TRUMMAR - A Trust Model for Mobile Agent Systems Based on Reputation

Ghada Derbas Ayman Kayssi Hassan Artail Ali Chehab
Department of Electrical and Computer Engineering
American University of Beirut
P. O. Box 11-0236, Beirut 1107-2020, Lebanon
{gwd00, ayman, hartail, chehab}@aub.edu.lb

Abstract

In this paper we present TRUMMAR, a reputation-based trust model that mobile agent systems can use to protect agents from malicious hosts. TRUMMAR is unique in being a truly comprehensive model since it accounts, in a unified framework, for a multitude of concepts such as prior-derived reputation, first impression, loss of reputation information with time, hierarchy of host systems (neighbors, friends, and strangers), and the inclusion of interaction results in reputation calculation. TRUMMAR is also general enough to be applied to any distributed system. We show simulation results that verify the correctness of this model and the effects of its various parameters.

1. Introduction

Security is an important issue in networks and distributed systems. In order to interact with an entity on a network, it is important to investigate the entity's trustworthiness and reliability to ensure the correctness of the entity's responses. The ideal situation to address this issue would be to operate in a trusted environment – a trusted LAN, a trusted internet, etc. Thus, evaluating the trust of entities is a vital step towards avoiding security problems.

In mobile agent technology, releasing an agent into a network involves a risk of exposing the agent to attacks. Thus, it is of great importance to the agent source that the environment – comprised of hosts and other agents – is secure enough for the agent to move. The approach we propose in this paper involves inquiring about the reputation of the host to which the agent will be moving. This reputation depends on many factors such as a previously-calculated reputation of the destination with respect to the agent source, reported reputations from neighboring hosts under the same administrative control, reported reputations from friends under different, but trusted administrative control, and reported reputations from other hosts that wish to volunteer information. The proposed model, which we will refer to as TRUMMAR (TRUst Model for Mobile Agents based on Reputation), also takes into consideration the duration of time since last interaction, host cooperation in exchanging reputation information, first impression of the host concerning the destination host, the host position in its society/network and the host's sociability.

The rest of the paper is organized as follows. In Section 2 a survey of previous work on trust and reputation in agent technology is presented. Section 3 presents the proposed computational model for reputation and trust, and Section 4 discusses the results of experiments and simulations carried out using this proposed model. Section 5 provides some conclusions.

2. Related Work on Trust and Reputation

Trust and reputation have gained importance in diverse fields such as economics, evolutionary biology, distributed artificial intelligence, grid computing, agent technology, among others. Various definitions for trust and reputation have evolved as a result; in what follows, we review the work done on trust and reputation in the specific field of agent technology.

Mui et al. propose in [1] a computational model based on reciprocity (a mutual exchange of deeds, either in favor or revenge), trust (a subjective expectation an agent has about another's future behavior based on the history of their encounters) and reputation (a perception that an agent creates through past actions about its intentions and norms). Mui et al. also propose in [2] an intuitive typology summarizing different notions of reputation that have been studied across diverse disciplines. The typology divides reputation into several components: individual reputation which includes direct and indirect reputation, and group reputation. Direct reputation includes interaction-derived reputation and observed reputation, whereas indirect reputation includes prior-

0-7803-8577-2/04/$20.00 ©2004 IEEE

derived reputation, group-derived reputation and propagated reputation. The relative strengths of these different notions of reputation were studied in a set of evolutionary games.

Singh et al. [9] encourage the consideration of Granovetter's observation that a person whose social contacts were in the same cluster is worse off than someone who has social contacts in different clusters. This is because someone with widely distributed contacts has access to a larger variety of information and to far more opportunities.

Abdul-Rahman et al. [3] study reputation as a form of social control in the context of trust propagation — reputation is used to influence agents to cooperate for fear of gaining bad reputation. Although not explicitly described, they have considered reputation as a propagated notion which is passed to other agents by means of "word-of-mouth".

