An agent may have numerous task cores. While all three cores compose an agents knowledge, the organization core is the one of most interest in this research, and will be the focus of the discussion. This core represents all of the knowledge of the organization in which an agent participates. In simple terms, it is its own internal organization chart defining all components, structural relationships and state relationships of the organization. As the structures contained within the core are discrete, all agents work with an even base in which to share organization knowledge.

2.2 Organization Model Elements

Our organizational model (O) has a structural model, a state model and a transition function [9], described as:

$$O = (O_{structure}, O_{state}, O_{transition}) \tag{1}$$

Before approaching the details of information exchange, we must examine the *structural* and *state* elements of our model. The component and relationship elements are represented as the stored knowledge to be exchanged.

Structure The *structure* is defined by:

$$O_{structure} =< G, R, L, C, ach, rel, req, sub, con > \tag{2}$$

where *ach* is *achieves*, *rel* is *related*, *req* is *requires*, *sub* is *subgoal* and *con* is *conjunctive*, respectively. G describes the set of *goals*, R is the set of *roles*, L is the set of *laws* or *rules* required, and C is the set of *capabilities*. The

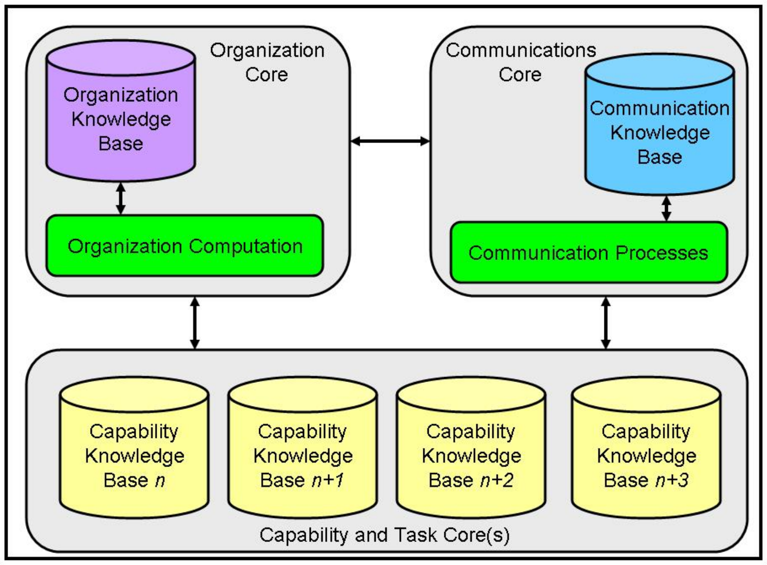

Organization Core
Organization Knowledge Base
Organization Computation
Communications Core
Communication Knowledge Base
Communication Processes
Capability Knowledge Base *n*
Capability Knowledge Base *n+1*
Capability Knowledge Base *n+2*
Capability Knowledge Base *n+3*
Capability and Task Core(s)

Fig. 1. Knowledge Cores of an Agent

organization structure also contains a set of relations. The achieves relation, $achieves : R, G \rightarrow [0..1]$, states the relative ability of a *role* to satisfy a given *goal*. The *related* function $related : R, R \rightarrow Boolean$ exists only if two *roles* are related. *Roles* require *capabilities* to satisfy a set of *goals* and this is captured by the $requires : R, C \rightarrow Boolean$. The organization may contain subgoal relationships $subgoal : G, G \rightarrow Boolean$. The conjunctive relationship between *goals* is $conjunctive : G \rightarrow Boolean$.

State The *state* is defined by:

$$O_{state} =< A, possesses, capable, assigned, coord > \tag{3}$$

where an A defines a set of *agents* available to participate in the organization. There are several relationships in the state element of the organization. An *agent* capable of playing a certain role possesses the necessary *capabilities* described by the possesses relation, $possesses : A, C \rightarrow [0..1]$. An *agent* is capable of playing a *role* in the organization as described by the capable relation, $capable : A, R \rightarrow [0..1]$. The assigned relation, $assigned : A, R, G \rightarrow [0..1]$, is used to match the best *agent*, *role*, *goal* combination that maximizes the capability of the organization. The coordination relation, $coord : A, A \rightarrow Boolean$, allows a relationship between two *agents*.

Transition Transition is the main topic of knowledge exchange as transition requires that knowledge be exchanged by all agents participating in the organization. There are two specific transition processes, *initial organization* and *reorganization*. From organization $state_0$ to $state_1$ is initial organization. All other transitions are reorganization. Transition is expressed by:

$$O_{transition} = (O, \Phi, \delta, s_n, S_{optimal}, S_{possible}, S_{final}) \tag{4}$$

Where O is the organization over which the transition will occur, Φ is the set of properties that can trigger a transition of the organization, δ is the transition function, s_n is the set of relative states of the organization, $S_{optimal}$ is the set of optimal states that result from transition and $S_{possible}$ are states that are possible to reach, from the current state. S_{final} is a set of organization states where all goals are satisfied, or the lst goal is satisfied, or it is determined that not all goals can be satisfied. Even though the outcomes are different, each final state draws a conclusion to the organization's set of transitions. Because an organization can only exist as a single entity or instance, the current state s_n is always a unique value [10].

The basic transition is defined as a product of the O, Φ and S resulting in a set of reachable organization states:

$$\delta : O \times \Phi \times S \Rightarrow S \tag{5}$$

So, the transition function is of the form:

$$\delta(O, \phi, s_n) \Rightarrow S' \tag{6}$$

Where transition function δ takes the organization O, a *specific* transition property ϕ, and a state of the organization s_n and can transition to a set of new states S' where $S_{optimal} \subseteq S_{possible}$, $S_{optimal} \subseteq S'$ and $S_{final} \subseteq S_{possible}$.

Transition Properties Transition properties Φ represent stimuli that can change the organization. They are represented in logical format which capture the generic nature of what they can define. In general terms, an organization will need a set of properties Φ, for example, *capabilities* or *agents*, which can be the stimulus of transition. An individual property $\phi \in \Phi$ is eligible to act as a reorganization trigger. Some examples of ϕ include a change in the real value of a capability, the loss of overall capability or agent function, loss of an agent, the reentry of an agent, or the addition of a new agent.

Each domain problem, represented by knowledge in a task core, may create a number of task specific transition properties. We will first show general, abstract properties and then discuss specific properties. These general properties can be instantiated to fit specific examples. Some general transition properties are:

1. Loss of an agent participating in the organization
2. An agent loses capability required to play some role
3. A new agent becomes available
4. Capability of an agent increases
5. Capability of an agent decreases
6. A goal is removed
7. A goal is added

8. A goal is relaxed (changed)
9. Change in goals to roles achieves relationship
10. Change in role to capability requires relationship

Changes in organization structure and participants will drive transition activities. Transition properties can be triggered internally or externally. The general transition properties can be split into properties that are external and those that are internal.

Transition Predicates A transition predicate is a formalization of a transition property. The formalization of transition predicates enables the exchange of information. Transition predicates can also be expressed as $\Phi = \{\phi_1 \ldots \phi_n\}$. In general, Φ can be expressed as a set of standard, abstract predicates, $\Phi = \{\phi_{lose}, \phi_{add}, \phi_{change}\}$, where ϕ_{lose} is the abstract property dealing with loss, such as losing an agent from the organization or an agent losing capability to play a role. The add property ϕ_{add} describes the action when an agent becomes available for invitation to the organization. The change property ϕ_{change} can either be an increase or decrease and further specializes the change predicate, $\phi_{change} = \{\phi_{decrease}, \phi_{increase}\}$ [11].

The *primitive predicates* exhibit polymorphic behavior as each can be applied to different organization elements to capture different properties.

1. Loss of an agent participating in the organization $\phi_{lose}(a)$
2. An agent loses capability required to play some role $\phi_{lose}(c_i, a)$
3. A new agent becomes available $\phi_{add}(b)$
4. Capability of an agent increases $\phi_{increase}(c_i, a)$
5. Capability of an agent decreases $\phi_{decrease}(c_i, a)$
6. A goal is removed $\phi_{lose}(g)$
7. A goal is added $\phi_{add}(g)$
8. A goal is relaxed (changed) $\phi_{change}(g)$
9. Change in goals to roles achieves $\phi_{change} achieves(r_i, g_j)$
10. Change in role to capabilities $\phi_{change} requires(r_i, c_j)$

Primitive predicates can be used to formalize single properties. If there is a loss of an agent participating in the organization, it can be formalized as the predicate $\phi_{lose}(a)$. An agent a losing some capability can be captured as $\phi_{lose}(c_i, a)$. *Complex predicates* represent the combination of simple predicates logically constructed using common *and* ($\wedge$), *or* ($\vee$) and *not* ($\neg$) relations.

Some predicates will encompass others, but in some cases two properties can be successfully combined to form a single property of transition. In the case that an agent exits an organization, it can be reasoned that all capability of that agent will also exit. So combining the two previous predicates of losing an agent and losing a capability by an agent are redundant, in respect to the capability predicate $\phi_{lose}(a) \wedge \phi_{lose}(c_i, a)$. In another situation, an organization may lose two agents simultaneously. If agents a and b both leave, we can capture that by $\phi_{lose}(a) \wedge \phi_{lose}(b)$, where one primitive predicate does not contain the other.

As there are primitive and complex predicates, another perspective shows *component* and *relationship* predicates. A component predicate is defined as a predicate where the property relates to a component of the organization, such as an agent being added or a goal being deleted. A relationship predicate is defined by a property where a relationship between two components is added, deleted or altered. Relationship predicates can be primitive. Component predicates must be complex as the component must collaborate with a relationship to connect to the organization.

2.3 Exchange Processes

A model is not necessarily sufficient to completely explain the exchange of knowledge. The process must also show how the agents interact to share the information. This definition only describes the basic mechanics of the exchange. It must be further explored to answer questions on what basis is information exchanged. Will the information be shared with anyone who asks? Will the information be shared with all agents? Will it be shared with agents who do not specifically ask for it? These questions not only pose a set of philosophical queries, but also pose some practical problems in exchange. Automatically sending data to an agent that does not need it, as it already possesses the information, is wasteful in terms of resources.

Our approach to knowledge exchange is similar to the *mind-body problem* of Descartes. In the mind-body problem, the mind is differentiated from the matter of the body. The knowledge of an organization, which resides in the individual mind of each agent, within the organization, is different than the physical manifestation of the organization. Each agent carries an image of the organization with all components and relationships. The key is for all agents to have the same image of the organization, in other words, perfect information.

The basic premise is that when each transition occurs, all agents need to be updated with the current organization knowledge. When a human organization requires change, a decision is made and the change is then communicated by the decision maker to those affected. As with human organizations, a single agent will receive the change, ϕ and propagate the change to each of the other agents in the organization.

There exists a risk of a transition property not correctly propagating from the sending agent to the receiving agent. If this occurs, the receiving agent will not compute a new organization and will be different than those agents who successfully received the message. If for some reason, such as an agent being deleted, another agent will sense the agent loss and update the others. It can also be said that each of the others can self update in the event of a loss, but questions whether each is required to recompute. A key goal is to minimize the amount of information transferred for each organization transition.

2.4 Integration

Each agent optimally has the same organization knowledge. This supports the premise that all agents operate on full information. Fig. 2 shows an organization of 4 agents $\{agent_1, agent_2, agent_3, agent_4\}$. $Agent_1$ receives a transition property from either an internal or external force. $Agent_1$ then propagates the predicate to $agent_2$, $agent_3$ and $agent_4$. The organization core represents the part of the *mind* of the agent concerned with where it fits in the organization. The agents themselves represent the physical manifestation, or the *body*.

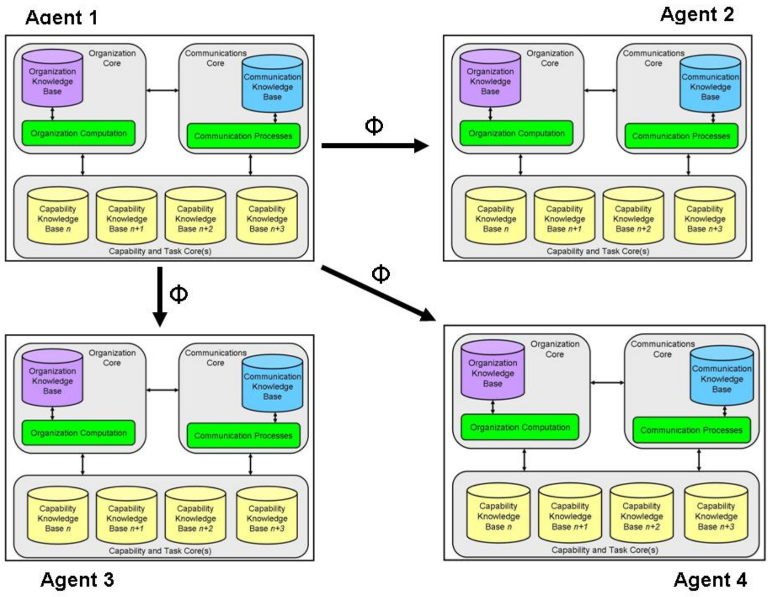

Agent 1
Agent 2
Agent 3
Agent 4
Organization Core
Organization Knowledge Base
Organization Computation
Communications Core
Communication Knowledge Base
Communication Processes
Capability Knowledge Base n
Capability Knowledge Base n+1
Capability Knowledge Base n+2
Capability Knowledge Base n+3
Capability and Task Core(s)
Φ

Fig. 2. Knowledge Transfer

3 Implementation

The organization formalisms and knowledge exchange processes have been implemented to complete this work. The implementation is a combination of Java used as the main development platform with *JESS* utilized to implement the knowledge bases. *JESS* has a natural relationship with Java as described by Friedman-Hill [6] and utilizes the rete algorithm of Forgy et al. [7] and Albert [1] shows the computational fit for this algorithm applied to this technical problem.

In this section, the implementation of the structural and state elements as logical constructs in *JESS* are discussed. Each component and relationship are expressed as logical predicates. This logical expression represents the *mind* of the organization. Each predicate is sent to each agent in the *body* and then each agent recomputes a new organization image within their own structure. Thus the *mind* of each agent in the *body* recomputes its own like image of the organization after each change. All *JESS* logical functions are constructed with rules and facts, based on templates. The organization object is then embedded inside a Java shell for integration with the body of the organization, written in Java.

3.1 Structure and State

Each predicate of the organization model's structure and state can be directly represented by a template in *JESS*. For example, the structural templates are:

```
(deftemplate goal (slot goal))
(deftemplate goal (slot role))
(deftemplate goal (slot capability) (slot score))
(deftemplate achieves (slot role) (slot goal) (slot score))
(deftemplate related (slot role) (slot role))
(deftemplate requires (slot role) (slot capability))
(deftemplate subgoal (slot goal) (slot goal))
(deftemplate conjunctive (slot goal))
```

The state templates are:

```
(deftemplate agent (slot agent))
(deftemplate possesses (slot agent) (slot capability) (slot score))
(deftemplate capable (slot agent) (slot role) (slot goal) (slot score))
(deftemplate coord (slot agent) (slot agent))
```

So a ϕ property of adding a goal, $\phi_{add}(g)$, will exist in JESS as $(goal(goalg))$ added by a rule in *JESS*.

3.2 Transition and Exchange

Each agent is a complete independent entity communicating via TCP/IP sockets. All knowledge is exchanged using Java via networking between distributed agents. This technology is employed specifically for loss reduction and error handling abilities in relation to knowledge exchange.

