Supporting Experimentation for Trust Models in Virtual Organizations: TOAST *

Ramón Hermoso, Roberto Centeno, Holger Billhardt, and Sascha Ossowski

Artificial Intelligence Group - DATCCCIA
University Rey Juan Carlos, Madrid (Spain)
{ramon.hermoso, roberto.centeno, holger.billhardt, sascha.ossowski}@urjc.es

Abstract. The problem of selecting adequate acquaintances to interact with is a key problem in open multiagent systems, as agents frequently enter and leave the system, and benevolent and competent behaviour cannot be guaranteed. A broad variety of trust and reputation mechanisms have been developed to address this problem.
In recent years, the concepts of Electronic Institutions and Virtual Organizations have been put forward, that make use of organizational concepts to impose a certain structure on open MAS, while accounting for a high degree of agent autonomy. Elsewhere we have proposed mechanisms that takes advantage of the information contained in such organizational structures to improve the efficiency of trust models. However, there is currently no way to experimentally evaluate such mechanisms in Virtual Organizations. In this paper we present TOAST, an organization-based trust testbed that intends to overcome this problem.

1 Introduction

The concept of *organization* has become a central notion in Multi-Agent Systems (MAS) research. It is widely accepted that organizational structures and properties can provide significant advantages when developing agent software, since they allow building up complex MAS designs using a reduced set of simple abstractions [12]. Such organizational abstractions often endow the system with some sort of structure that shapes agent behavior. Agents that decide to join an organization usually must consider some constraints, such as enacting some *roles* for participating in different *interactions*. Furthermore, these abstractions can be complemented by others of higher level, *e.g.*, sets of *norms* and their mechanisms intending to prevent from unexpected or undesirable behaviors [4]. When MAS are conceived based on these types of organizational structures and their resulting constraints, they are frequently termed Virtual Organizations (VO) [15].

* The present work has been partially funded by the Spanish Ministry of Education and Science under projects TIC2003-08763-C02 and TIN2006-14630-C03-02 (FPI grants program) and by the Spanish project "Agreement Technologies" (CONSOLIDER CSD2007-0022, INGENIO 2010)

The recent trend towards more open and heterogeneous MAS has fostered research in the field of trust and reputation mechanisms. These mechanisms are especially useful for agents involved in an open system when choosing counterparts to interact with. Trust mechanisms usually are based on two different aspects: *i) agent's local past experience* and *ii) opinions from others about third parties* (*exchange of reputation*) [9, 16, 17]. The former provide more reliable information but requires a large amount of previous interactions. The latter gives the agent a faster way to acquire acquaintance information, but it often suffers from inaccuracy [5].

We argue that VOs can benefit considerably from using those kind of trust and reputation mechanisms, since, in spite of structural constraints or normative abstractions that keep agents from performing certain actions, their autonomy still endows agents with the freedom *i) to choose the actions to do next* and *ii) to select with whom to perform them.* However, to back such a claim, it is necessary to count on a testbed capable of simulating and evaluation key characteristics of trust and reputation mechanisms in VOs.

We have developed a new testbed called TOAST (Trust Organizational Agent System Testbed) with this intention in mind. There are other tools for experimenting with trust mechanisms, e.g. the ART-testbed [6], but they do not fit our purposes, since they do not allow for the simulation of organizations, but the evolution of individuals in a shared environment.

In Section 2 previous work that has motivated to build TOAST is briefly explained. Section 3 is focused on the design of the tool and its execution. In Section 4 we analyze which advantages have been obtained from the use of TOAST as a simulation tool. Finally, in Section 5 we will summarize our work and will stress on future lines.

2 Organizational Trust Model

In the last years, trust and reputation mechanisms have become an important research field both in open and heterogeneous systems in general and in multiagent systems in particular. Those mechanisms provide agents with *expectations* about the future behavior of their acquaintances based on their history within the system [9, 14]. Therefore the concept of *trust* is very useful for an agent when it has to choose a counterpart to interact with.

As we have mentioned before, trust mechanisms usually are based on two different dimensions: a dimension based on an agent's local past experience – *confidence* – and another one based on opinions from others about third parties – *reputation.* Both are ratings agents may use in order to evaluate the trustworthiness of other agents in a particular situation.