Pujol et al. [4] propose a method of calculating reputation based on the position of each member of a community within the corresponding social network. A new algorithm, NodeRanking, was developed to obtain such measures. This algorithm assesses reputation by using local information only.

Cubaleska and Schneider [8] propose a method for a posteriori identification of malicious hosts to build a trust policy. Depending on how much the source host trusts the other hosts, it can either define an appropriate order in which selected hosts should be visited, or it can decide which hosts it does not want to contact again.

Barber and Kim [11] propose a computational model that combines belief revision and trust reasoning processes and show how deceptive or incompetent agents can be isolated from an agent's decision making process with this model. An agent learns reputations of other agents using dissimilarity measures calculated from the previous belief revision processes (Direct Trust Revision) and/or communicated trust information that contains reputations (Recommended Trust Revision). Agents utilize this model to detect fraudulent information and to identify potential deceptive agents as a form of social control in which an individual member is responsible for taking care of security.

In another work, Barber, et al. [10] discuss the difficulty of assigning initial reputations when either the truster or the trustee is new to the system. They state that some interaction must take place for recommendation–based reputations to build up, and that when initial reputation assignments are arbitrary or have default values, a stable base for reputation has not yet been reached. Since information collection takes time, agents have to evaluate the time available for decision-making against the increased reputation base stability gained as more reputation processing is performed. Furthermore, Barber and Kim [12] define trust as the agent's confidence in the ability and intention of an information source to deliver correct information. Reputation on the other hand, is defined as the amount of trust an information source has created for itself through interactions with other agents. They also propose a multi-agent belief revision algorithm that utilizes knowledge about the reliability or trustworthiness of information sources.

Jurca and Faltings [13] state that the most reliable reputation information can be derived from an agent's own experience. However, much more data becomes available when reputation information is shared within an agent community. They attempt to encourage agents to truthfully share reputation information by providing incentives (a side-payment scheme) for recommenders to tell the truth. They point out that it is not in the best interest of an agent to truthfully report reputation information because reporting reputation information provides a competitive advantage to others, and that by reporting positive ratings an agent slightly decreases its own reputation with respect to the average of other agents.

Tran and Cohen [14] propose a reputation-oriented reinforcement learning algorithm for buying agents in electronic market environments, taking into account the fact that the quality of a good offered by different selling agents may not be the same and that a selling agent may alter the quality of its goods. Modeling the reputation of sellers allows buying agents to focus on those sellers with whom a certain degree of trust has been established. The authors also include the ability for buying agents to explore the marketplace in order to discover new reputable sellers.

Finally, we mention the work done by Azzedin et al. [5, 6, 7] in the field of trust and reputation in grid computing systems. They present a formal definition of behavior trust and reputation and discuss a behavior trust management architecture that models the process of evolving and managing behavior in grid computing systems.

3. The Computational Model

In this section we present TRUMMAR as a general model for the calculation of reputation values and the determination of trust decisions. Consider the situation where a Host X wants to send a mobile agent to another Host Y in order for the agent to accomplish a certain task. Host X will not send the agent unless it is sure that Host Y is trustworthy, i.e. that Host Y will provide a trusted environment which will not alter, corrupt, manipulate, delete, or misinterpret the agent's code, data, or status. In order to find out whether Host

Y is trustworthy, Host X calculates a reputation value for Host Y, as a combined result of previous reputation information calculated and stored by Host X, and inquiries about Host Y's reputation from neighbors of Host X, friends of Host X, and other hosts willing to volunteer reputation information concerning Host Y. Figure 1 illustrates the hierarchy of trust with respect to information providers considered in this model.

The first step in making judgments is by trusting one's own information, i.e. using previous information available at Host X about the reputation of the destination, Host Y. Then Host X goes further out to trusting other hosts on its own network/that are under the same administrative control (neighbors). These hosts are assumed to be as vulnerable to an attack as Host X on the network. Friends – hosts from different networks that are under different, but trusted administrative control – follow neighbors in this hierarchy of trust. Finally, stranger hosts that are willing to volunteer information come in last in this hierarchy of trust, as shown in Figure 1.