There are three specific change categories which can effect the exchange process. The first is change to a structural element of the organization. Examples of structural change are to add or lose a goal. The second is the change in a state element. An example of state change is an agent gaining or losing capability, thereby requiring a computation of the organization. The third option is the loss or gain of an agent. Each of these changes will be described using the transition predicates and exchange of *JESS* constructs.

Structural Change If a goal is added or lost, the agent first notified must send a message to all others to retain the state of perfect information. If a transition predicate $\phi_{add}(g_n)$ is received by agent x, a_x, then a message must be propagated to all other agents to add the new goal, as a fact. For each organization knowledge core a new fact is added.

State Change If there is a state change such as the capability of agent x increases $\phi_{increase}(c_i, a_x)$, then that agent will propagate the new fact to all other agents.

Agent Change When there is no change to the collection of agents, within the organization, it is straightforward to propagate the new information to all agents. When an agent is gained or lost, the matter of communication takes on a new level of complexity. When an agent is lost, one or more of the agents remaining must recognize the loss. One of the agents must define a predicate $\phi_{lose}(a_x)$, create the update and send to all agents. If an agent is gained to the organization, $\phi_{gain}(a_x)$, the new fact that an agent has been added is sent to all agents by one of the agents, already in the organization.

4 Results

We must first distinguish between results split by the two transition processes, initial organization and reorganization. The result indicates the initial organization is computationally more intense and is based on the number of components and relationships. Since it will only be computed once in each organization's life, its effect is discounted. Reorganization is much smaller, due to the incremental nature of only having to recompute around new components and relationships of the ϕ predicate. If ϕ is quite large, it may alter the computational intensity. For example, if the number of components and relationships in ϕ is equal or greater than the existing organization, reorganization may be computationally large.

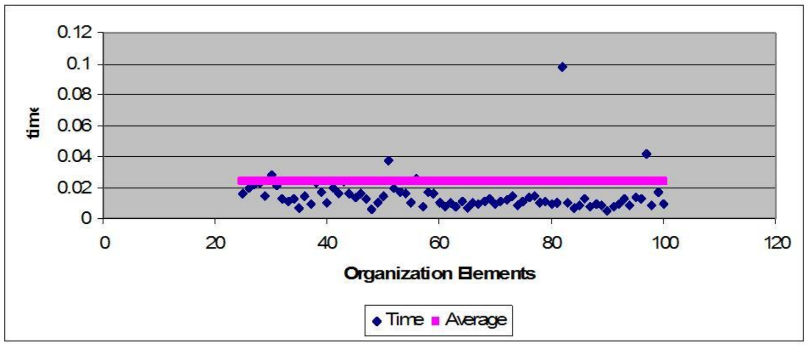


Fig. 3. Results

The computation of a transitioning organization differs from an initial organization to a reorganization. In a strictly structural context, initial organization and reorganization do not differ a great deal. In our mind body approach the difference is significant. For a small organization size of 10 goals, 10 roles, 10 agents and complete relationship set, the time for initial organization is 0.03219 seconds. The time for a reorganization based on one new component and all relationships is 0.01754 seconds. Fig. 3 shows the time to compute a transition against the size of the organization, in elements. The initial organization used in this analysis has 10 organization components, such as roles, agents or goals, and 15 relationships between those components. The total number of components, on the lower end, is 25. The data shows the time to compute the transition going from 25 components to 100, which is beyond a trivial organization. The transition process is based on computing an optimal organization configuration. The key is that the time to recompute is not significantly different for the larger organization. This is due to the incremental nature of the computation process. This indicates use of this method, is at least initially, scalable.

If we compare this timing to another result by Zhong it shows the difference. In Zhong's research[15], based on a similar model, using only the constructive version of the structural model algorithm to transition, the results of a structural computation yields two interesting points. First, the structural model transition algorithms grows at a fast rate as the number of organization components grow. Secondly, the ability to scale to large organizations will be significantly hindered by a strictly structural approach. This indicates as the size of the organization grows, the difference between our approach and a more structural-based approach will grow, in terms of time to compute.

Instantiating an organization and its transition processes in terms of a mind-body approach has advantages over a strictly structural computational approach. While there are also a few disadvantages, these are overcome by the positives.

Computation minimization is the best result of this approach. While larger, more complex organizations must be tested, the early results show promise. The computation is performed locally and in parallel, which allows the transition process to be completed more rapidly. The intent is for each agent to work with perfect information and each agent will have the same organization image, without transferring the entire structure each transition cycle. The rate of message growth is small. Even with a large change, all computation is local. This will allow a near linear growth rate during organization augmentation. This will reduce temporal computation problems in transition processes.

There are a few negative side effects of this approach. Perfect information requires that information is transferred from agent to agent without interruption or error. If there is a transfer loss, the synchronization of the organization image maps will suffer. Recovering from loss, during exchange, is key for the design. There must be synchronization allowing each agent to recompute simultaneously with all others. If there are lag times, it can create temporal problems in transition.

5 Further Work

The initial algorithm will be extended to a complete distributed model and a hybrid model, which allows an integration of command mode and complete distributed behavior. Larger organizations will be theoretically analyzed and empirically analyzed to determine performance over large, distributed agent organizations and societies. The scalability question will be further developed to see if there is a breaking point of the design.

References

1. Luc Albert, Average Case Complexity Analysis of Rete Pattern-match algorithm and Average Size of Join in Databases. Rapports de Reserche, No. 1010. Institut National de Reserche en Informatique and Automatique, Rocquencourt, France, April, 1989.
2. Franz Baader, Logic-based Knowledge Representation. Artificial Intelligence Today, Recent Trends and Developments, no. 1600. Springer-Verlag, M. Wooldridge and M. Velosa (eds.), 13-41, 1999.
3. Scott DeLoach, Eric Matson. An Organizational Model for Designing Adaptive Multiagent Systems. Agent Organizations: Theory and Practice at the National Conference on Artificial Intelligence (AAAI-04), July 25-29, 2004, San Jose, CA.
4. Thomas Dietterich, Learning at the Knowledge Level. Machine Learning, 1:287-316, 1986, Kluwer Academic Publishers, Boston, MA, USA.
5. Virginia Dignum, Frank Dignum.The Knowledge Market: Agent-Mediated Knowledge Sharing, Lecture Notes in Computer Science Springer Berlin/Heidelberg, vol. 2691, Multi-Agent Systems and Applications III: 3rd International Central and Eastern European Conference on Multi-Agent Systems, CEEMAS 2003, Prague, Czech Republic, June 16-18, 2003.
6. Ernest Friedman-Hill. JESS in Action: Rule Based Systems in Java. Manning Publications, Inc., Grennwich Connecticut, USA, 2003.
7. Charles Forgy, Allen Newell, Anoop Gupta. High-Speed Implementation of Rule-Based Systems. ACM Transactions on Computer Systems, Vol. 7, no. 2, May 1989, pages 119-146.
8. Diana Gordon, Devika Subramanian, A MultiStrategy Learning Scheme for Agent Knowledge Acquisition, Informatica, 17:4, 1993.
9. Eric Matson, Scott DeLoach. Organizational Model for Cooperative and Sustaining Robotic Ecologies, Proceedings of Robosphere 2002 Workshop, NASA Ames Research Center, Moffett Field, California, November 14-15, 2002.
10. Eric Matson, Scott DeLoach, Formal Transition in Agent Organizations, IEEE International Conference on Knowledge Intensive Multiagent Systems (KIMAS '05), Waltham, MA, April 18-21, 2005.
11. Eric Matson, Raj Bhatnagar. Properties of Capability Based Agent Organization Transition. 2006 IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT-2006), Hong Kong, December 18-22, 2006.
12. H. Rybinski and D. Ryzko. Knowledge Sharing in Default Reasoning-Based Multiagent Systems, IEEE/WIC/ACM International Conference on Intelligent Agent Technology, 2003. October 13-16, 2003, pp. 576 - 579.
13. Kaile Su, Xudong Luo, Huaiqing Wang, Chengqi Zhang, Shichao Zhang, Qingfeng Chen. A Logical Framework for Knowledge Sharing in Multi-agent Systems, Lecture Notes in Computer Science, vol. 2108, Springer Berlin/Heidelberg. Computing and Combinatorics : 7th Annual International Conference, COCOON 2001, Guilin, China, August 20-23, 2001,
14. Amy Soller and Paolo Busetta. An Intelligent Agent Architecture for Facilitating Knowledge Sharing Communication, in Proceedings of Workshop on Humans and Multi-Agent Systems, International Conference on Autonomous Agents and Multi-Agent Systems, Melbourne, 2003, pp. 94-100.
15. Christopher Zhong, Scott A. DeLoach. An Investigation of Reorganization Algorithms. Proceedings of the International Conference on Artificial Intelligence (IC-AI'2006). June 2006, Las Vegas, Nevada, CSREA Press, 2006.

Distributed Norm Enforcement via Ostracism

Adrian Perreau de Pinninck, Carles Sierra, and Marco Schorlemmer

IIIA – Artificial Intelligence Research Institute
CSIC – Spanish National Research Council
Bellaterra (Barcelona), Catalonia, Spain
adrianp,sierra,marco@iiia.csic.es

Abstract. An agent normative society has to deal with two main concerns: how to define norms and how to enforce them. Enforcement becomes a complex issue as agent societies become more decentralized and open. We propose a new distributed mechanism to enforce norms by ostracizing agents that do not abide by them. Our simulations have shown that, although complete ostracism is not always possible, the mechanism substantially reduces the number of norm violations.

1 Introduction

In a normative Multi-Agent System (MAS) a set of norms are added to restrict the set of available actions in order to improve the coordination between agents. An autonomous agent has the choice whether or not to support a norm. It is up to the agent to decide if it is convenient for it to abide by it. For a utility maximizer agent if following a norm is profitable, it is in the agent's own interest to act as the norm establishes. But this is not always the case, as some norms are profitable even when not all agents abide by them. For example, a norm that dictates that owners must clean the common areas. Cleaning entails a cost, and a clean area is a benefit to all. If an owner does not clean the common area (*i.e.*, a norm violator) thus not bearing its cost, but the others do, the area is still clean.

The aim of this paper is to introduce a new distributed mechanism that attains norm compliance by ostracizing norm violating agents. Our scenario allows agents to interact with each other. An agent can interact with the agents it is linked to directly or indirectly through a path of links (*i.e.*, agents can interact with direct neighbors, with neighbors of neighbors, and with their neighbors and so on...). An initiator agent will search for a path in the society to find a partner agent with which to interact. All the agents in the path that are not the initiator or the partner agent will be called mediator agents (*i.e.*, agents mediating the interaction).

We use a game-theoretic approach to interactions, which we model as a two-player game with two possible strategies; cooperate and defect. The utility function will be that of a prisoner's dilemma (see Figure 1).

The norm in this scenario is for all agents to cooperate, thus attaining the maximum utility for the society. Nonetheless, agents can choose to ignore the

PD	Cooperate	Defect
Cooperate	3,3	0,5
Defect	5,0	1,1

Fig. 1. Prisoner's Dilemma Payoff Matrix

norm and defect (*i.e.*, violate the norm) thus gaining more utility. In order to attain norm enforcement, some agents (we will call them enforcer agents) are given the ability to stop interacting with violators, and to stop them from interacting with the enforcer's own neighbors. When enough agents use this ability against a violator, it is ostracized. An agent is ostracized when it cannot interact with anyone else in the society, in this case it is a consequence of defecting in the interaction against many different agents.

The motivation behind using ostracism comes from the study of norm enforcement in primitive societies [11]. When a member of a community repeatedly ignored its customs, it was forced to leave. No one in the community would interact with the ostracized member from then on. Ostracism is achieved in human societies through force and physical constraint. In order to achieve ostracism of electronic entities, which interact through a network, we seek inspiration from the network security area. The most commonly used component in this case is a firewall, which blocks those communications which appear to be harmful. While firewalls are usually set up by humans through complex rules, enforcer agents will use gossip as a way to inform each other about malicious agents.

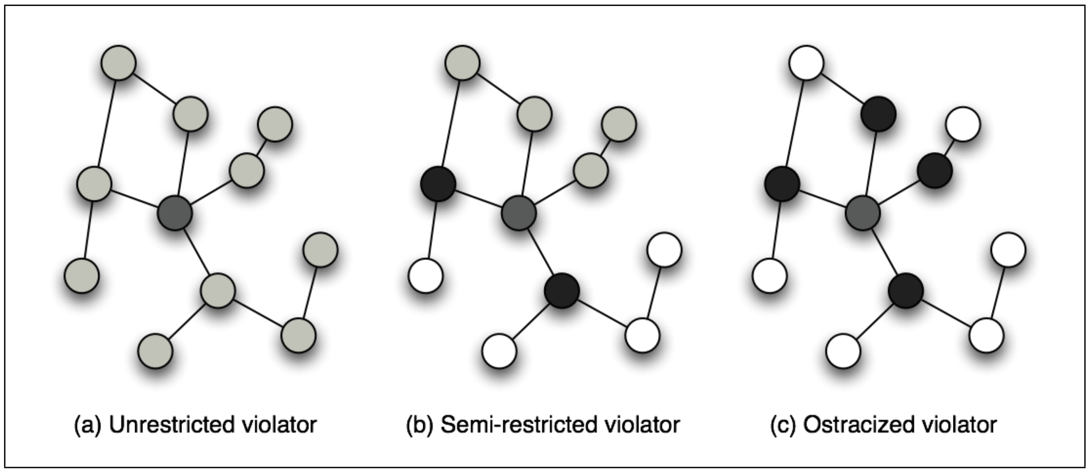


Fig. 2. Ostracizing a violator

The ostracism process can be seen in Figure 2. At first an undetected violator in the network (the dark gray node) can interact with all the other agents (light gray nodes are liable to interact with the violator). When the violator interacts, and defects, it can be detected by enforcer agents which will block it (black nodes are blocking agents, and white nodes are agents that the violator cannot interact with). When all the violator's neighbors block it, it is ostracized.

Gossip is essential to find out information about other agents in a distributed environment. We will use gossip as part of the enforcement strategy to ostracize agents. Information is gossiped only to agents mediating the interaction, to minimize the amount or resources it takes. If agent ag_v violates the norm when interacting with agent ag_1, ag_1 may spread this information to all mediator agents so they may block ag_v in the future.

By running a set of simulations, we study under which conditions the mechanism works, and give measures of its success (such as the violations received or the utility gained). Our hypotheses are:

- **H1** - Norm violations can be reduced by applying a local blocking rule.
- **H2** - The society's structure influences its enforcement capabilities.
- **H3** - The choice of blocking strategy influences the number of violations received.
- **H4** - Enforcement makes norm-abiding a rational strategy.

Section 2 describes related work in the area of norm enforcement. Section 3 presents a detailed description of the scenario we employ in the simulations. Section 4 describes the simulations and analyzes the resulting data. Finally, section 5 presents future work.

2 Related Work

The problem of norm enforcement has been dealt with in human societies through the study of law, philosophy, and the social sciences. Recently it is being dealt with in computer science, where norms are studied as a coordination mechanism for multi-agent systems. Axelrod [1] first dealt with the application of norms from an evolutionary perspective. Enforcement is seen by Axelrod as a sort of meta-norm to punish agents that do not punish violators. The norm game is often modeled as an N-Player Iterated Prisoner's Dilemma [1, 8]. In these cases the norm is to cooperate and ways are sought to ensure that agents prefer cooperation. Other research studies norms that avoid aggression or theft [4, 7, 12, 15]. In these cases agents gain utility by either finding items or receiving them as gifts. But these items can be stolen by other agents through aggression. An agent that abides by the possession norms will not steal food possessed by another agent, therefore avoiding aggression.