As we have argued in previous work [8, 7, 2] trust mechanisms may also be important within virtual organizations in which agents act according to some predefines rules and organizational structures. In these contexts, trust models have to be seen not only as a means for agents to achieve their goals the best way they can (*micro level*), but also as a mechanism that can influence and improve

the behavior of the organization as a whole (*macro level*). Additionally, acting within virtual organizations may provide agents with certain elements that can be used to improve their trust-related decision making processes, overcoming shortcomings of classical trust mechanisms. Such elements may be organizational meta-services that provide agents with extra information, for example, about reliable counterparts. But also the organization structure that defines the functioning of the organization may be used by agents to infer the trustworthiness of other agents. In particular, virtual organizations usually define roles agents can play and interactions agents can participate in and often describe those concepts in taxonomies. Such taxonomies allow categorizing roles and interactions and, thus, provide a notion of similarity among them.

In previous work [8, 7, 2] we have introduced a trust model that takes advantage from the existence of taxonomies. This model is based on the assumption that *agents tend to behave similarly when enacting similar roles in similar interactions*. Let a be an agent playing role *Purchaser* that wants to *buy a new laptop*. So, a needs to find a "good" *seller* to buy her new laptop. If a has never bought a laptop before the problem of selecting a "good" *seller* is even more complex. However, it is probably that, among potential *sellers* she knows, there will be some of them who she has previously performed other kind of interaction with, for instance *buy a personal computer*. Based on our assumption, it is reasonable to think that the agent a could infer the behaviour of other agents playing similar roles in similar situations. Thus, a could decide to trust more in other agent that has sold her *a personal computer* successfully rather than in other that sells her *vegetables in the market*.

Using this assumption, an agent is able to assess (to a certain extent) the future behavior of another agent in a certain situation by considering its past behavior in "similar situations". That is, an agent can infer trustworthiness, even if it has, first, no direct past experience about an issue , and, second, it can not collect opinions from other agents either because the opinions from others are unreliable ("liars") or none of the agents has enough proper experience. Hence, the use of such additional information included in the definition of a virtual organization or explicitly provided by it can help to overcome the classical problem of confidence (e.g., lack of experience) and of reputation ("cheating" and "liars"), since the less agents need to ask others the less risk they assume of managing inaccurate information.

The use of trust models does not involve that agents are forced to make counterpart selections based only on trust, but motivations for decision making process could be determined also by other factors, such as *risk*, *dependencies* and so on.

3 TOAST

In this section we present TOAST (*Trust Organizational Agent System Testbed*), the framework we have developed in order to evaluate and compare different

trust models. In contrast to other tools, TOAST has been developed to observe organizations both at a *micro* and a *macro* level.

Trust is an abstract concept that cannot be directly measured. One common way of assessing trust is based on how the actions performed by the trustee affect the truster. Therefore, by measuring the utility of actions derived from the act of *trusting*, we can estimate in a long term how good the trust mechanism is, and compare it to others at the *micro* level. Nevertheless, agents, who are responsible of their own acts, belong to organizations, where the actions take place. Thus, consequences of interactions (even if good) will affect in some way the whole organization (*macro* level). So, TOAST allows the user to observe both the individual evolution of the agents and the global evolution of the organization.

This tool simulates organizations where agents can *join* and *enact* some roles to perform some *interactions* with other members in order to fulfill some individual *goals*. Agents in TOAST are Java-programmed entities with a well-defined selection politics (in our case, it is based on the trust model implemented by the agent) [1]. In Figure 1 we can observe different modules that compose TOAST.

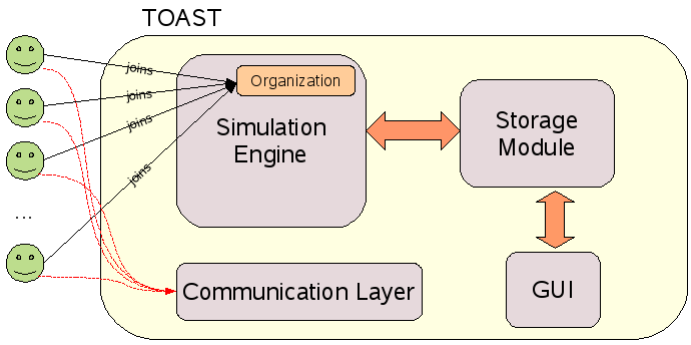

Fig. 1. TOAST modules

The organization, on the one hand, defines *a priori* which types of interactions can be performed, as well as the roles that can be played on them. This information (roles and interactions) will be public and available for any agent that joins the organization. Furthermore, roles and interactions will be structured through taxonomies of concepts. Such taxonomies are relevant for trust models we have been experimenting with [8]. The organizational concepts TOAST relies on are represented in Figure 2.[2]