We first define the term *interaction* as a process which involves an agent source sending its agent to a desired destination to accomplish a certain task, and the degree of success in accomplishing this task. In TRUMMAR, The reputation value is then calculated as follows:

$$repY/X(0) = A\,repY/X + B\frac{\sum_i \alpha_i repY/X_i}{\sum_i \alpha_i} + C\frac{\sum_j \beta_j repY/X_j}{\sum_j \beta_j} + D\frac{\sum_l \delta_l repY/Z_l}{\sum_L \delta_l} \quad (1)$$

where

- $repY/X(0)$ represents the value that is being calculated *now* for the reputation of Y at X, since as we will see later, the reputation values change with time.
- $repY/X$ represents the last calculated reputation of Y with respect to X, modified to account for the time interval since the last time that Host X was interested in finding Host Y's reputation.
- $\sum_i \alpha_i repY/X_i$: This represents the weighted sum of reputations of Y as reported by the neighbors of X (X_i).
- $\sum_j \beta_j repY/X_j$: This represents the weighted sum of reputations of Y as reported by the friends of X (X_j).
- $\sum_l \delta_l repY/Z_l$: This represents the weighted sum of reputations of Y as reported by strangers (Z_l) in the host space that volunteer to provide information about the reputation of Y.
- $\alpha_i, \beta_j, \delta_l$ are weighing factors which depend on the reputation of the individual neighbors, friends, and strangers in the host space, respectively. These factors are functions of the last calculated reputation of the specific hosts. They may also be functions of parameters such as *cooperation*, which depends on the ratio of successful interactions to total requests, *sociability*, which indicates the number of hosts communicated/communicating with, and *rank* which indicates the importance of a specific host with respect to other hosts in its network.
- A, B, C, and D are weighing factors for the respective reputation of Y with respect to neighbors of X, reputation of Y with respect to friends of X, and reputation of Y with respect to strangers in the agent space. These factors are empirically-determined constants which should satisfy the constraint $A > B > C > D$.

Reputation values are restricted to values between 0 and k, where k is a pre-defined constant, such that $0 \le repY/X \le k$. This means that if all reputations used in calculating $repY/X$ have values between 0 and k, $repY/X$ will also have a value between 0 and k. To achieve this condition, the constant coefficients A, B, C, and D, in (1) should satisfy the constraint $A + B + C + D = 1$.

Figure 1. Hierarchy of trust

As time passes by, a host reputation with respect to other hosts changes to an unknown state if little or no interaction occurs between them. Thus, in TRUMMAR, we lose reputation information with

time, if no interactions are taking place. When a host Z receives a request (from Host X) for reputation information about Host Y, it modifies its reputation information and sends that result to the requesting host. This modified value is used in (1). The change with time is exponential according to $e^{-\frac{(t-t_0)}{\tau}}$ where t_0 is the last time Z computed the reputation of Y and τ is an empirical constant that determines how quickly or slowly reputation information becomes invalid. It is important to have the reputation of a host (whether good or bad) converge to a neutral value as time passes by and no interactions take place. Thus, a bad host does not remain bad for life, and a good host is not considered good for ever. To achieve this, the reputation values are modified with time using:

$$repY./Z = final\_value + (initial\_value - final\_value)e^{-\frac{(t-t_0)}{\tau}} \quad (2)$$

where the initial value is the reputation value at $t = t_0$, and the final value is the neutral value explained above.