Two enforcement strategies have been studied to attain norm compliance: the use of power to change the utilities through sanctions or rewards [2, 3, 8, 14], and the spread of normative reputation in order to avoid interaction with violators [4, 6, 7, 12, 15]. Both strategies have the goal of making norm adopters better off than norm violators. But this is not always accomplished [4, 7], since all agents benefit from the norm while only enforcers agents bear its cost.

Norm enforcement models in [2, 6] show how violating the norm becomes an irrational strategy when punishment is possible. But these models assume the following: (i) agents are able to monitor other agents' activities; and (ii) agents have the ability to influence the resulting utility of interactions. Assumption (i) can be materialized by having a central agent mediate all interactions [2], or by having agents recognize violators through direct interaction with them, or through gossip with other agents [4]. The first solution does not scale, since the mediator agent would be overwhelmed in a large system. The second scales because no agent is the enforcement bottleneck, but it is less efficient since in a distributed environment not all violations can be detected. Assumption (ii) can be carried out through third-party enforcement [2], or self-enforcement [6] in which each agent carries out sanctions to agents it interacts with. Third party does not scale since it can easily be overwhelmed in a large system. For self-enforcement, all agents must have the ability to affect the outcome utility of interactions.

Axelrod [1] defines the "shadow of the future" as a mechanism to affect an agent's choice in iterated games. An agent is deterred from defecting when the probability of interacting with the same agent in the future is high, and agents will defect in future interactions with known violators. Nonetheless, this mechanism makes enforcers violate the norm as they also defect. Another method is the threat of ostracism or physical constraint. By not interacting with violators, an agent can interact with another agent and achieve a higher payoff. Younger has studied [15] the possibility of avoiding interaction with norm-violators, but does not prevent norm-violators from interacting with anyone else.

Kittock [9] was the first to study how the structure of a multi agent system affected the emergence of a social norm. He studied regular graphs, hierarchies, and trees. In [5] Delgado studied emergence in complex graphs such as scale-free and small-world, and in [10] studied the relationship between a graph's clustering factor and emergence.

Using the scenario presented in this paper, agents can monitor other agents' activities, and influence future interactions. The spread gossip, and sanctioning norm-violators with ostracism via blockage are the techniques used to achieve this influence. We have studied norm enforcement using these techniques in societies with differing structures.

3 The Scenario

We model our multi-agent system as an undirected, irreflexive graph: $MAS = \langle Ag, Rel \rangle$, where Ag is the set of vertices and Rel the set of edges. Each vertex models an agent and each edge between two vertices denotes that the agents are linked to each other. We have chosen three kinds of graphs for their significance: Tree, Random, and Small-World. We define a tree as a graph in which each node has one parent and some number of children; one node, the root node, has no parent, and the leave nodes have no children. Nodes are linked to their parents and children. In a random graph any node can be linked to any other one with a given probability. A small-world graph is created by starting with a regular graph[1], and adding enough random edges to make the average distance between any two vertices significantly smaller [13]. A small-world graph is highly clustered (*i.e.*, if a node has two neighbors, the probability of them being linked is high), and there are some links between distant parts of the graph that make the average distance between any two edges small. The graph structures have been generated with a similar average number of links per node.

We use a game-theoretic approach by modeling interactions as a two-player prisoner's dilemma game. The norm is that agents ought to cooperate (*i.e.*, an agent disobeys the norm by defecting). In order for two agents to interact, there must be a path in the graph between the two. One agent will search for a path that leads to another agent with which to interact. We call the searching agent *initiator agent*, the agent chosen to interact *partner agent*, and the remaining agents in the path *mediator agents*. The partner finding process is explained below, but first we need to formally describe some terms.

We define the set of neighbors of an agent a_i as the set of agents it is linked to directly in the graph: $N(a_i) = \{a_j \in Ag \mid (a_i, a_j) \in Rel\}$. Each agent also has a set of agents it blocks (an agent cannot block itself): $B(a_i) \subseteq Ag \setminus \{a_i\}$. An agent a_i can query another agent a_j for a list of its neighbors. We call the set of agents that a_j returns, reported neighbors: $RN(a_i, a_j) \subseteq N(a_j)$. The set of reported neighbors depends on the blocking strategy of a_j. The strategies used in our simulations are explained below. A path is the route (without cycles) in the graph structure through which interaction messages are delivered. We represent a path as a finite (ordered) sequence of agents $p = [a_1, a_2, \ldots, a_n]$ such that for all i with $1 \leq i \leq n-1$ and $n \geq 2$ we have that $a_{i+1} \in N(a_i)$, and for all i, j with $1 \leq i, j \leq n$ and $i \neq j$ we have that $a_i \neq a_j$. The agent a_1 of a path is the initiator agent, agent a_n is the partner agent, the remaining ones are mediator agents.

In order to find a partner, the initiator agent a_i creates a path $p = [a_i]$ with itself as the only agent in it. A path with one agent is not valid, since an agent cannot interact with itself. Therefore, the initiator agent will query the last agent in the path (the first time it will be itself) to give it a list of its neighbors. It will choose one of them randomly[2] (a_j) and add it to the end of the path $p = [a_i, ..., a_j]$. At this point, if agent a_j allows it, the initiator agent can choose agent a_j as the partner. Otherwise, it can query agent a_j for its neighbors and continue searching for a partner. This choice is taken randomly, with probability $p = 0.3$ a_j becomes the partner, and with probability $1-p$ it becomes a mediator and a_i asks it for its neighbors.

If the path's last element is an agent a_n that refuses to interact with the initiator agent, and a_n returns an empty list of agents when queried for its neighbors, backtracking is applied. Agent a_n is removed and a different agent is chosen from the list of a_{n-1}'s neighbors and added to the end of the list.

[1] $C_{N,r}$ is a regular graph on N vertices such that vertex i is adjacent to vertices $(i+j)$ mod N and $(i-j)$ mod N for $1 \leq j \leq r$.

[2] To avoid loops, an agent that is already part of the path cannot be chosen again.

Once the partner is chosen, a prisoner's dilemma game is played between the initiator and the partner. The game results and the path are known by both playing agents. Playing agents can choose to send the game results to all the mediators in the path. This is what we call *gossip*, which formally speaking is a tuple that contains the agents' names and their strategy choices for the given game: $Gossip = \langle ag_i, ch_i, ag_j, ch_j \rangle$, where ch_i and ch_j are either *cooperate* or *defect*.

During the whole process agents can execute any of the following actions:

– Return a list of neighboring agents when asked for its neighbors.
– Accept, or reject, an offer to interact.
– Choose a strategy to play in the PD game when interacting.
– Inform mediators of the outcome of the interaction.

The society of agents is composed of three types of agents, each one characterized by a different strategy for the actions it can execute. A *meek agent* is a norm-abiding agent that always cooperates. It will always return all its neighbors to any agent that asks. A meek agent will always accept an offer to interact, it will always cooperate in the PD game, and it will never gossip. A *violator agent* follows the strategy of a meek agent, except that it always defects when playing a game, therefore it is not a norm-abiding agent. Violator agents in our simulations are very naive, they never model the other agents, or treat them differently depending on their actions. In short, they cannot change the strategies. Future work will look into more sofisticated norm-violators.

Finally, an *enforcer agent* has the ability to block violators, which is essential in order to achieve their ostracism. An enforcer agent shares the same strategies with meek agents with the following exceptions: It will add agents that have defected against it to its set of blocked agents, and will gossip to all mediators when defections happen. If an enforcer is informed of the results of a game it was mediating, it will act as if it had played the game itself. An enforcer agent will never choose an agent in its blocked set as a partner, and will not allow an agent in its blocked set to choose it as a partner. Therefore, a violator agent cannot interact with an enforcer who is blocking it. When an enforcer agent a_m is asked to return a list of its neighbors by an agent a_i who is not in its blocked set, two different strategies are possible. The Uni-Directional Blockage (UDB) strategy, where all its neighbors will be returned ($RN(a_i, a_m) = N(a_m)$). Or the Bi-Directional Blockage (BDB) strategy, where only those neighbors not in its blocked set are returned ($RN(a_i, a_m) = N(a_m) \setminus B(a_m)$). When the querying agent is in the enforcer agent's blocked set, it always returns an empty set.

The choice of enforcement strategy entails a trade off. Intuitively, one can see that enforcer agents are better off with the UDB strategy, since they will be able to use violator agents as mediators to reach other parts of the society. Enforcers will not be tricked by a violator more than once, so they are sure not to interact with them. Therefore, using violators as mediators benefits enforcers. Meek agents, on the other hand, do not learn to avoid violators. They may choose one unknowingly as their partner repeatedly. BDB is a better strategy for meek agents, it reduces their chances of choosing violator agents. Furthermore, a structure with a violator as a cut vertex, may be split into two different societies when the BDB strategy is used, and the violator is ostracized. If the UDB strategy is used, the society stays connected, since the ostracized violator can stil be used as a mediator.

In order to focus on the most relevant aspects in our simulations, we made the following limiting assumptions:

– Agents cannot change their strategy (*i.e.*, a violator is always a violator).
– Agents cannot lie when sending gossip.
– There are no corrupt enforcer agents.
– There is no noise (*i.e.*, an agent knows its opponent's chosen strategy).

These assumptions imply that modeling agents' reputation is simple. Being informed once about an agent's strategy is enough, since information will never be contradictory. Therefore, there is no place for forgiveness, and sanctions are indefinite. Relaxation of these assumptions will be studied in future work.

4 Simulations

The simulations have been run using the scenario specified in Section 3. Each simulation consists of a society of 100 agents. The society will go through 1000 rounds, in a round each agent tries to find a partner with which to interact. If the agent finds a partner a prisoner's dilemma with the utility function of Figure 1 is played.

The parameters that can be set in each simulation are:

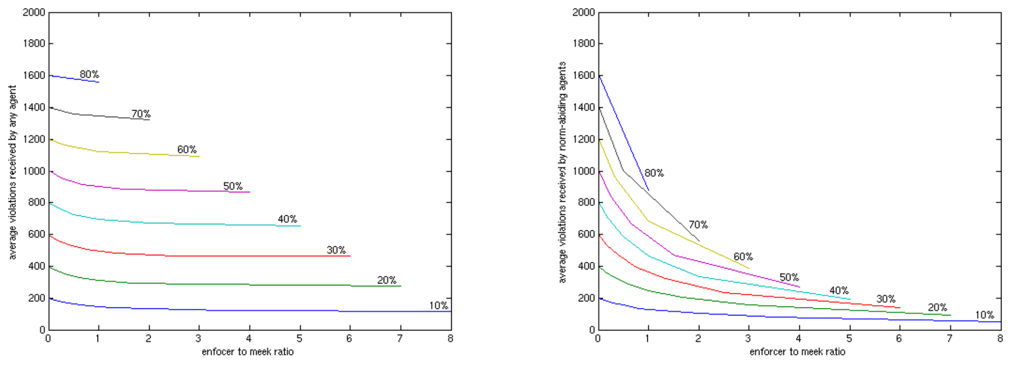
– Percentage of Violators (V) - from 10% to 90% in 10% increments.
– Percentage of Enforcers (E) - from 0% to 100% in 10% increments[3].
– Type of Graph (G) - either tree, small world, or random.
– Enforcement Type (ET) - Uni-Directional Blockage (UDB), or Bi-Directional Blockage (BDB).

An exhaustive set of simulations have been run with all the possible values for each parameter. Each simulation has been run 50 times in order to obtain an accurate average value. The metrics that have been extracted are: the mean violations received per agent, and the mean utility gained per agent. The metrics have been calculated for the whole society and for each agent type. The data gathered from the simulations supports our hypotheses.

(H1) Norm violations can be reduced by applying a local blocking rule. The graph in Figure 3(a) shows that the higher the percentage of norm-abiding agents that use a blocking rule, the lower the average number of norm violations received by any agent in our system. There are five different lines in the graph, each one stands for a different percentage of violating agents. In all cases a higher enforcer to meek agent ratio (x-axes) leads to lower violations

[3] The percentage of meek agents is computed through the following formula: $M = 100\% - V - E$. Therefore, $V + E$ cannot be more than 100%.

received in average by any agent (y-axes). When the ratio of enforcers is high, violators end up interacting with each other. Therefore, the y-axes measures the violations received by "any" agent, the reduction in violations in Figure 3(a) is not significant. The data referring to the violations received only by norm-abiding agents shows a larger reduction (see Figure 3(b)). Enforcer agents can perceive a norm violation at most once per violator agent. But if we look at the violations received by meek agents, we see that they experience an increment of violations when the ratio of enforcers is high (see Figure 7(a)). This means that enforcer agents have blocked violator agents, which are forced to interact with the small number of meek agents left unprotected. Since the meek are a small portion of the norm supporters, this does not influence the total violations perceived by norm supporters as a whole. Therefore, the higher the ratio of enforcer agents, the lower the average of violations perceived by norm-abiding agents.



(a) all agents (b) norm-abiding agents

Fig. 3. Local blocking rule reduces violations

(H2) The society's structure influences its enforcement capabilities. It is also seen from the data that different organizational structures in the multi-agent system influence norm enforcement. In Figure 4(a) and 4(b) we have extracted the average norm violations (y-axes) for each of the different structures tested: Random, Small World, and Tree. We have only shown the simulations where violator agents account for 10% and 20% of the population, therefore at most there will be 90% or 80% of enforcers, respectively. The x-axes contains the different percentages of enforcer agents tested. It can be seen that both random and small world networks have an almost identical graph line. On the other hand the tree structure has shown to improve the enforcement capabilities. The main difference between a tree and the other structures studied is that in a tree there is only one path between any two agents. In random and small world graphs, many paths can be usually found between any two agents.

(H3) The choice of blocking strategy influences the number of violations received. The data in Figure 5 supports this hypothesis. The x-axes shows

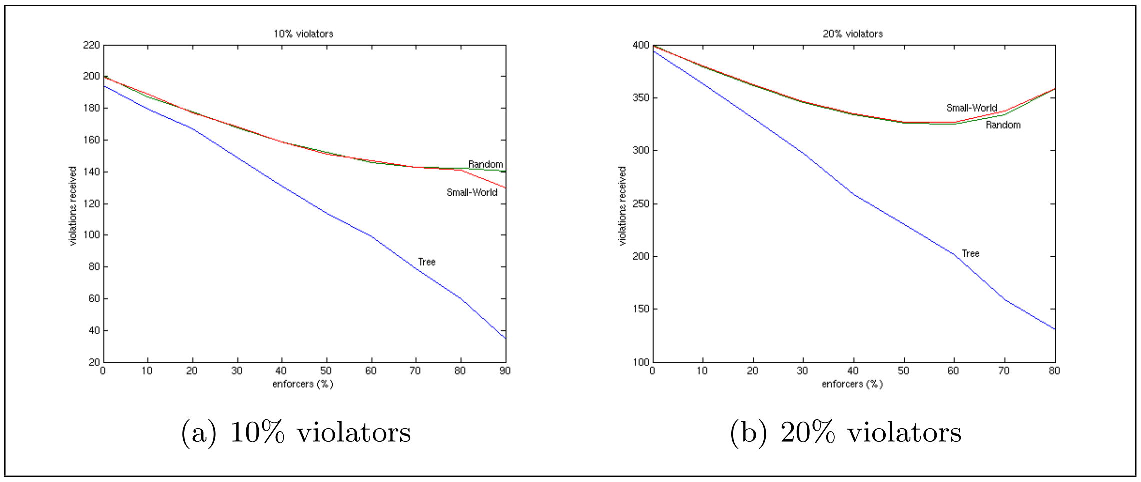


Fig. 4. Enforcement capabilities vary depending on structure

the enforcer to meek agent ratio. The y-axes contains a metric for the increment in efficiency at protecting meek agents from violations. Efficiency is the difference (calculated in percentage) in violations received by meek agents for each of the two different enforcement strategies $\Delta E = ((V_{UDB}/V_{BDB}) - 1) \times 100$. ΔE calculates the increase in violations received by agents when using uni-directional blockage in respect to bi-directional blockage.