[1] We considered not to use a multi-agent platform e.g. JADE [1] because this would have made the system too complex. With the same reason, we decided not to use a specific simulation tool (e.g. [3, 11])

[2] For reasons of complexity, we have assumed that every goal can be accomplished through only one interaction.

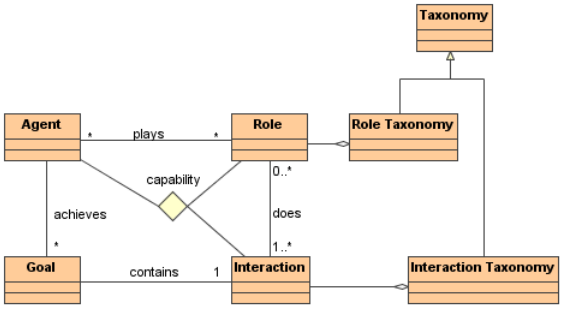

Fig. 2. Organizational elements in TOAST.

In section 3.1 we outline how the organizational elements are instantiated, while section 3.2 describes the dynamics of the simulation of a VO.

3.1 Defining a VO in TOAST

In order to use the framework, a user has to specify the type of organization to simulate, that is, she has to specify the organizational domain. In this regard, the user needs to define the following elements:

- *Roles taxonomy*: defines types of functionalities agents can play in this kind of organizations.
- *Interactions taxonomy*: defines types of interactions that can be performed in this kind of organizations in terms of the roles the agents participating in an interaction have to play.
- *Types of goals*: specifies goals that can be fulfilled by agents within the organization. They are associated to an interaction, which has to be performed in order to accomplish the goal.

All this input information is specified through XML. The following is an excerpt from a XML file defining an interactions taxonomy.

Instance of a VO in TOAST Once the organizational scenario has been defined, the next step is to instantiate a VO for that domain. In this step, user has to specify the number of agents that will participate in the VO, as well as the trust model each agent will use together with its parameters (if any).

As we have mentioned before, agents join an organization enacting one or more roles. In the process of instantiation of a VO the system assigns randomly roles to agents. In this regard, the user has to fix a *dispersion* factor for each role. This factor specifies the percentage of the agents that will play a given role in the organization. This feature becomes very important when experimenting with trust mechanisms, because it determines among how many potential candidates an agent has to choose an appropriate counterpart for its interactions. Thus,

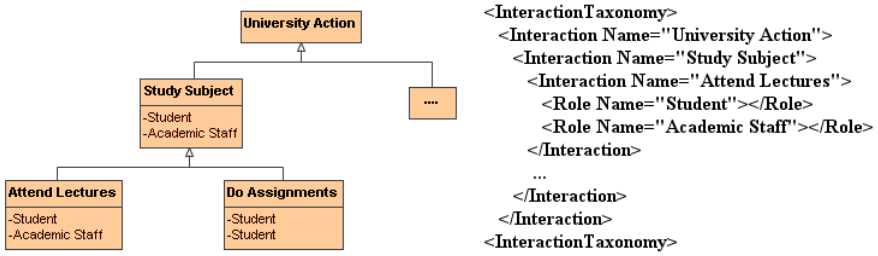

Fig. 3. Interactions taxonomy example file.

the parameter has an influence on the complexity of the agents' decision making processes.

TOAST models agents' behaviors as their *capability* for enacting a specific role in a specific interaction. For that, for each situation $\langle A, R, I\rangle$ ($\langle Agent$, $Role$, $Interaction\rangle$), TOAST generates a capability distribution (a normal probability distribution) that models agent A's capability playing role R in interaction I.

$$Capability_{\langle A,R,I\rangle} \equiv N_{\langle A,R,I\rangle}(\mu, \sigma) \tag{1}$$

where mean μ ($0 \leq \mu \leq 1$) represents the typical (mean) agent behavior for that situation ($\langle R, I\rangle$), and σ ($0 \leq \sigma \leq 0.5$) is the standard deviation representing possible variations from that typical behavior. When instantiating a VO, the system generates the capability distributions (that is, μ and σ) for all possible $\langle Agent, Role, Interaction\rangle$ tuples.