The step that follows the calculation of the reputation of a certain host is to determine trust, by associating to the host the label "trustworthy" or "untrustworthy". This can be determined by introducing two threshold values θ and φ, referred to as the absolute trust and absolute mistrust thresholds, respectively. Three cases are considered:

- If $repY/X \geq \theta \Rightarrow$ Y can be trusted
- If $repY/X \leq \varphi \Rightarrow$ Y cannot be trusted
- If $\varphi < repY/X < \theta \Rightarrow$ Y can be considered as either trustworthy or untrustworthy depending on how paranoid or trusting Host X is. Figure 2 shows how paranoid hosts can project the reputation value down towards the curve labeled "paranoid host", and may as a result trust hosts for which the projected value is greater than a certain threshold, e.g. $0.5(\theta + \phi)$. For trusting hosts, the reputation value is projected upwards towards the "trusting host" curve. A probabilistic approach is also possible where the decision to trust or not to trust is biased by how close the reputation value is to θ or φ, respectively.

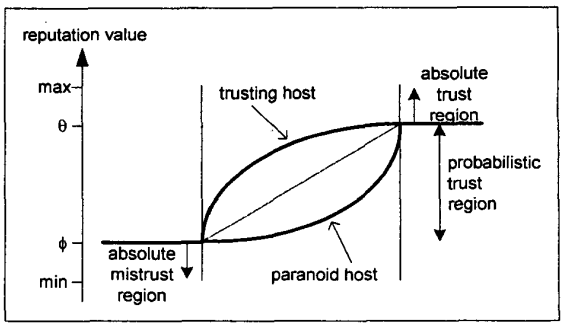

Figure 2. Trust thresholds and curves

If Host X decides that Host Y is trustworthy it will send the mobile agent to Y in order to accomplish the specified task. When the interaction is over, X recalculates the reputation of Y using:

$$repY/X(0) = \xi \times repY/X(0) + (1-\xi) \times RI \quad (6)$$

where *RI* is the result of the interaction as perceived by X. ξ is typically 0.1 – 0.3, which gives more weight to the results of the interaction that just took place, and less weight to previous reputation. Note that the result of interaction *RI* is a number in the same range as reputation values.

A final point worth mentioning is that, no matter how large the number of hosts in the system gets, we can limit the number of hosts contacted for reputation information by taking a portion of the total. If the number of hosts is below a defined threshold, we request information from all hosts in the system. However, if it exceeds this threshold, we can select a percentage of the hosts in the system (e.g. 10 %). This process can be taken a step further by defining an upper limit to the number of hosts, thus controlling the number of requests being issued in case the system grows to a large size.

3.1 Special Cases

Prior to calculating the reputation of Host Y, it is of relevance to know whether Y is a new host to the system, is a neighbor of X, is a friend of X, or is none of the above.

Case 1: Host Y is new to the system: In the case where Host Y is a new host that has just joined the system and which, consequently, has not yet interacted with any other hosts, X interacts with Y according to a pre-defined *first impression* value that X uses, which may either be dull enough such that X refuses to interact with Y (X is a paranoid host) or that may be courageous enough for X to take a risk at interacting

with a stranger that has no previous history (X is a trusting host). It is in this case that interaction-based reputation cannot be avoided, due to lack of previous reputation information about the host. From what has been stated, the reputation of Y is assumed to be:

$$repY/X(0) = first\_impression \quad (3)$$

This aspect of TRUMMAR is consistent with Barber's statement [10] that interaction-based models must assume an initial default reputation which may result in unfair losses to the truster or trustee. However, these losses may be reduced by subjecting new hosts to a preliminary test period during which knowledge acquired through direct interaction is considered unreliable until the reputation information is stabilized.

Case 2: Host Y is a neighbor of Host X: If Y is a neighbor of X, it is sufficient to ask other hosts in the neighborhood about the reputation of Y. Thus, Equation (1) reduces to:

$$repY/X(0) = ArepY/X + B\frac{\sum_i \alpha_i repY/X_i}{\sum_i \alpha_i} \quad (4)$$

where $A + B = 1$.