Figure 5 shows that for random and small world networks the efficiency is positively correlated with the enforcer to meek agent ratio. We can conclude that Bi-Directional Blockage has a higher efficiency at protecting meek agents from violator agents. This is not observed in the tree network. In this case the efficiency stays along the 0% line with some deviations. We argue that in networks organized as trees, the choice of enforcement strategy does not have a significant influence in the outcome. The tree network is already good for ostracizing offenders, and the blockage strategy does not improve it.

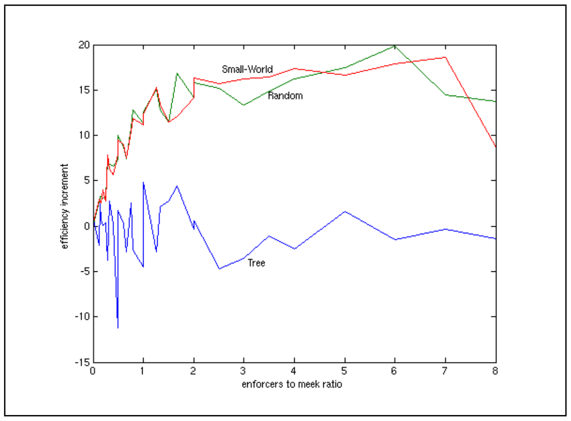
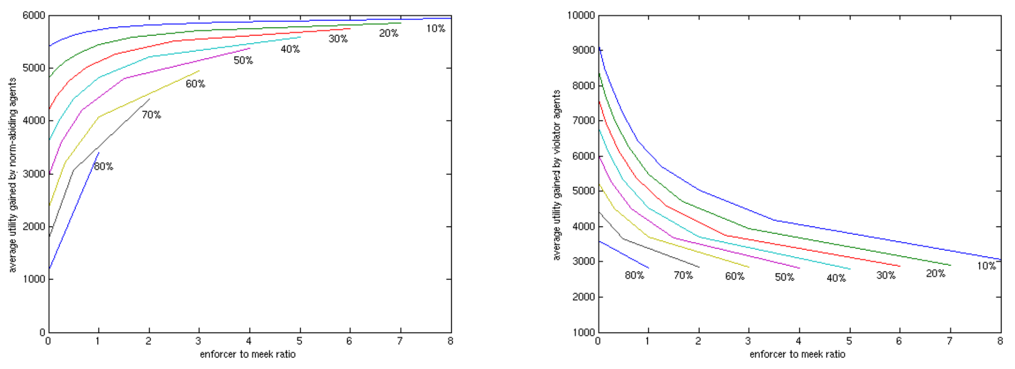


Fig. 5. Enforcement strategy influences received violations

(H4) Enforcement makes norm-abiding a rational strategy. This hypothesis is supported by the utility gained by agents. A strategy is rational if it maximizes the agent's utility. What has been tested is whether following the norm maximizes the agent's utility, and in which conditions. Figure 6(a) shows the utility gained (y-axes) by norm supporting agents, its x-axes shows the enforcer to meek agent ratio. Figure 6(b) instead shows the utility gained by norm violating agents. In both figures each line represents the amount of violating agents in the system. As the number of enforcers increases there is a tendency for norm supporters to gain more utility, while the opposite tendency is observed for violator agents. When the number of enforcer agents is low, the utility gained by violator agents is much higher than the one gained by norm supporters. As the number of enforcer agents grows the roles are reversed. The inflection point depends on the amount of violator agents in the system. For simulations with 10% of violator agents, supporting the norm becomes rational when the enforcer to meek ratio is higher than 1.25. For simulations with 50% of violator agents, the ratio needs to be higher than 0.7. The rest of simulations have inflection points between those two values.



(a) norm-abiding agents

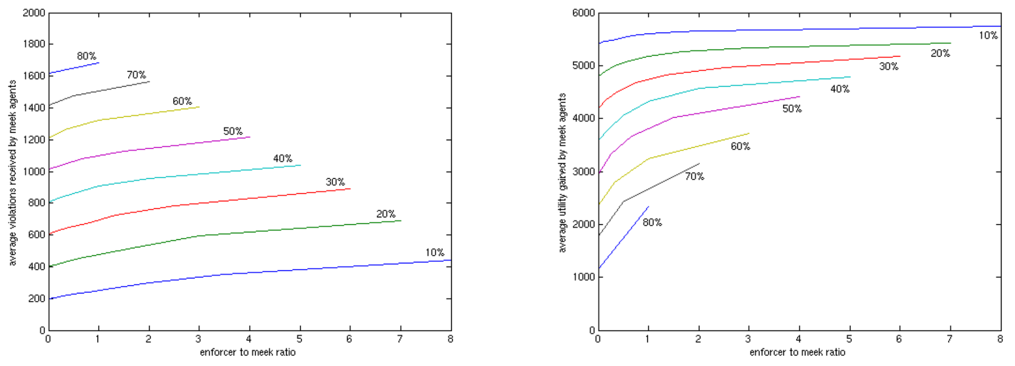
(b) norm-violating agents

Fig. 6. Utility gained by norm-abiding

It is interesting to note that even though meek agents receive more violations as the number of enforcer agents grows (see Figure 7(a)), the utility gained by them surprisingly increases (see Figure 7(b)). This is due to the fact that meek agents are still able to interact with other norm-abiding agents. Since violators are being blocked the ratio of defection to cooperation is lowered and the utility is increased.

5 Future Work

This paper is part of ongoing research on norm enforcement. Future work will relax the set of assumptions about agents, by giving them the ability to change



(a) Violations Received (b) Utility gained

Fig. 7. Local blocking rule increases both utility and violations to meek agents

their strategies in time, to lie, and to allow enforcer agents to violate the norm (*i.e.*, corrupt enforcers). The assumption of perfect information will be relaxed by adding uncertainty and noise. For these cases elaborate gossip techniques and reputation management will allow agents to enforce the norm. In future work the agent's reputation will be modeled not through gossip but through interaction overhearing. Mediating agents could overhear the interactions instead of waiting for interacting agents to report the outcome. More so, other conservative blocking strategies can be studied; such as blocking off agents that mediate norm violators, or blocking agents until they are shown to be norm-abiders.

Furthermore, the impact of other network parameters and dynamic networks will be analyzed. New links between agents could be added dynamically and test how this affects norm enforcement. New enforcement techniques will be studied to take advantage of dynamic networks.

Finally, other studies have shown that the efficiency of enforcement diminishes when enforcement conveys a cost to the enforcing agent [1, 8]. In future work there will be cost associated to blockage. One way to associate cost to enforcers is by removing their ability to stop agents from interacting with them. In this case, enforcers can withhold information from known violators, but if asked will have to interact with them and endure the norm violation.

Acknowledgments

This work is supported by the FP6 OpenKnowledge[4] Project. A. Perreau de Pinninck is supported by a CSIC predoctoral fellowship under the I3P program, and M. Schorlemmer is supported by a *Ramón y Cajal* research fellowship from Spain's Ministry of Education and Science, both of which are partially funded by the European Social Fund.

[4] `http://www.openk.org`

References

1. Robert Axelrod. An evolutionary approach to norms. *The American Political Science Review*, 80:1095–1111, 1986.
2. Guido Boella and Leendert van der Torre. Enforceable social laws. In *AAMAS '05: Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 682–689, 2005.
3. Jeffrey Carpenter, Peter Matthews, and Okomboli Ong'ong'a. Why punish: Social reciprocity and the enforcement of prosocial norms. *Journal of Evolutionary Economics*, 14(4):407–429, 2004.
4. Cristiano Castelfranchi, Rosaria Conte, and Mario Paoluccci. Normative reputation and the costs of compliance. *Journal of Artificial Societies and Social Simulation*, 1(3), 1998.
5. Jordi Delgado. Emergence of social conventions in complex networks. *Artificial Intelligence*, 141(1):171–185, 2002.
6. Amandine Grizard, Laurent Vercouter, Tiberiu Stratulat, and Guillaume Muller. A peer-to-peer normative system to achieve social order. In *AAMAS '06 Workshop on Coordination, Organization, Institutions and Norms in agent systems (COIN)*, 2006.
7. David Hales. Group reputation supports beneficent norms. *Journal of Artificial Societies and Social Simulation*, 5(4), 2002.
8. Douglas D. Heckathorn. Collective sanctions and compliance norms: a formal theory of group-mediated social control. *American Sociological Review*, 55(3):366–384, 1990.
9. James E. Kittock. The impact of locality and authority on emergent conventions: initial observations. In *AAAI '94: Proceedings of the Twelfth National Conference on Artificial Intelligence*, volume 1, pages 420–425, Menlo Park, CA, USA, 1994. American Association for Artificial Intelligence.
10. Josep M. Pujol, Jordi Delgado, Ramon Sangüesa, and Andreas Flache. The role of clustering on the emergence of efficient social conventions. In *IJCAI '05: Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, pages 965–970, 2005.
11. Michael Taylor. *Community, Anarchy & Liberty.* Cambridge University Press, 1982.
12. Adam Walker and Michael Wooldridge. Understanding the emergence of conventions in multi-agent systems. In Victor Lesser, editor, *Proceedings of the First International Conference on Multi–Agent Systems*, pages 384–389, San Francisco, CA, 1995. MIT Press.
13. Duncan J. Watts and Steven H. Strogatz. Collective dynamics of small-world networks. *Nature*, (393):440–442, 1998.
14. Fabiola López y López, Michael Luck, and Mark d'Inverno. Constraining autonomy through norms. In *AAMAS '02: Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 674–681, New York, NY, USA, 2002. ACM Press.
15. Stephen Younger. Reciprocity, sanctions, and the development of mutual obligation in egalitarian societies. *Journal of Artificial Societies and Social Simulation*, 8(2), 2005.

A Dynamic Coordination Mechanism Using Adjustable Autonomy

Bob van der Vecht[1,2], Frank Dignum[2], John-Jules Ch. Meyer[2], and Martijn Neef[1]

[1] TNO Defence, Safety and Security, The Hague,
`bob.vandervecht, martijn.neef@tno.nl`
[2] Department of Information and Computing Sciences, Universiteit Utrecht, Utrecht
`dignum, jj@cs.uu.nl`

Abstract. Agents in an organization need to coordinate their actions in order to reach the organizational goals. This research describes the relation between types of coordination and the autonomy of actors. In an experimental setting we show that there is not one best way to coordinate in all situations. The dynamics and complexity of, for example, crisis situations require a crisis management organization to work with dynamic types of coordination. In order to reach dynamic coordination we provide the actors with adjustable autonomy. Actors should be able to make decisions at different levels of autonomy and reason about the required level. We propose a way to implement this in a multi-agent system. The agent is provided with reasoning rules with which it can control the external influences on its decision-making.

1 Introduction

The motivation of this research lies in coordination challenges for crisis management organizations. Crisis situations in general are complex and share environmental features; there is no complete information, the evolvement of the situation is unpredictable and quick response is required. A crisis management organization should control the crisis as fast as possible, and therefore, it should be able to cope with such situations. For an adequate, quick response the organization needs high control. At the same time the organization needs to be able to adapt to unexpected events and therefore it needs to be dynamic and robust.

In this paper we describe different ways of coordination, and show that there is not one best way to coordinate in all situations. When modelling the decision-making process of the actors we see that there is always a trade-off between local autonomy and global control. In this paper we describe levels of autonomy in decision-making of actors, and we propose a way to implement adjustable autonomy in artificial actors in order to achieve a dynamic coordination mechanism.

In Section 2 we argue why we need dynamic coordination mechanisms in multi-agent systems. We describe the relation between types of coordination and the autonomy of actors. Using an experiment we point out the strong and the weak points of different coordination types. In Section 3 we define agent autonomy and we introduce adjustable autonomy as a concept that allows dynamically switching between coordination types. Section 4 proposes a way to implement adjustable autonomy in agents.

We extend the experiment with an implementation of adjustable autonomy. After that, Section 5 discusses our results and describes future research.

2 Why Dynamic Coordination?

In this section we argue why dynamic coordination mechanisms are relevant to achieve coordinated behavior in multi-agent systems. We discuss different types of coordination and their relation with the autonomy of the actors. Using an experiment we point out the weak and strong points of the coordination types and show that a static coordination mechanism is not optimal in all situations.

2.1 Autonomy and Coordination

All organizations designed for a certain purpose require coordinated behavior of the participants. There are several approaches to reach coordination, ranging from emergent coordination to explicit coordination by strict protocols. At the same time the actors in an organization are seen as autonomous entities that make their own decisions. In this paragraph we investigate the relation between autonomy of actors and coordination of behavior.

Autonomy is one of the key features of agents. It is often being used in the definition of agents [1]. In Jennings' use of the term, agent autonomy means that agents have control over both their internal state and over their behavior. The agent determines its beliefs and it decides by itself upon its actions. Multi-agent systems consist of multiple autonomous actors that interact to reach a certain goal. We will first take a closer look at coordination mechanisms for multi-agent systems.

One approach to reach coordinated group behavior is *emergent coordination*. Autonomous actors perform their tasks independently and the interaction between many of them leads to coordinated behavior. This approach is often used for agent-based social simulations. One characteristic of emergent coordination is that the actors have no awareness of the goals of the organization they are part of. The actors make their own local decisions and are fully autonomous. Although the actors have no organizational awareness, the designer of such a system has. The coordination principles are specified implicitly within the local reasoning of all actors. The organization is relatively flexible within the single task for which it has been designed. However, in the extreme case, the agents are fully autonomous, and there is no point of control that can force the organization to change its behavior if unexpected situations occur that cannot be solved by the local reasoning rules of the actors.

Where the fully emergent approach is one extreme type of coordination, the other extreme is fully *controlled coordination*. This is the case in a hierarchical organization, where there is a single point of control that determines the tasks all the others have to perform. The actors are autonomous in performing their task, but they do not make their own decisions. Therefore, the actors do not meet the autonomy definition as used in [1].

A characteristic of such a centralistic approach is that the task division is made from a global perspective. Therefore an organization can adapt quickly to changes in the environment by sending out new orders to all actors. However, such an organization is sensitive to incomplete information. Wrong information at the global level can lead to wrong decisions. Furthermore, the organization is highly dependent on the decision maker at the top of the hierarchy and it misses the flexibility at the local level. Fully controlled coordination can be a good solution if there is always complete information about the situation. Task specifications and interaction protocols can be defined for all possible cases.

In between the two extreme types there are several ways to achieve coordination. For example, the designer can allow the agents to communicate and exchange information. Or he can divide the organizational task in roles, and define the interaction in protocols. This is the approach that is taken in several methodologies for multi-agent systems design, e.g. Opera [2]. Drawback of those approaches is that the specified coordination rules are static. There is no flexibility within the predefined roles and interactions.

2.2 Experiment

We have set up an experimental environment in which we can test the characteristics of coordination principles. A simple coordination task is performed by an organization, and different scenarios contain situational features that can reveal the strong and the weak points of each coordination mechanism.