As we have briefly explained in Section 2, trust models focus on past experiences, possibly complemented with inferences an agent can make from organizational structures. In our previous work we have argued that *agents tend to behave in a similar way in similar situations*. In order to take into account this assumption, TOAST generates the capability distributions (μ and σ values) for $\langle A, R, I\rangle$ tuples as follows:

1. First, the simulator generates randomly a capability distribution for every agent ($Capability_{A_i}$) that defines its general behavior (for any kind of situation).
2. Using the agent capability and following role similarities specified in the role taxonomy, a capability distribution is generated for each role the agent can play ($Capability_{(A_i,R_j)}$). In particular, the μ value for $Capability_{(A_i,R_j)}$ is generated as a random value from the capability distribution $Capability_{A_i}$, whereas the standard deviation remains constant. Role similarities, when specified in the role taxonomy, determine the maximum difference that may exist between the mean values (μ) of the capability distributions of the same agent for two different roles. In this regard, μ values are generated until they fulfill the specified restrictions.
3. Finally, TOAST generates the capability distributions $Capability_{A_i,R_j,I_k}$ for all $\langle A_i, R_j, I_k\rangle$ tuples by generating the μ values from the corresponding

agent/role capability distributions $Capability_{(A_i,R_j)}$ (σ values remain constant). In this process, restrictions on interaction similarities are taken into account in the way described in step 2.

After generating the agents' capability distributions TOAST generates a list of goals, which are randomly assigned to the agents in the organization. The number of goals has to be specified by the user.

3.2 TOAST execution flow

In order to explain the dynamics of the simulation of a VO we part from a simple example from a university domain with typical roles as *student*, *teacher*, etc. Let's suppose an agent S participating in the VO and playing role $R_1 \equiv$ *student*:

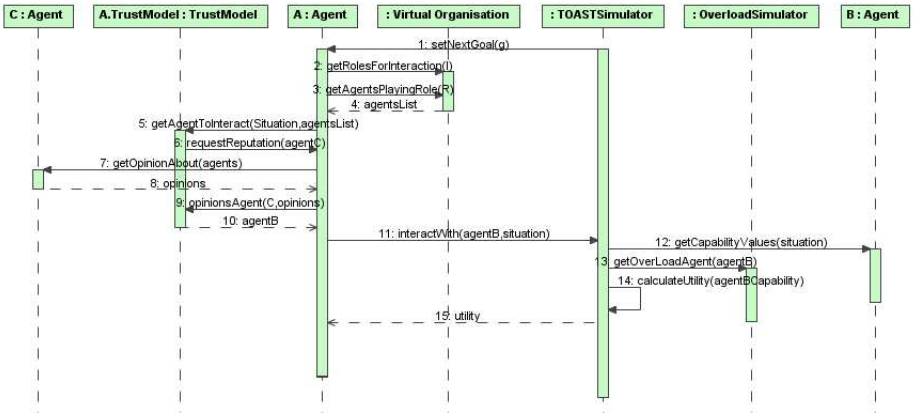

Fig. 4. TOAST execution flow diagram.

- S receives a goal $g \equiv$ *clear up a doubt about AI* at its task queue. S will observe the interaction required to accomplish the goal $I \equiv$ *have a tutor session about AI with a teacher*.
- Agent S knows that in order to perform this interaction and, thus, to fulfill its goal, it needs to find another agent, playing the role $R_2 \equiv$ *teacher*, to interact with.[3] In order to find other agents who play that role, S requests a list of such agents from the VO (information about roles and interactions is publicly available).
- The VO provides S with a list of all agents that can play the role *teacher*.

[3] At the current state, we consider that all agents will interact with other if they are asked for.