Organizational Description The basic setting is a Firefighter organization. The organization operates in a world where fires appear that need to be extinguished as fast as possible. In the organization we define two roles; *coordinator* and *firefighter*. The coordinator makes a global plan and tells the firefighters which fire they should extinguish. Therefore the coordinator has a global view of the whole world. The firefighters perform the actual tasks in the world; they move to a fire location and extinguish the fires. They have only local views.

There is a hierarchical relation between the two roles, the coordinator is superior of the firefighters and can send orders to the firefighters, which fire they have to extinguish. We want to show different forms of coordination within this organization. In our implementation we achieve this by changing the autonomy level of the decision-making process of the firefighters. We have created different types of firefighters; obedient agents that follow the orders of their superior (no decision-making autonomy) and disobedient agents that ignore their superior and make their decisions only based on local observations. Now we can describe the coordination types:

– *Emergent coordination*: disobedient firefighters, choices are made based on local information
– *Explicit coordination*: obedient firefighters, choices are made based on global information

The performance of the organization should be measurable. In our experiment we can measure the time it takes to extinguish the fires for each of the coordination types. The best organizational performance has the lowest score.

Scenarios We will describe the scenarios in more detail. The organization in our experiment has one coordinator and four firefighters. The start position of the firefighters in the world is equally distributed. We have one standard scenario, scenario A, in order to test whether both coordination types perform equally well. In this scenario four fires are distributed equally over the world. The start situation of scenario A is shown in Figure 1.

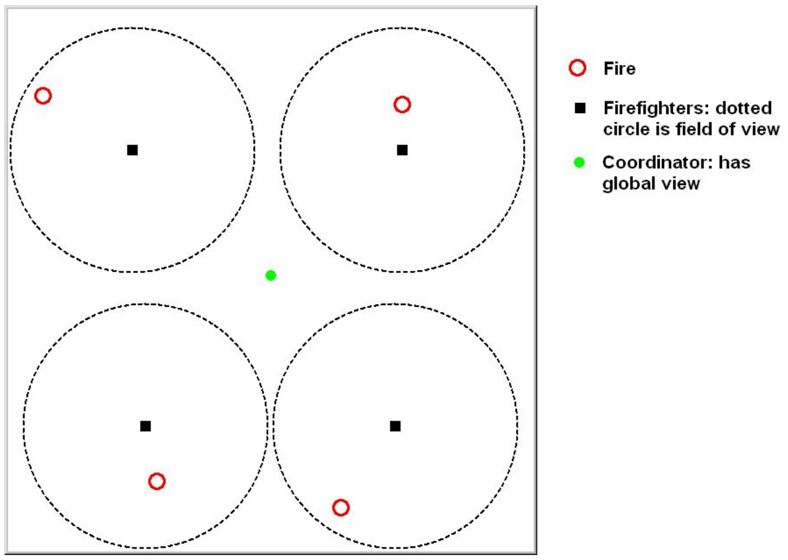

Fire
Firefighters: dotted circle is field of view
Coordinator: has global view

Fig. 1. Screenshot of the experimental environment: begin situation of scenario A

Two other scenarios have been created that make this situation more complex. They contain the features that also return in real world situations. Scenario B is a setting where the fires are distributed equally over the world, but the coordinator has no global view, he can only see half of the world at once. As result there is no complete information at the global level. The third scenario, Scenario C, reflects a situation where the fires are not distributed equally.

Results The results of the experiment are shown in Table 1. The score is equal to the time it took until all fires where extinguished and is measured per scenario and coordination type. Scenario A shows no significant difference in the performance of both organizations. In scenario B the firefighters reach a better performance based on their local information than the coordinator based on its information. The coordinator has no complete knowledge, and therefore he might miss important information for his planning task. In scenario C the fires were not equally distributed. The global information of the coordinator was more useful than the local information of the firefighters.

The difference between the two organizations was that the decisions were made at a different level of the organization and based on different information. None of the levels proved to be sufficient for all situations. We can conclude that in a scenario with a dynamic environment in which the agents experience these situations successively,

Table 1. Results of our Experiment; time (s) until all fires are extinguished per scenario and coordination type

	Explicit Coordination: No Autonomy	Emergent coordination: Full Autonomy
Scenario A: Standard scenario	38.7	36.8
Scenario B: No complete global information	93.8	69.8
Scenario C: No equal distribution of fires	36.8	66.6

both coordination types perform badly because of the weak points that are pointed out in the previous scenarios.

2.3 Dynamic Coordination

From our experiment, we can conclude that a dynamic coordination mechanism can outperform the presented organizations in a dynamic environment. In each coordination mechanism mentioned in Section 2.1 the autonomy of the actors with respect to the organization is fixed. We want to achieve dynamic coordination by allowing the agents to make local decisions about their autonomy level. We want them to act following organizational rules, but also allow them to decide not to follow the rules in specific situations. We believe that organizations in complex environments can benefit from agents that show *adjustable autonomy*. In the next paragraph we define adjustable autonomy in more detail and propose a way to achieve this in artificial agents.

3 Adjustable Autonomy

In this section we explain the concept of adjustable autonomy. Recall the autonomy requirement for agents as it is used by [1]. It states that agents should have control over their internal state and their behavior. We have argued that this conflicts with the extreme form of explicit coordination. The agents just follow orders and they do not determine their own actions.

We will take a closer look at agent decision-making. We believe that the decision-making process can take place at different levels of autonomy. An autonomous agent should be able to select its style of decision-making. This process is what we call *adjustable autonomy*. In this section we define levels of autonomy in agent decision-making and we propose a way to implement adjustable autonomy in agents.

3.1 Autonomy Levels in Agent Decision-Making

The difference between the two agent types in the experiment, obedient and disobedient, was the knowledge they used for their own decision-making. With autonomous decision-making the agent makes its own decisions based on its own observations, disregarding information and orders from other agents. The other extreme is that agents perform only commands that are given, and do not choose their actions based on their own knowledge.

The degree of autonomy of decision making can be defined as the degree of intervention by other agents on the decision making process of one agent [3]. Using this definition, the disobedient agent from our experiment makes its decisions autonomously, whereas the obedient agent had no autonomy at all concerning the decision making. An agent that switches between different levels of autonomy of its decision-making shows adjustable autonomy. We propose a reasoning model in which different levels of autonomy can be implemented.

3.2 Adjustable Autonomy

An agent's level of autonomy is determined by the influence of other agents on the decision-making process. Adjustable autonomy implies that the level of autonomy in the decision-making process can be adjusted. Therefore, an agent should control external influences that it experiences. The agent should choose which knowledge it uses for its decision-making. Figure 2 shows the reasoning process of an agent schematically. The module for event-processing determines the level of autonomy of the decision-making process.

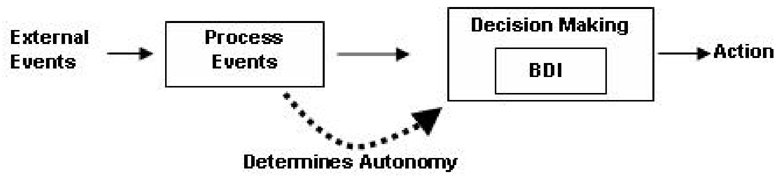


Fig. 2. The adjustable autonomy module within the reasoning process

In the reasoning model the agent is provided with reasoning rules that give him control over external influences. These external influences are the agent's own observations and messages that it gets from other agents. The agent can make an explicit choice about the knowledge that it will use for its decision-making process.

3.3 Related Work on Adjustable Autonomy

The topics *agent autonomy* and *adjustable autonomy* have been subject of many studies. However, there is no common definition of autonomy. As a result, the approaches taken to tackle the problem are quite distinct. We discuss the concept of autonomy and the way it is used in related work. And we investigate what adjustability is in the different perspectives that are taken. We will relate the other views on autonomy with our own view.

Castelfranchi and Falcone, [4] [5], have investigated autonomy in the context of (social) relations between agents. Considering a hierarchical relation, the abstraction level of decision-making of the delegate determines the agent's level of autonomy with respect to the master. Levels of autonomy they distinguish are *executive autonomy* (agent is not allowed to decide anything but the execution of delegated task), *planning autonomy* (agent is allowed to plan (partially), the delegated task is not fully specified) and *goal autonomy* (agent is allowed to find its own goals). Verhagen, [6], has added *norm autonomy* (the agent is allowed to violate organizational norms) as an extra level.

Adjustable autonomy is the process of switching between the abstraction levels of decision making. The autonomy levels as presented above concern goals, actions, plans and norms. We believe that also *beliefs* should be part of the autonomy definition, since beliefs are another concept used in the reasoning process. If an agent does not control its own beliefs, it can hardly be called autonomous. In our definition the autonomy level is gradually related to the influence an agent allows on its decision-making process. We propose reasoning rule that capture more explicit knowledge for reasoning about autonomy.

Schurr et al. [7] and Tambe et al. [8] use the term adjustable autonomy for the process in which a decision maker transfers the control of the decision-making process to another agent (or human). The researchers do not give a definition of autonomy, but it is related to decision-making control with respect to a certain goal. A coordination mechanism that runs independent of the agent's decision-making, handles the transfer-of-control (t-o-c) process. A t-o-c strategy consists of a list of decision makers and the constraints for transferring the control. An agent's position in the list of decision-makers determines an agent's level of autonomy with respect to the goal. They do not use autonomy as a gradual property of the decision-making process of the agent itself. Their reasoning mechanism for adjustable autonomy can only be used when there are more agents that have the capability to making the decision. The mechanism should make sure the optimal decision maker is selected.

In contrast, our approach focuses on the decision-making process of a single agent. The agent should select the optimal input (beliefs, goals, plans) for its own reasoning process. Those resources determine the autonomy level of a reasoning process. We look at adjustable autonomy as a process within an agent's reasoning, whereas they view it as a separate mechanism.

Barber and Martin, [9], look at the decision-making process of a group of agents. An agent's level of autonomy with respect to a task is measured as its share in the group decision-making process. In their context adjustable autonomy concerns different decision-making strategies for a group of agents. They present an Adaptive Decision-Making Framework, in which agents propose strategies to the group, and therewith change their own autonomy level. This way, adjustable autonomy becomes a group process, because other agents can accept or reject proposed decision-making strategies.

The focus of Barber and Martin is on the decision-making process of a group of agents. In contrast, our focus is on the decision-making of a single agent. In our work, adjustment of the autonomy is a local process within the agent's reasoning process. Furthermore Barber and Martin do not specify how an agent can determine the right decision-making strategies. In the experiments they conducted they provided the agents with knowledge about the best strategy for each situation. We want the agents to reason about what the best strategy is, based on local observations.

Dastani et al., [10], argue that the deliberation cycle of an agent determines autonomy of an agent as well. Autonomy levels can be viewed at as an agent's commitment to its own decisions. For example, one deliberation cycle makes that an agent commits to a goal until it has been fulfilled, whereas another cycle makes an agent to reconsider its goals every time it receives new information. They propose a meta-language to describe the deliberation cycle of an agent. The functions used in the deliberation cycle as well as their actual implementation are relevant for agent autonomy. Levels of autonomy can be constructed changing the deliberation cycle.

In their approach, levels of autonomy are determined by the deliberation cycle, and therefore by the way decisions are made. Our approach focuses on the sources that are used for decision-making and on the process of how an agent determines its autonomy level. The two approaches can exists next to each other and complement each other.

As we see in this discussion of related work there is not a single definition of agent autonomy and adjustable autonomy. Sometimes autonomy and adjustable autonomy is viewed in the context of group decision-making, whereas others look at single agent decision-making. Furthermore different aspects of agent decision-making are taken into account, such as decision-making control or abstraction levels of decision-making. Our approach is to give the agent control over the external influences it experiences.

4 Agent Reasoning Model

Here we present a reasoning model for agents that enables the agent to control its autonomy level. The level of autonomy depends on the influence of other agents on the reasoning process. In the reasoning-process we distinguish a phase for event-processing and a phase for decision-making, as shown in figure 2. The event-processing phase gives the agent control over its autonomy. The decision phase focuses on the decision on action. We describe the implementation of the two phases, starting with the latter one.

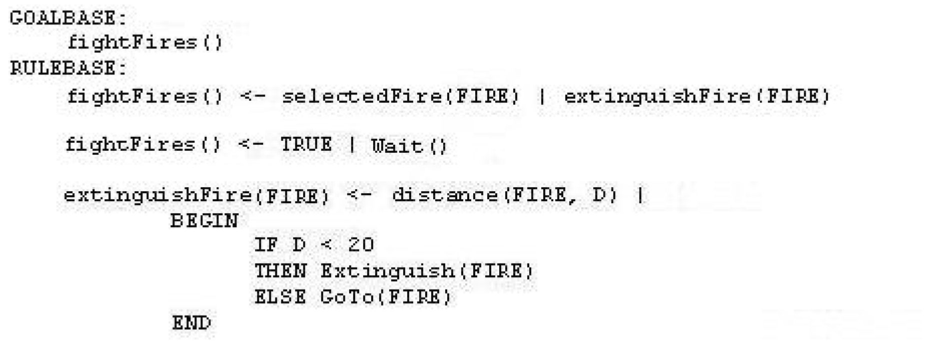
4.1 Decision Making

In the decide-phase the agent will decide upon the next action. A popular approach for goal-directed reasoning is to use of Beliefs, Desires and Intentions (BDI), introduced by Rao and Georgeff [11]. Several BDI reasoning-models have been proposed. For example, 3APL [12], [13] provides the designer with a formalized programming language which is designed for BDI-agent programming. A 3APL agent uses reasoning rules to create plans to reach a certain goal. Such reasoning rules have the following form:

```
<HEAD> <- <GUARD> | <BODY>
```

The head of a rule should match the goals of an agent. The guard should match the beliefs of the agent. The body of the agent contains sets of actions. If head and body match, the agent can commit to the plan in the body and start to execute it.

The firefighters in our experiment have been implemented using 3APL. They have a goal to fight fires and they have reasoning rules to reach their goal. Figure 3 shows the source code. If they have a certain fire selected, they are going to extinguish it. If no fire is selected, they wait. Depending on the distance to this fire, they will perform either the action GoTo or Extinguish.



```
GOALBASE:
    fightFires()
RULEBASE:
    fightFires() <- selectedFire(FIRE) | extinguishFire(FIRE)

    fightFires() <- TRUE | Wait()

    extinguishFire(FIRE) <- distance(FIRE, D) |
            BEGIN
                  IF D < 20
                  THEN Extinguish(FIRE)
                  ELSE GoTo(FIRE)
            END
```

Fig. 3. Source code of 3APL plan to fight fires

Each decision the agent takes depends on its beliefs. The beliefs that are used in this plan are: *selectedFire* and *distance*. These beliefs are determined before the plan reasoning starts. Therefore we describe the event-processing phase, which prepares the actual decicion-making phase.

4.2 Event Processing

In the event-processing phase the agent prepares the decision-making phase. External influences are processed here. External influence can be an agent's observations or messages from other agents. We have chosen to implement the orient phase using 3APL rules as well. This gives us the opportunity to reason with semantic knowledge. The main process consists of three functions: *handle observations*, *handle messages*, and *prepare decision-making*.

The autonomy level of the decide phase is determined by those functions. Will the agent follow the commands from the coordinator, or will it create own goals? Does the agent adopt information from the coordinator, or does it use its own observations? We show how we can implement reasoning rules that provide the agent with choices. We will take the firefighters from our experiment as example.

Reasoning rules can be added to make the agent choose to handle observations differently. We gave one rule to our firefighters, which states that is believes all its own observations:

```
handleObservations() <- TRUE | Observations2Beliefs()
```

Our firefighters use only this rule for observation processing. It is possible too add more rules that distinguish between different situations. To use the rule, the guard of the rule has to match with the beliefs of the agent. Adding rules with a specified guard, the agent handles its observations differently if that guard is true.

Agents can receive messages from other agents. An agent can be programmed to handle messages in different ways by adding the same types of rules. If an agent functions in an organization, it needs to know how to deal with relations towards other agents. We have implemented the following rule for a hierarchical relation. When the agent gets a request from another agent who is his superior, he interprets the content as a command.

```
handleMessages() <- message(SENDER, request, CONTENT)
  AND superior(CONTENT) | AcceptCommand(SENDER, CONTENT)
```

The firefighters believe that the coordinator is their superior. They will process the requests of the coordinator as commands. In a similar manner other rules that can be defined. For example, an agent can have a rule to ignore all messages when it feels it is in danger.

```
handleMessages() <- danger() | ignoreMessages()
```

If an agent has both rules for message handling it is dependent on the agent whether it processes messages or not. Does the agent perceive danger or not? By adding such a rule, local beliefs of the agent can change the way it handles external influences, and therefore it can influence the autonomy level of the agents' decision-making.

Finally, in the function *prepare decision-making* rules are specified that determine the autonomy level of the agent. The reasoning rules in the decide-phase use certain beliefs. Here we specify per goal what kind of belief processing should take place. Recall from Figure 3 that the beliefs that are used for the goal to fight fires are *selectedFire* and *distance*. We have specified the following rules:

```
prepareDecisionMaking() <- goal(fightfires) AND
  command(FIRE) | SelectFire(FIRE); CalculateDistance(FIRE)

prepareDecisionMaking() <- goal(fightfires) AND noCommand()
  AND seeFire(FIRE) | SelectFire(FIRE); GetDistance(FIRE)
```

These two rules specify how the beliefs for the decision-making process are determined dependent on the situations. The SelectFire and CalculateDistance statements are capabilities of the agent that construct the selectedFire and the distance belief respectively. The variable given to those functions has a different origin in both cases. If the agent has a command, he will follow the command. If there is no command, but the agent sees a fire, it will use this observation for further reasoning.

5 Extending the Experiment

We have extended the experiment of Section 2. We have constructed a third organization with firefighters that show adjustable autonomy. They are at certain moments disobedient to the commands of the coordinator and at other moments they follow the orders, depending on their local beliefs. So, the organization can switch between explicit coordination and emergent coordination. We have implemented reasoning rules for event processing, we have used the same rules as presented in the section 4.2.

5.1 Results

We have run all three scenarios. Table 2 shows the results. We can see that the organization with agents that use adjustable autonomy performs well in all scenarios compared to the other two organizations. The organization adapts its coordination mechanism to the environmental features.

Table 2. Results of our Experiment, including adjustable autonomy

	Explicit coordination	Emergent coordination	Adjustable autonomy
Scenario A: Standard scenario	38.7	36.8	37.0
Scenario B: No complete global information	93.8	69.8	70.2
Scenario C: No equal distribution of fires	36.8	66.6	37.1

From the experiment we can conclude that dynamic coordination is powerful in agent organizations; the organization using adjustable autonomy will perform well in dynamic scenarios. The way we achieve a dynamic coordination mechanism, is by letting the agents adjust their autonomy level. The agents have reasoning rules to control external influences in the reasoning process. The agents decide locally on their autonomy level.

5.2 Discussion

We provide the agents with reasoning rules to control external influences. This gives the agents additional, not task-specific knowledge that it can use in its reasoning process. It allows the agent to use its beliefs and its goals to reason about its openness towards other agents. The reasoning rules make use of criteria based on *introspection*, *social knowledge*, or *coordination requirements*.

Using introspection, the agent assesses its own mental state. Castelfranchi, [4], argues the importance of introspection in the reasoning process. For example, *relevance of information* can be determined by introspection. Certain information can be more or less relevant depending on an agent's goals. Therefore an agent may observe the world differently depending on its goals.

An agent may have a reasoning rule that makes the agent react differently to external input when it feels danger than when it feels at ease. To make such adaptive behavior possible, the agent also needs to have the capability to determine when it is in danger.

Social and organizational knowledge are other examples of criteria that can be used to control external influences. The importance of explicitly modelling organizational awareness for coordination is argued by Oomes [14]. For example, knowledge about the sender of a message is useful when deciding what to do with the content. If we assume that an organization is implemented following a methodology as Opera [2], organizational concepts are available in the beliefbase. By using them in reasoning rules, we add the social knowledge to the reasoning process of the agents.

The third example of knowledge that can be used for autonomy adjustment is knowledge about coordination requirements. Given that an agent acts in a coordination mechanism, it can encounter environmental changes that influence the coordination. For example, if an agent follows orders from a superior and the communication fails at a certain moment, it can choose to increase its autonomy in order to fulfill the goals.

We will conduct more experiments to develop general heuristics that an agent can use to control external influences. This way, we want to combine single-agent decision-making and multi-agent interaction to develop dynamic coordination mechanisms.

6 Conclusion

There are several ways to achieve coordination within an agent organization. Approaches range from emergent coordination, where the actors are autonomous and the coordination is implicitly implemented, to explicit coordination, such as a hierarchical organization where the actors have no decision autonomy but just follow the orders from their superiors. We have shown that there is not one best way to coordinate in all situations. Complex and dynamic situations therefore require a dynamic coordination mechanism.

We have implemented a dynamic coordination mechanism by providing the actors with adjustable autonomy. An agent's level of autonomy depends on the influence of others on the reasoning process. The actors have reasoning rules that control the external influences they experience. This way we have defined situations at the individual level in which the actor can change its autonomy level. In addition to the task specific domain knowledge, the knowledge for event processing is used in the agent's reasoning process.

References

1. Jennings, N.R.: On agent-based software engineering. Artificial Intelligence **117**(2) (2000) 277–296
2. Dignum, V.: A Model for Organizational Interaction: based on Agents,founded in Logic. Utrecht University, PhD Thesis (2004)
3. Barber, K., Han, D., Lui, T.H.: Strategy selection-based meta-level reasoning for multi-agent problem-solving. In: Ciancarini, P., Wooldridge, M.J. (eds): Agent-Oriented Software Engineering: AOSE 2000. Lecture Notes in Computer Science **1957** (2001) 155–187
4. Castelfranchi, C.: Guarantees for autonomy in cognitive agent architecture. Intelligent Agents (890) (1995) 56–70
5. Falcone, R., Castelfranchi, C.: The human in the loop of a delegated agent: the theory of adjustable social autonomy. IEEE Transactions on Systems, Man, and Cybernetics, Part A **31**(5) (2001) 406–418
6. Verhagen, H.: Norm Autonomous Agents. Stockholm University, PhD Thesis (2000)
7. Schurr, N., Marecki, J., Lewis, J., Tambe, M., Scerri, P.: The defacto system: Training tool for incident commanders. AAAI05 (2005)
8. Tambe, M., Scerri, P., Pynadath, D.: Adjustable autonomy for the realworld. Journal of Artificial Intelligence Research (17) (2002) 171–228
9. Barber, K., Martin, C.: Agent autonomy: Specification, measurement, and dynamic adjustment. Proc. of the Autonomy Control Software Workshop at AA-1999 (1999) 8–15
10. Dastani, M., Dignum, F., Meyer, J.J.C.: Autonomy and agent deliberation. Agents and Computational Autonomy 2003 (2003) 114–127
11. Rao, A.S., Georgeff, M.P.: BDI-agents: from theory to practice. In: Proceedings of the First Intl. Conference on Multiagent Systems, San Francisco (1995)
12. Dastani, M., van Riemsdijk, B., Dignum, F., Meyer, J.J.C.: A programming language for cognitive agents: Goal directed 3apl. PROMAS 2003 (2003) 111–130
13. Hindriks, K.V., de Boer, F.S., van der Hoek, W., Meyer, J.J.C.: Agent programming in 3apl. Autonomous Agents and Multi-Agent Systems **2**(4) (1999) 357–401
14. Oomes, A.H.J.: Organization awareness in crisis management. Proceedings of the International Workshop on Information Systems on Crisis Response and Management (ISCRAM) (2004)

Model Checking Norms and Sanctions in Institutions [⋆]

Francesco Viganò[1] and Marco Colombetti[1,2]

[1] Università della Svizzera italiana, via G. Buffi 13, 6900 Lugano, Switzerland
{francesco.vigano,marco.colombetti}@lu.unisi.ch,
[2] Politecnico di Milano, piazza Leonardo Da Vinci 32, Milano, Italy
marco.colombetti@polimi.it

Abstract. In this paper we enrich FIEVeL (a modelling language for institutions amenable to model checking) with new constructs to describe norms and sanctions. Moreover, we present a specification language to reason about the effectiveness of norms and sanctions in shaping agent interactions. Finally we show that when properties of artificial institutions reflect certain interpretations of norms of human institutions, it is not always possible to satisfy them. As a consequence, regimentation of norms is not always a viable solution.

1 Introduction

Rules defined by *artificial institutions* and enforced by their software implementations, named *electronic institutions* [5], have been put forward as means to regulate open multiagent systems. Institutions define two kinds of rules [17]: *norms* (also named *regulative rules* [17]), which regulate existing activities, and *constitutive rules*, which create the very possibility of certain institutional actions. Artificial institutions are often designed to reflect constitutive and regulative rules defined by human institutions in artificial systems [10, 9, 7], and model checking can play an important role to evaluate the compliance of artificial institutions with rules of human institutions and to compare design alternatives arising from different interpretations of such rules. In particular, when we map human rules only onto constitutive rules of artificial institutions, we obtain systems where violations cannot occur (they are regimented [10, 9]). Instead, when we introduce regulative rules into artificial institutions, we obtain systems where violations may occur due, for instance, to the agents' autonomy. This fact is particularly important when we consider results obtained by a model checker: if a norm of a human institution has been mapped onto a set of constitutive rules of an artificial institution and a property that reflects it does not hold, then the artificial institution is incorrect. Instead, when a norm n has been mapped onto regulative rules of the artificial institution, we have to analyze whether: (i) norms of the artificial institution are correct, that is, a property reflecting expected effects of norm n holds over paths compliant with norms, and (ii) sanctions applied when norms are violated enforce desirable effects of norm n over all other possible evolutions.

The main contributions of this paper are threefold: first, we extend FIEVeL [19], a modelling language for institutions amenable to model checking, with new constructs

[⋆] Supported by the Swiss National Science Foundation project 200020-109525, "Artificial Institutions: specification and verification of open distributed interaction frameworks."

to describe norms and sanctions, exemplifying how norms can be defined and enforced with our language; second, we present a flexible specification language which provides temporal operators that select paths compliant with certain sets of norms, showing that existing proposals (e.g. [12, 16, 1]) can be reduced to particular patterns of specification of our language; finally, we contribute to the ongoing debate about *regimentation* and *enforcement* of norms [10, 9, 6, 8], showing that when human institutions impose a specific interpretation of norms, it may be the case that properties that reflect them cannot be satisfied by artificial institutions under the assumption that agents are autonomous. As a consequence, *regimentation* of norms is not always a viable solution.

The remainder of this paper is structured as follows: Section 2 introduces the OMSFOTL logic which is used to define the semantics of FIEVeL and to state properties of institutions in Section 3, where we provide an overview of our framework by resuming results discussed in our previous works. Section 4 presents how norms can be described with FIEVeL, while Section 5 introduces a language to define properties which consider only evolutions of institutions that comply with certain sets of norms. Section 6 explains how to formalize sanction mechanisms with FIEVeL and finally Section 7 provides a comparison of our approach with related works and presents some conclusions.

2 Ordered Many-Sorted First-Order Temporal Logic

An ordered many-sorted first-order temporal logic (OMSFOTL) is a many-sorted first-order logic [13] enriched with temporal operators and hierarchies of sorts. The signature of an OMSFOTL logic consists of a finite nonempty set of *sort symbols* Σ, a *hierarchy of sorts* $\leq_\Sigma$ (where $\sigma_1 \leq_\Sigma \sigma_2$ means that sort σ_1 is a *subsort* of sort σ_2), finite sets of *constants* (C), *function symbols* (F), and *predicate symbols* (P), and a denumerable set of *variables* (V). Moreover, an OMSFOTL signature defines function ξ which assigns a sort to every variable and every constant, and a signature (i.e. a sequence of sorts) to every function and predicate symbol. Given sorts Σ, the set T_σ of *terms of sorts* σ is the smallest set such that:

- $v \in \mathrm{T}_\sigma$ if $v \in \mathrm{V}$ and $\xi(v) \leq_\Sigma \sigma$;
- $c \in \mathrm{T}_\sigma$ if $c \in \mathrm{C}$ and $\xi(c) \leq_\Sigma \sigma$;
- $f(t_1, ..., t_n) \in \mathrm{T}_\sigma$ if $f \in \mathrm{F}$, $\xi(t_i) \leq_\Sigma [\xi(f)]_i$ for $1 \leq i \leq n$ and $[\xi(f)]_0 \leq_\Sigma \sigma$

where $[\xi(q)]_i$ refers to the i-th sort of the signature of a predicate or function symbol q. The set T of *terms* is the union of the sets T_σ for all $\sigma \in \Sigma$ and the set A of *atomic formulae* is the smallest set such that:

- $(t_1 = t_2) \in \mathrm{A}$ if there exists sort σ such that $\xi(t_1) \leq_\Sigma \sigma$ and $\xi(t_2) \leq_\Sigma \sigma$;
- $p(t_1, ..., t_n) \in \mathrm{A}$ if $p \in \mathrm{P}$ and $\xi(t_i) \leq_\Sigma [\xi(p)]_i$ for $1 \leq i \leq n$.

The set of *formulae* is defined according to the following grammar:

$$\varphi ::= \alpha \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall\varphi \mid \mathbf{X}\varphi \mid \varphi\mathbf{U}\varphi \mid \mathbf{E}\varphi \mid \mathbf{G}\varphi$$

where α is an atomic formula. Expressions *true*, *false*, $(\varphi \vee \psi)$, $(\psi \rightarrow \varphi)$, $(\varphi \leftrightarrow \psi)$, $\exists x\varphi$, $\mathbf{F}\varphi$, and $\mathbf{A}\varphi$ are introduced as abbreviations as usual.

In [20] we have shown that if we assume that each sort σ is associated to a finite domain D_σ, then OMSFOTL is as expressive as CTL* [4, 3] and its models can be encoded with a finite number of atomic propositions. Despite it, we adopt OMSFOTL for two main reasons: (i), it represents an abbreviated form for long and complex formulae and (ii), institutions describe rules that typically are independent of the cardinality of domains and which can be naturally expressed by allowing quantification over sorts.