– From the list of potential counterparts, and using its trust model, S will choose another agent to perform with it the required interaction. The trust model intends to estimate how trustworthy the interaction with another agent is regarding the fulfillment of a specific goal. Consequently, S will select the agent that is most trustworthy (according to its trust model) in playing role *teacher* in the interaction *"have a tutor session about AI with a teacher"*. In order to select an appropriate counterpart for an interaction, an agent's trust model can use reputation mechanisms, local information about past experiences, or any information provided by the VO:
– At this point S could ask other agents for opinions about third parties (if it is implemented in her trust model). S could use this information in order to better estimate trust for different potential agents which she can interact with.
– Let's suppose agent S has selected agent T for performing the interaction I. The next step consists in simulating the interaction execution. TOAST will simulate that agent S, enacting role R_1 will perform interaction I with agent T, which plays roles R_2.
– As a result of the interaction simulation TOAST returns an utility value to agent S, representing the degree of fulfillment or satisfaction of S when performing the interaction I with T. This value will be generated with the following equation:

$$Utility_{S \rightarrow \langle T,R_2,I \rangle} = randomPM(Capability^*_{\langle T,R_2,I \rangle}) \quad (2)$$

where $Capability^*_{\langle T,R_2,I \rangle} \equiv N_{\langle T,R_2,I \rangle}(\mu^\star, \sigma)$, and *randomPM* is a function that generates a random value from a normal distribution according to the *Polar Method* [10]. T's capability behavior may be influenced by its *workload*. For example, its utility may decrease due to too many requests for participating in interactions with other agents. The mean value $\mu^\star$ includes this fact. It is calculated as follows:

$$\mu^\star = \mu - (\mu \cdot WL^t_T) \quad (3)$$

where $WL^t_T \in [0..1]$ represents a workload factor for agent T at the time instance t. Workload factor is used to test how trust models evolve and can recover from sudden changes in the behavior of others.
– Once TOAST has generated the utility value this is sent to agent S, which may use it to update its trust model.
Following this execution flow, all agents complete all the goals in their task queue. During this process, TOAST stores all valuable information (utility evolution, reputation requests, expectation errors, etc.) making it possible to monitor the evolution of both the agents and the organization.

4 Experimenting with TOAST

The current version of TOAST allows to compare the evolution of VO's in general (in terms of utility), as well as, in particular, the performance of trust models of agents participating within it.

4.1 Prototypical input

TOAST allows the user to perform simulations customizing the following features:

- *Organizational structures.* User can change organizational structure by loading different concepts taxonomies, tuning role spreading factors, introducing new goals, etc.
- *Agents type.* It is also possible to change agents' behavior to simulate.
- *Trust models.* For any simulated agent, user can change her trust model, in order to compare different scenarios.

4.2 TOAST output

This section outlines the representation of overcomes given after a simulation in TOAST.

Utility evolution Utility represents the fulfillment degree from an agent's perspective after performing an interaction with a counterpart within the VO. This degree is related to the quality of service of the counterpart when performing the interaction. The utility value could be seen as an economic value (using the *money* metaphor) obtained by agents as the outcome of interactions they have performed. The utility gains allow for a comparison between agents (the trust models they use) and also between different experiments.

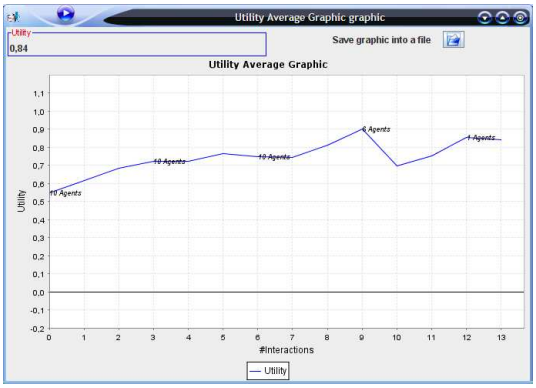

Fig. 5. Global utility graphic screenshot.

TOAST allows the user to observe the utility evolution at a *macro* level (utility obtained by VO[4]) and at a *micro* level (utility obtained by each agent).

[4] Overall utility could be assessed as an utility function implemented by the VO, where other factors (*e.g.* fulfillment degree of organizational purposes) could contribute to the aggregated measure.

Macro level utility can be obtained, for instance, by means of an aggregation of individual agent utilities. This overall "value" can be used as a normalization method in order to compare the behavior of virtual organizations based on the use of different trust models.

At agent level, the tool allows to refine different views in a graphic way: utility evolution for a specific interaction, agent utility (as an aggregation of all the interactions), utility evolution with a specific counterpart and interaction, temporal evolution of confidence, etc. In Figure 5 we can see a graphic provided by TOAST with information about overall utility of the organization.