3 Describing, Specifying, and Verifying Institutions

In [19] we proposed a metamodel of institutions based on the notion of an agent status function, which can be interpreted as a position involving a (possibly empty) set of institutionalized powers [11], obligations, prohibitions, etc. To formalize status functions and related concepts, we map them onto sorts, functions, and predicates of an OMSFOTL signature and define a set of axioms to capture their interrelations and temporal evolution. For instance, common aspects of status functions are represented by introducing sort σ_{sf}, which also defines the function $subject$ denoting the agent (σ_{aid}) the status function has been assigned to. Sort σ_{sf} also induces the two predicates $assigned$ and $modified$, which respectively represent if a status function is currently assigned (or revoked) and if it has been modified by the occurrence of an institutional event. Finally, the metamodel defines a set of axioms based on such symbols, for instance requiring that if a status function is not affected, then its subject does not change:

$$\mathbf{AG}\forall f(\neg\mathbf{X}modified(f) \rightarrow \exists a(subject(f) = a \wedge \mathbf{X}subject(f) = a)) \qquad \text{(A.1)}$$

An institution evolves because events (σ_{ev}) occur or agents perform actions ($\sigma_{act} \leq_\Sigma \sigma_{ev}$). Each event-type e induces a sort σ_e and three predicates, $happens_e$, $prec_e$, and eff_e, which express when an event of type e happens and what conditions must be satisfied before and after its occurrence. In contrast with *base-level events* (e.g., *exchange-message* events), the occurrence of an *institutional event* (σ_{ie}) requires that another event conventionally associated to it occurs and that, in the case of institutional actions, the actor must be empowered to perform it:

$$\begin{aligned}\mathbf{AG}\forall\overline{x}((prec_{ia}(\overline{x}) \wedge \exists f(subject(f) = x_1 \wedge empowered_{ia}(f, \overline{x}) \wedge assigned(f)\\ \wedge \bigvee_{a\in\sigma_{act}} \mathbf{X}(conv_{a-ia}(\overline{x}) \wedge happens_a(\overline{x}'))) \leftrightarrow \mathbf{X}happens_{ia}(\overline{x}))\end{aligned} \qquad \text{(A.2)}$$

where: $\overline{x}$ is a set of variables determined by predicate $happens_{ia}$; the first variable of $\overline{x}$ refers to the actor of action ia; predicate $empowered_{ia}$ states when status functions are empowered to perform institutional action ia; predicate $conv_{a-ia}$ represents the existence of a convention among action a and institutional action ia; and $\overline{x}'$ reflects how arguments of ia are mapped over arguments of action a.

To model institutions in terms of the concepts described by our metamodel, in [19] we introduced FIEVeL, a modelling language for institutions, whose syntax is exemplified in Figure 1 and whose semantics is given by providing a translation of its constructs

```
basic-sorts:
  $\sigma_{resources}$;
  $\sigma_{reqState}$ = {answ,notAnsw};
base-events:
  message giveResource(rec:$\sigma_{aid}$,res:$\sigma_{resources}$);
      ...
institution resourceManagement {
  status-function member() {...}
  status-function requested(reqRes:$\sigma_{resources}$,ag:$\sigma_{aid}$,sta:$\sigma_{reqState}$){...}
  status-function holder(resource:$\sigma_{resources}$){
  key resource;
  powers give <- (∃ r:$\sigma_{requested}$ (assigned(r)∧reqRes(r)=resource(f)
          ∧ag(r)=rec∧sta(r)=answ)∧res=resource(f));
  }
      ...
  institutional-events:
     institutional-action give(rec:$\sigma_{aid}$,res:$\sigma_{resources}$)
     pre ∃ x:$\sigma_{member}$(assigned(x)∧subject(x)=rec∧¬subject(x)=actor);
     eff r:$\sigma_{requested}$ revoke (reqRes(r)=res),
         r:$\sigma_{holder}$ assign (subject(x)=rec,resource(x)=res);
     ...
  conventions
     exch-Msg(giveResource) [true]=c=> give[rec=c=>rec res=c=>res]
     ...
}
```

Fig. 1. Fragments of the Resource Management institution.

into a set of symbols and formulae of an OMSFOTL logic. According to Figure 1, in the Resource Management institution a $member$ can request a $holder$ to $give$ the control of one of its resources. When an agent accepts to satisfy the request, it is empowered to give a resource to the agent that has requested it, which becomes its new $holder$. The model described in Figure 1 induces, among others, sort $\sigma_{holder} \leq_{\Sigma} \sigma_{sf}$, function $resource$ of signature $\xi(resource) = \langle \sigma_{resources}, \sigma_{holder} \rangle$, and predicate $happens_{give}$ such that $\xi(happens_{give}) = \langle \sigma_{aid}, \sigma_{aid}, \sigma_{resources} \rangle$.

In our framework, also properties are specified in terms of OMSFOTL formulae such that temporal operators ($\mathbf{X}$, $\mathbf{G}$, $\mathbf{F}$, and $\mathbf{U}$) are always preceded by a path quantifier ($\mathbf{E}$ or $\mathbf{A}$). One of the main advantages of our approach resides in the fact that any symbol introduced by our metamodel or by an institution can appear in a property. Furthermore, to increase the flexibility of the language, occurrences of events are referenced with a generic predicate $happens$ and we write "$x : \sigma$" to say that variable x is of sort σ. For instance, the following property requires that whenever an agent receives a positive answer to its requests, it will eventually become the *holder*:

$$\mathbf{AG}\forall act : \sigma_{aid}\forall rec : \sigma_{aid}\forall res : \sigma_{resources}(happens(accept, act, rec, res) \\ \rightarrow \mathbf{AF}\exists h : \sigma_{holder}(subject(h) = rec \wedge resource(h) = res)) \quad \text{(P.1)}$$

Analogously, we can also check if whenever a holder accepts to give a resource, it will eventually do so:

$$\mathbf{AG}\forall act : \sigma_{aid}\forall rec : \sigma_{aid}\forall res : \sigma_{resources}(happens(accept, act, rec, res) \\ \rightarrow \mathbf{AF}happens(give, act, rec, res)) \quad \text{(P.2)}$$

In [20] we presented a symbolic model checker specifically developed to verify FIEVeL institutions. Given an institution and a set of properties, our tool proceeds as follows: (i) it converts the institution into a set Φ of OMSFOTL formulae by considering the semantics of FIEVeL constructs and axioms determined by our metamodel; (ii) formulae Φ are translated into propositional logic and subsequently converted into a formula in conjunctive normal form (CNF); (iii) given the set of assignments satisfying the CNF (whose disjunction constitutes the transition relation of a Kripke structure) and a formula φ_0, representing a set of initial states, a symbolic representation of an institution is built and is exploited to verify properties by applying standard symbolic algorithms [3]. According to our model checker, properties (P.1) and (P.2) do not hold: since constitutive rules reported in Figure 1 define possible actions that agents can carry out, but do not ensure that empowered agents will necessarily perform them, it may be the case that agents accept to give their resources but do not perform action $give$.

4 Norms

To define the semantics of norms, our metamodel assumes the existence of sort σ_o, whose individuals reify norms of institutions. Sort σ_o is used to express prohibitions and obligations characterized by certain deadlines (not necessarily a time expression), and we consider that a state of affairs is permitted if it is reached without violating any norm. In particular, for the sake of conciseness, in this paper we focus only on norms which are considered fulfilled or violated only *once* after a given status function is imposed on an agent and certain conditions are met. Given sort σ_{state}, which introduces constants $unfired$, $activated$, and $inactive$, sort σ_o is characterized by function $state$ ($\xi(state) = \langle \sigma_{state}, \sigma_o \rangle$), which keeps trace of the temporal evolution of a norm, a set of timers (e.g., function $activation$ which counts how many time events have occurred since a norm has been activated), and by a set of predicates ($start$, $fulfillment$, and $violation$ of signature $\xi(violation) = \langle \sigma_{sf}, \sigma_o \rangle$). Agents are subject to norms when certain status functions are imposed on them: to model the interdependency among norms and status functions, we introduce function $ofStatus$ ($\xi(ofStatus) = \langle \sigma_{sf}, \sigma_o \rangle$) which denotes the status function an obligation is associated to. When a status function is not assigned, then its norms are considered to be $inactive$ and cannot be violated: we represent this fact by the following axiom, which states that norms of a revoked status function are always $inactive$:

$$\mathbf{AG}\forall o \forall f((ofStatus(o) = f \land \neg assigned(f)) \rightarrow state(o) = inactive) \quad \text{(A.3)}$$

where $\xi(o) = \sigma_o$ and $\xi(f) = \sigma_{sf}$. Similarly, Axiom (A.4) requires that when a status function is imposed on an agent, then the state of a norm is set to $unfired$ if predicate $start$ is not satisfied, otherwise it is set to $activated$:

$$\mathbf{AG}\forall o \forall f((ofStatus(o) = f \land \mathbf{X}(assigned(f) \land modified(f))) \rightarrow ((\neg start(o, f) \land \mathbf{X} state(o) = unfired) \lor (start(o, f) \land \mathbf{X} state(o) = activated))) \quad \text{(A.4)}$$

Axioms (A.3) and (A.4), as well as other axioms omitted here for the sake of brevity, describe the temporal evolution of functions $state$ and $activation$, which in combination with predicates $fulfillment$ and $violation$, determine when an obligation should be considered to be infringed. In particular, given predicate $violated$ of signature $\xi(violated) = \langle \sigma_o \rangle$, a norm is violated if and only if it was $activated$, the associated status function is not modified, $violation$ holds while $fulfillment$ is false:

$$\mathbf{AG}\forall o \forall f (ofStatus(o) = f \rightarrow (\mathbf{X}violated(o) \leftrightarrow (state(o) = active \wedge (violation(o) \wedge \neg fulfillment(o) \wedge \neg \mathbf{X}modified(f))))) \qquad \text{(A.5)}$$

Norms are described in FIEVeL according to the following syntax:

```
norm ::= symbol start fulfillment violation ;
start ::= "start" "<->" expression ";" ;
fulfillment ::= "fulfillment" "<->" expression ";" ;
violation ::= "violation" "<->" expression ";" ;
```

where `expression` is an OMSFOTL formula which does not contains **U**, **E**, **G**, or nested occurrences of **X**. Moreover, given that a norm is described within a status function σ_s, free occurrences of a variable f of sort σ_s may appear in any formula used to describe a norm's condition. A norm $symbol$ induces sort $\sigma_{symbol} \leq_{\Sigma} \sigma_o$ and defines under what conditions predicates $fulfillment$, $violation$, and $start$ hold when are evaluated over an obligation of sort σ_{symbol}, as exemplified by the following axiom schema:

$$\mathbf{AG}\forall o \forall f (fulfillment(o, f) \leftrightarrow (ofStatus(o) = f \wedge \texttt{expression})) \qquad \text{(A.6)}$$

where $\xi(o) = \sigma_{symbol}$ and $\xi(f) = \sigma_s$. Combining instances of Axiom Schema (A.6) (and similarly for predicates $violation$ and $start$) with Axiom(A.5), it is possible to automatically classify states with respect to each norm defined by an institution. In contrast with other approaches (e.g., [16] and [1]), in our framework designers can describe norms at a high-level in terms of institutional concepts, ignoring the actual number of states and transitions admitted by an institution. For instance, the following norm, named $h1$ and associated to the $holder$ status function, states that once a holder accepts to give the control of a resource, then it ought to do so before a certain time interval elapses:

```
h1  start<->X ∃ ag:$\sigma_{aid}$ ∃ rec:$\sigma_{aid}$ ∃ res:$\sigma_{resources}$ (subject(f)=ag ∧
            resource(f)=res ∧ happens(accept,ag,rec,res));
    fulfillment<->∃ ag:$\sigma_{aid}$ ∃ rec:$\sigma_{aid}$ ∃ res:$\sigma_{resources}$ (subject(f)=ag
            ∧ res=resource(f) ∧ X happens(give,ag,rec,res));
    violation<->(activation(o)=1 ∧ X happens(time));
```

Without proper sanction mechanisms, the introduction of norms typically does not change the set of properties satisfied by an institution, given that autonomous agents may not comply with such norms [5, 2, 9, 18, 7]: as a consequence certain properties may not hold in an institution even if its rules are correctly stated. For instance, properties (P.1) and (P.2) do not hold in the new model of the Resource Management institution obtained by adding norm $h1$, despite this correctly requires that a $holder$ gives a resource after it has positively answered to an agent. This is due to the fact that norms regulate existing activities, describing what evolutions of an institution should be considered as legal, but do not change the temporal evolution admitted by an institution.

5 Normed Temporal Operators

To analyze whether an institution may lead a system into certain states when its norms are respected, we can exploit predicate $violated$ and the fact that in our framework norms are reified as norm individuals. Therefore, it is possible to quantify over sort σ_o (and its subsorts induced by each norm), investigating how norms condition the evolution of an institution. In particular, in this paper we define operators that allow designers to reason about what properties are satisfied by an institution when a set of norm individuals are not violated. More precisely, given a set of norms which constitute the extension of formula φ_o (an open formula in which variable o of sort σ_o occurs free), *normed temporal operators* are defined as follows:

- $\mathbf{EG}^{\varphi_o}\varphi =_{def} \mathbf{EG}(\forall o : \sigma_o(\varphi_o \rightarrow \neg violated(o)) \wedge \varphi)$;
- $\mathbf{EX}^{\varphi_o}\varphi =_{def} \mathbf{EX}(\forall o : \sigma_o(\varphi_o \rightarrow \neg violated(o)) \wedge \varphi)$;
- $\mathbf{E}\psi\mathbf{U}^{\varphi_o}\varphi =_{def} \mathbf{E}(\forall o : \sigma_o(\varphi_o \rightarrow \neg violated(o)) \wedge \psi)\mathbf{U}(\forall o : \sigma_o(\varphi_o \rightarrow \neg violated(o)) \wedge \varphi)$;

Since the satisfaction of CTL temporal operators (with the exception of $\mathbf{EX}$) refers to the initial state π_0 of a path π [4, 3], then also their normed counterparts refer to state π_0. As a consequence, if state π_0 violates norms φ_o, then the normed operators $\mathbf{EG}^{\varphi_o}$ and $\mathbf{EU}^{\varphi_o}$ are trivially falsified. This may occur when the system is inconsistent or because normed temporal operators are nested and external operators do not ensure compliance with norms considered by internal operators. While in the former case we would conclude that our system is irrational, in the latter case we may get counter-intuitive results. To avoid this, we can prefix internal operators with $\mathbf{EX}^{\varphi_o}$, ensuring that the initial state is not considered and only paths compliant with norms of internal operators are taken into account. Despite this problem may be avoided by different definitions of normed temporal operators, we consider more relevant the fact that normed and unnormed operators are evaluated over the same set of states and are expressed in terms of the standard semantics of CTL [4, 3]. In doing so, if formula φ_o refers to an empty set of obligations, then normed temporal operators are equivalent to their temporal counterpart (e.g., $\mathbf{EG}^{false}\varphi \equiv \mathbf{EG}\varphi$), and $\mathbf{EG}^{\varphi_o}$, $\mathbf{EX}^{\varphi_o}$, and $\mathbf{EU}^{\varphi_o}$ constitute an adequate set of operators, since we have the following equivalences:

- $\mathbf{EF}^{\varphi_o}\varphi \equiv \mathbf{E}true\mathbf{U}^{\varphi_o}\varphi$;
- $\mathbf{AG}^{\varphi_o}\varphi \equiv \neg\mathbf{EF}^{\varphi_o}\neg\varphi \wedge \mathbf{EG}^{\varphi_o}true$;
- $\mathbf{AX}^{\varphi_o}\varphi \equiv \neg\mathbf{EX}^{\varphi_o}\neg\varphi \wedge \mathbf{EX}^{\varphi_o}true$;
- $\mathbf{A}\psi\mathbf{U}^{\varphi_o}\varphi \equiv \neg(\mathbf{E}\neg\varphi\mathbf{U}^{\varphi_o}(\neg\varphi \wedge \neg\psi)) \wedge \neg\mathbf{EG}^{\varphi_o}\neg\varphi \wedge \mathbf{EF}^{\varphi_o}\varphi$;
- $\mathbf{AF}^{\varphi_o}\varphi \equiv \neg\mathbf{EG}^{\varphi_o}\neg\varphi \wedge \mathbf{EF}^{\varphi_o}\varphi$;