Reputation management In order to properly evaluate a trust model, the framework has to manage opinion requests if reputation mechanism are involved. An agent's request for an opinion about a third party always implies risk. In open and heterogeneous systems, doing this kind of requests can bring some problems, because there may exist *liars*, *malicious* agents, etc. with the consequent risk of utility loss. Furthermore, reputation requests often raise computational costs due to message ex-change and aggregation of opinions.

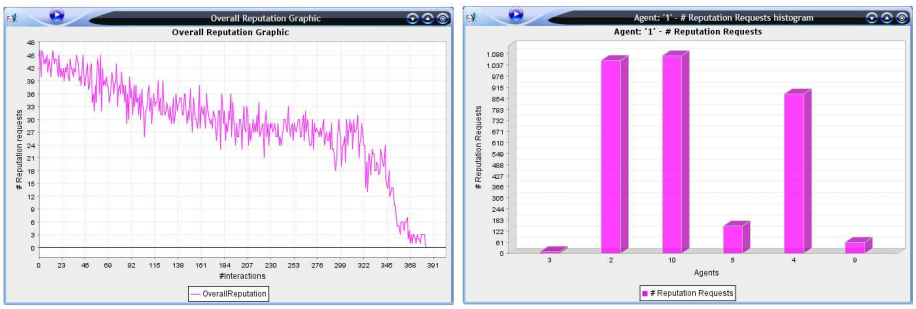

Fig. 6. Evolution of reputation requests over time in the VO and histogram of reputation requests for agent 1.

Hence, although using reputation can help agents to assess trust, there should be a trade-off between reputation and the estimation of trust based on an agent's local information. In this sense, TOAST allows the user to observe how reputation request evolve over time in different models. With this feedback, users can vary agents' behaviors in order to adjust this trade-off. Similarly to utility monitoring, TOAST provides users with graphical diagrams about reputation requests in the overall organization, as well as with requests evolution at the agent and interaction level. Figure 6 shows the evolution of reputation requests over time in the organization. This graphic output is also provided by TOAST.

Agent data In order to deeply examine the behavior of each trust model, TOAST allows viewing the state of agents within the organization. We can ob-

serve agents' *static* details, e.g. the roles they can play, capability distributions, etc. On the other hand, we can also observe *dynamic* values, e.g. transactions between agents (interactions and opinion requests), the state of agents' internal data structures, the other agents they have interacted with, the interaction frequency, etc.

With these data we can monitor each agent within the VO and the VO itself, making it possible to compare VO based on different coordination mechanisms.

5 Conclusions

In this work we describe TOAST, a testbed for simulating agent-based virtual organizations (VOs) and evaluating trust models. The development emerges from the necessity of evaluating different trust models in organizational environments [8] , where not only the agents' individual utility is of interest, but also the evolution of the organization as an entity that has to fulfill its own goals and purposes. TOAST provides users with different views to monitor simulation experiments at a *macro* level (organization) and *micro* level (agents).

The system has been tested with different basic scenarios such as, academic organizations (university example), tourism agency, and also in service-oriented environments [2] . There exist many different trust and reputation models together with their respective test frameworks [13, 18, 17]. Each of them is focused on performing experiments related to the specific domains and scenarios the authors wanted to evaluate. Consequently, a comparison between different models becomes a hard task. In the last years, valuable efforts have been made to build a framework able to compare any kind of trust model and that could be used to experiment with heterogeneous trust and reputation mechanisms. The result of this effort is the ART-testbed [6]. However, ART focuses on closed societies with no complex organizational structures. Furthermore, the competitive environment it involves as well as its design constraints (it is neither open nor scalable), raise the need for new testbeds that allow to analyze trust models and behavior patterns at the level of a group of agents, a virtual organization or an agent society. TOAST is one step into this direction.

Currently we are working on a distributed implementation of the system which will make it possible to carry out experiments with bigger organizations and with more complex organizational structures. Furthermore, we plan to introduce more dynamicity into the system, e.g., agents should be able to join and leave an organization at any time, and several factors may change the behavior of agents over time.

References

1. Fabio L. Bellifemine, Giovanni Caire, and Dominic Greenwood. *Developing Multi-Agent Systems with JADE (Wiley Series in Agent Technology).* Wiley, April 2007.
2. Holger Billhardt, Ramón Hermoso, Sascha Ossowski, and Roberto Centeno. Trust-based service provider selection in open environments. In *22nd Annual ACM*

Symposium on Applied Computing (SAC2007), Seoul, Korea, March 11-15, pages 1375–1380, 2007.
3. N. Collier. Repast: An extensible framework for agent simulation. 2002.
4. Marc Esteva, Bruno Rosell, Juan A. Rodríguez-Aguilar, and Josep Ll. Arcos. AMELI: An agent-based middleware for electronic institutions. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, volume 1, pages 236–243, 2004.
5. Karen K. Fullam and K. Suzanne Barber. Dynamically learning sources of trust information: Experience vs. reputation. In *Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'07)*, pages 1055 – 1062, 2007.
6. Karen K. Fullam, Tomas B. Klos, Guillaume Muller, Jordi Sabater, Andreas Schlosser, Zvi Topol, K. Suzanne Barber, Jeffrey S. Rosenschein, Laurent Vercouter, and Marco Voss. A specification of the agent reputation and trust (art) testbed: experimentation and competition for trust in agent societies. In *AAMAS '05: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pages 512–518, New York, NY, USA, 2005. ACM Press.
7. Ramón Hermoso, Holger Billhardt, Roberto Centeno, and Sascha Ossowski. Effective use of organisational abstractions for confidence models. In Oguz Dikenelli Gregory O'Hare, Michael O'Grady and Alessandro Ricci, editors, *Engineering Societies in the Agents World*, volume 4457 of *LNAI*, pages 368–383, Berlin, In press. Springer.
8. Ramón Hermoso, Holger Billhardt, and Sascha Ossowski. Integrating trust in virtual organisations. In Pablo Noriega and Javier Vázquez-Salceda, editors, *Coordination, Organization, Institutions and Norms in Agent Systems II*, volume 4386 of *LNAI*, pages 17–29, Berlin, In press. Springer.
9. T. Dong Huynh, Nicholas R. Jennings, and Nigel R. Shadbolt. FIRE: An integrated trust and reputation model for open multi-agent systems. In *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI)*, 2004.
10. Donald E. Knuth. *The Art of Computer Programming - Seminumerical Algorithms*, volume 2. Addison-Wesly, 1998.
11. N. Minar, R. Burkhart, C. Langton, and M. Askenazi. The swarm simulation system, a toolkit for building multi-agent simulations, 1996.
12. Andrea Omicini and Sascha Ossowski. Objective versus subjective coordination in the engineering of agent systems. In Matthias Klusch, Sonia Bergamaschi, Peter Edwards, and Paolo Petta, editors, *Intelligent Information Agents: An AgentLink Perspective*, volume 2586 of *LNAI: State-of-the-Art Survey*, pages 179–202. Springer-Verlag, March 2003.
13. Jordi Sabater and Carles Sierra. Regret: a reputation model for gregarious societies. In *4th Workshop on Deception, Fraud and Trust in Agent Societies*, pages 61–69, 2001.
14. Jordi Sabater and Carles Sierra. Reputation and social network analysis in multiagent systems. In *AAMAS '02: Proceedings of the first international joint conference on Autonomous agents and multiagent systems*, pages 475–482, New York, NY, USA, 2002. ACM Press.
15. M. Schumacher and S. Ossowski. The governing environment. In Weyns, Parunak, and Michel, editors, *Environments for Multiagent Systems II*, volume 3830, pages 88–104. Springer-Verlag, 2006.
16. W. T. Luke Teacy, Jigar Patel, Nicholas R. Jennings, and Michael Luck. Travos: Trust and reputation in the context of inaccurate information sources. *Autonomous Agents and Multi-Agent Systems*, 12(2):183–198, 2006.

17. Bin Yu and Munindar P. Singh. An evidential model of distributed reputation management. In *AAMAS '02: Proceedings of the first international joint conference on Autonomous agents and multiagent systems*, pages 294–301, New York, NY, USA, 2002. ACM Press.
18. Giorgos Zacharia, Alexandros Moukas, and Pattie Maes. Collaborative reputation mechanisms in electronic marketplaces. In *HICSS*, 1999.