It is worth observing that by definition, the consistency of norms represents a necessary condition for the satisfaction of normed temporal operators universally quantified over paths, otherwise they would be trivially satisfied by an inconsistent normative system. In contrast with other specification languages characterized by a normative flavor (e.g. [14, 16, 1]), which assume that the normative system is consistent (i.e., there exists a legal outward transition for every state) either by assuming axiom D [14] or as an explicit hypothesis on the transition system [16, 1], in our approach the absence of contradictory norms represents a desirable property that a rational institution ought to satisfy and that can be verified by our model checker. To exemplify the use of normed temporal operators, we modify Property (P.2) such that if holders respect all norms of the institution and they perform action $accept$, then they will $give$ their resources:

$$\mathbf{AG}\forall act:\sigma_{aid}\forall rec:\sigma_{aid}\forall res:\sigma_{resources}(happens(accept, act, rec, res)\rightarrow \mathbf{AF}^{\exists h:\sigma_{holder}\exists f:\sigma_{sf}(subject(h)=subject(f)\wedge ofStatus(o)=f)} happens(give, act, rec, res)) \quad \text{(P.3)}$$

We can also rewrite property (P.1) to investigate whether norm $h1$ is capable of directing the behavior of holders in such a way that when an agent has requested a good and has received a positive answer, it will eventually become the holder of the good:

$$\mathbf{AG}\forall act:\sigma_{aid}\forall rec:\sigma_{aid}\forall res:\sigma_{resources}((happens(accept, act, rec, res) \rightarrow \mathbf{AF}^{\exists w:h1(w=o)}\exists h: holder(subject(h)=rec\wedge resource(h)=res))) \quad \text{(P.4)}$$

To conclude this section we compare the expressiveness and the flexibility of our approach to the specification languages proposed in [1] and [12]. In [1] the authors proposed *Normative Temporal Logic* (NTL), a language similar to CTL with the exception that operators **A** and **E** are replaced by O_η and P_η, which intuitively can be read as "for all paths compliant with the normative system η" and "there exists a path compliant with the normative system η". Given the semantics provided in [1] and assuming that η represents a set of norms, NTL operators are equivalent to normed temporal operators characterized by a formula φ_η representing all individuals of sorts belonging to η. For instance, formula $\mathrm{O}\Box_\eta\varphi$ of NTL corresponds to $\mathbf{AX}^{\varphi_\eta}\mathbf{AG}^{\varphi_\eta}\varphi$, where φ_η is defined as follows: $\varphi_\eta \equiv \bigwedge_{\sigma_n\in\eta}\exists k:\sigma_n(k=o)$.

In [12] Lomuscio and Sergot presented a modal operator $\mathrm{O}_a\varphi$ which expresses the fact that φ holds over reachable states where agent a complies with its protocol. Assuming that a is an agent, $\mathrm{O}_a\varphi$ is equivalent to $\mathbf{AX}^{\exists f(ofStatus(o)=f\wedge subject(f)=a)}\varphi$. While NTL does not provide any construct to reason about agents, in [12] it is possible to investigate only the compliance of agents with the whole set of norms (described as a protocol): instead, normed temporal operators allow us to reason about subsets of norms and agents, and to express complex interdependencies among them as exemplified by Property (P.3).

6 Sanction Mechanisms

To guarantee that those agents that follow norms are not damaged by those who do not, institutions should provide rules that describe what kind of sanctions are applied when agents violate norms. According to [17], the imposition of status functions constitutes a necessary condition for the application of sanctions, since “with that new status come the appropriate punishment” [17, pag. 50]. Such status functions not only may provide new powers and new obligations (prohibitions), but may also revoke or change existing powers or norms: for instance, the exclusion of an agent from an interaction ruled by an institution (e.g., an auction) means that powers and norms defined by such institution have been revoked. Analogously, officials can apply sanctions only if they have the necessary powers, and certain obligations (prohibitions) may further regulate how such powers ought to be exerted. Therefore, given that sanctions modify the powers and norms of agents, we propose to model sanction mechanisms as rules that impose or revoke status functions when a norm is violated. In our framework sanction mechanisms are defined according to the following grammar:

```
sanction ::= "sanction" symbol "pre" expression ";" "eff" effect
  ("," effect)* ";" ;
precondition ::= expression;
effect ::= (var ("," var)* "(" expression ")" "-X->")?
  var ("assign"|"revoke") "(" term "=" term ("," term"=" term)* ")";
```

For instance, the following sanction mechanism describes that when a norm $h1$ is violated, then the resource is assigned to the agent that has requested the good and powers and obligations associated to status function *requested* are revoked:

```
sanction h1
pre true;
eff r2:$\sigma_{requested}$ revoke (reqRes(r2)=resource(f)),
    r1:$\sigma_{requested}$ res:$\sigma_{resources}$ a:$\sigma_{aid}$(res=resource(f)$\wedge$reqRes(r1)=res
         $\wedge$ a=requester(r1)) -X->
         r2:$\sigma_{holder}$ assign(resource(r2)=res,subject(r2)=a)
```

Sanction mechanisms do not induce any new sort: instead, each of them introduces two predicates, pre_{san_i} and eff_{san_i}, which respectively represent a condition that must be satisfied before a violated obligation activates the i-th sanction mechanism, and its effects. Predicates pre_{san_i} (and analogously predicates eff_{san_i}) are determined by the obligation sort that must be sanctioned (σ_{symbol}) and the status function that defines it (σ_s). Furthermore, predicate pre_{san_i} must satisfy the following axiom schema:

$$\mathbf{AG}\forall o \forall f(pre_{san_i}(o,f) \leftrightarrow \texttt{precondition}_i) \tag{A.7}$$

where $\xi(o) = \sigma_{symbol}$ and $\xi(f) = \sigma_s$. Similarly, each sanction mechanism instantiates the following axiom schema which defines what status functions are imposed or revoked when a sanction mechanism is activated:

$$\mathbf{AG}\forall o \forall f(eff_{san_i}(o,f) \leftrightarrow (\bigwedge_{k=0}^{K_i} \forall \overline{s}_{k_i}(\texttt{expression}_{k_i} \rightarrow \mathbf{X}\exists t_{k_i}([\neg]assigned(t_{k_i}) \wedge \bigwedge_{l=1}^{N_{k_i}} term_{k_i,l,1} = term_{k_i,l,2})))) \tag{A.8}$$

where variables $\overline{s}_{k_i}$ is a set of variables defined by the k-th effect expression of the i-th sanction mechanism and t_{k_i} represents status functions that will be assigned or revoked. Finally, the following axiom schema states that the i-th sanction mechanism brings about its effects when it is activated by the violation of an obligation and its preconditions are met:

$$\mathbf{AG}\forall o \forall f((ofStatus(o) = f \wedge pre_{san_i}(o, f) \wedge \mathbf{X}violated(o)) \rightarrow eff_{san_i}(o, f)) \tag{A.9}$$

Axiom Schema (A.9) suggests that, as institutional events, also sanction mechanisms concur to the definition of predicate $modified$, which ensures that a status is not assigned (revoked) when no institutional event or sanction mechanism affects it (see Section 3). Moreover, Axiom Schema (A.9) describes the main difference among institutional events and sanction mechanisms: while the former happen because other events occur and certain conditions are satisfied (see Axiom (A.2)), the latter are fired only by violations. To some extend, we can interpret Axiom Schema (A.9) as defining a single convention for the activation of any sanction mechanism.

Properties (P.1) and (P.2) can be regarded as two different interpretations of the human norm "when agents accept to give a resource, then requesters ought to become the new holders", where the latter property explicitly refers to the actor and the action that ought to be performed. Norm $h1$ introduced in Section 4 reflects such rule and the introduction of a sanction mechanism for norm $h1$ changes the set of constitutive rules in such a way that Property (P.1) is satisfied by the Resource Management institution. Observing Figure 1, we can notice that the violation of norm $h1$ forces the effects of action $give$, but not the performance of the action itself: therefore, we can expect that Property (P.2) still does not hold, which is confirmed by our model checker. As it has been formulated and unless we introduce a convention such that $accept$ counts as $give$ (which may be incompatible with the rules of a human institution), we think that it is impossible to devise a mechanism to satisfy Property (P.2), since it would mean that we are capable of forcing an autonomous agent to act.

7 Discussion and Conclusions

In this paper we have extended FIEVeL with new constructs to model normative aspects of institutions and we have introduced a flexible specification language to define properties regarding paths that are compliant with norms. We have also exemplified how an institution can be developed by using our approach, verifying that it satisfies certain requirements and modifying its constitutive and regulative rules to comply with them. We have also shown that when properties stem from norms of human institutions that artificial institutions should reflect, it is not always possible to satisfy them, at least under certain interpretations of the human institutions.

In [9] Grossi et al. presented an overview of the role of norms and sanctions in institutions. According to [9] it seems that every norm can be either regimented or enforced, while we think that the viability of such mechanisms depends on the meaning attributed by designers to norms. As we have seen, certain interpretations may exclude the possibility of regimenting them and, generally speaking, regimentation of norms regarding institutional aspects can be achieved only by converting regulative rules into constitutive rules. More precisely, prohibitions can be regimented by revoking powers [6, 7] while obligations can be enforced by changing the interpretation of certain terms. For instance, norm "all yes/not questions should be answered" can be trivially regimented by assuming that silence counts as a positive (negative) answer. Instead, assuming that only a message sent by an agent counts as a communicative act (like in [7]) it is impossible to regiment such norm.

In [6] sanctions are considered only as rules which *create* new obligations (commitments) and powers, while in this paper we have claimed that sanctions may also *delete* obligations and powers by revoking status functions. Moreover, the approach discussed in [6] is based on an intuitive semantics, which does not allow the development of a framework to verify properties guaranteed by institutions. Analogously, the correctness of protocols modelled in terms of institutional concepts by Artikis et al. [2, 15] is only guaranteed by systematic executions. Despite the terminologies used in this paper and in [2] are quite similar, in [2] *physical actions* can be performed only by agents playing a specific role, suggesting that such actions are actually institutional. Furthermore, the formalism used in [2, 15] does not provide any abstraction to describe that every institutional action must be empowered in order to be successfully executed. Instead, the authors have to specify this fact for every single action and for every role.

In [8] a rule language is introduced to model norms and to represent the effects of concurrent events. The author proposed the notion of *enforcing events*, which means that obligatory events are considered as if they were executed even when agents do not perform them. In our opinion, events' enforcement transforms regulative rules into constitutive rules, by defining when time events count as obligatory events, and represents an effective mechanism to describe automatic updates of institutions. In general, we believe that it is not possible to enforce all kinds of events, especially those (like actions) that can only be performed by autonomous agents.

The constructs presented in Section 4 constitute a high-level description of norms, and our tool automatically classifies transitions and states as compliant with each norm of the system. In this respect, our approach is similar to the one presented in [18]. Instead, the input language of the model checker described in [16] requires designers to explicitly list the set of states that each agent may reach, and to classify them as *red* (an agent violates the protocol) or *green*. Although red states are such only because they violate a protocol [12, 16], such classification is not inferred from the protocol but must be manually provided independently from it: therefore designers may introduce discrepancies among the protocol and the classification of states. Similarly, in [1] systems are described with a low-level language which requires to associate a name to each transition, and norms can be defined only by listing under what conditions a set of transitions is considered legal.

In the future we plan to define a translation of axioms stemming from our meta-model and from FIEVeL models into Prolog, providing a single framework for the definition, verification, and monitoring of institutions.

References

1. T. Ågotnes, W. van der Hoek, J. A. Rodríguez-Aguilar, C. Sierra, and M. Wooldridge. On the logic of normative systems. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 1175–1180, 2007.

2. A. Artikis, L. Kamara, J. Pitt, and M. J. Sergot. A Protocol for Resource Sharing in Norm-Governed Ad Hoc Networks. In *Declarative Agent Languages and Technologies II*, volume 3476 of *LNCS*, pages 221–238. Springer, 2005.
3. E. M. Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT Press, 1999.
4. E. A. Emerson and J. Y. Halpern. "Sometimes" and "not never" revisited: on branching versus linear time temporal logic. *Journal of the ACM*, 33(1):151–178, 1986.
5. M. Esteva, J. A. Rodríguez-Aguilar, C. Sierra, P. Garcia, and J. L. Arcos. On the Formal Specification of Electronic Institutions. In *Agent Mediated Electronic Commerce, The European AgentLink Perspective*, pages 126–147. Springer, 2001.
6. N. Fornara and M. Colombetti. Specifying and Enforcing Norms in Artificial Institutions. In *Proceedings of the 4th European Workshop on Multi-Agent Systems*, 2006.
7. N. Fornara, F. Viganò, and M. Colombetti. Agent Communication and Artificial Institutions. *Autonomous Agents and Multi-Agent Systems*, 14(2):121–142, 2007.
8. A. García-Camino. Ignoring, Forcing and Expecting Concurrent Events in Electronic Institutions. In *Proceedings of the AAMAS Workshop on Coordination, Organization, Institutions, and Norms in agent systems*, 2007.
9. D. Grossi, H. Aldewereld, and F. Dignum. Ubi lex, ibi poena: Designing norm enforcement in e-institutions. In *Coordination, Organizations, Institutions, and Norms in Multi-Agent Systems II*, volume 4386 of *LNCS*, pages 110–124. Springer, 2007.
10. A. Jones and M. J. Sergot. On the characterization of law and computer systems: The normative systems perspectives. In *Deontic Logic in Computer Science: Normative Systems Specification*, pages 275–307, 1993.
11. A. Jones and M. J. Sergot. A formal characterisation of institutionalised power. *Journal of the IGPL*, 4(3):429–445, 1996.
12. A. Lomuscio and M. Sergot. A formulation of violation, error recovery, and enforcement in the bit transmission problem. *Journal of Applied Logic*, 1(2):93–116, 2002.
13. M. Manzano. Introduction to many-sorted logic. In *Many-sorted logic and its applications*, pages 3–86. John Wiley & Sons, 1993.
14. J.-J. Meyer and R. J. Wieringa. Deontic Logic: A Concise Overview. In *Deontic Logic in Computer Science: Normative Systems Specification*, pages 3–16. John Wiley and Sons, 1993.
15. J. Pitt, L. Kamara, M. Sergot, and A. Artikis. Formalization of a voting protocol for virtual organizations. In *Proceedings of the 4th Conference on Autonomous agents and Multi-Agent Systems*, pages 373–380, 2005.
16. F. Raimondi and A. Lomuscio. Automatic Verification of Deontic Interpreted Systems by Model Checking via OBDD's. In *Proceedings of the 16th Eureopean Conference on Artificial Intelligence*, pages 53–57, 2004.
17. J. R. Searle. *The construction of social reality*. Free Press, New York, USA, 1995.
18. M. J. Sergot and R. Craven. The Deontic Component of Action Language nC+. In *Deontic Logic and Artificial Normative Systems*, volume 4048 of *LNCS*, pages 222–237. Springer, 2006.
19. F. Viganò and M. Colombetti. Specification and Verification of Institutions through Status Functions. In *Coordination, Organizations, Institutions, and Norms in Multi-Agent Systems II*, volume 4386 of *LNCS*, pages 125–141. Springer, 2007.
20. F. Viganò and M. Colombetti. Symbolic Model Checking of Institutions. In *Proceedings of the 9th International Conference on Electronic Commerce*, 2007. To appear.

Author Index