Case 3: Host Y is a friend of Host X: If Y is a friend of X, then it is sufficient to ask neighbors of X and friends of X about the reputation of Y. Thus, Equation (1) reduces to:

$$repY/X(0) = ArepY/X + B\frac{\sum_i \alpha_i repY/X_i}{\sum_i \alpha_i} + C\frac{\sum_j \beta_j repY/X_j}{\sum_j \beta_j} \quad (5)$$

where $A + B + C = 1$.

4. Experimental Verification

In order to illustrate TRUMMAR, we consider the system shown in Figure 3. Each host in this system will have a table containing all necessary coefficients, constants and reputation values. This table is updated as a result of interactions, and reputations are modified with time when reported to other hosts. A sample table (that of Host X) is shown in Table 1.

To verify TRUMMAR, we implemented the model in software using the C++ programming language, and used this implementation to simulate the system shown in Figure 3.

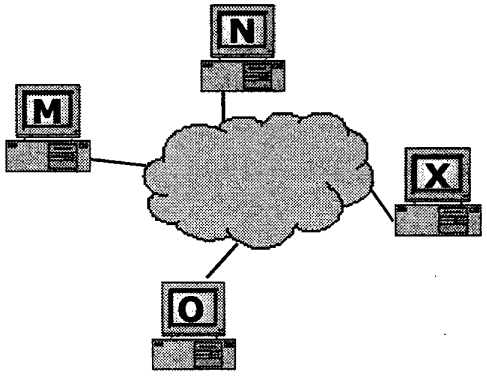

Figure 3. Example system

Table 1. Constants, coefficients and reputations at X

Field	Value	Explanation
Rep M/X	$rep\ (t - t_M)$	Reputation value of M at X
Rep N/X	$rep\ (t - t_N)$	Reputation value of N at X
Rep O/X	$rep\ (t - t_O)$	Reputation value of O at X
FR	FR_X	Final reputation value to which reputations converge
A	A_X	A and B are weighting factors; A>B and A+B=1
B	B_X	
α_M	α_M	Weighting factors of information sent by Hosts M, N and O to Host X
α_N	α_N	
α_O	α_O	
θ	θ_X	Absolute trust threshold
φ	φ_X	Absolute mistrust threshold
First Impression	FI_X	First impression given to a stranger by Host X
ξ	ξ_X	Previous reputation parameter
τ	τ_X	Time constant

The system consists of four hosts X, M, N, and O, any two of which will randomly interact at random time intervals. Only one interaction takes place at any time. Reputation takes values between 0 and $k = 5$. Hosts M, N and O are assumed to be good hosts. This is indicated by a high result of interaction (RI) that is always between θ and k. In the simulation RI is a random number between 4 and 5, and θ is 4. Host X is assumed to be a bad/malicious host. This is indicated by a low RI that is randomly generated between 0 and φ. The value of φ in the simulation is 1. The constants and coefficients for the TRUMMAR model are shown in Table 2 and are the same for all hosts. If a host is found "untrustworthy", its corresponding α in (1) is

reassigned a value of 0.1, otherwise it is assigned the value of 0.5. Note that *FI*, the first impression value is set midway between minimum and maximum reputation, at $k/2$.

Table 2. Values for constants and coefficients used in the simulation

Field	Value
A	0.55
B	0.45
θ	4
φ	1
FI	2.5
ξ	0.2
τ	1000

For values of reputation that are between 1 and 4, the decision in the simulation to trust the host or not is taken probabilistically.

We use Equation (4) in the simulation, since all hosts are neighbors. The simulation is run for 4000 cycles. During the first 1000 cycles, the interaction level was high, i.e. hosts often interact and communicate reputation information, while for the remaining 3000 cycles, the interaction level was very low.

The reputation of good hosts with respect to other hosts increases gradually from an initial value of 2.5 (the first impression value) to a value around 4.5, when interactions are taking place (below 1000 cycles). For the case of little or no interactions (after 1000 cycles), the host's reputation decays exponentially as shown in Figure 4.

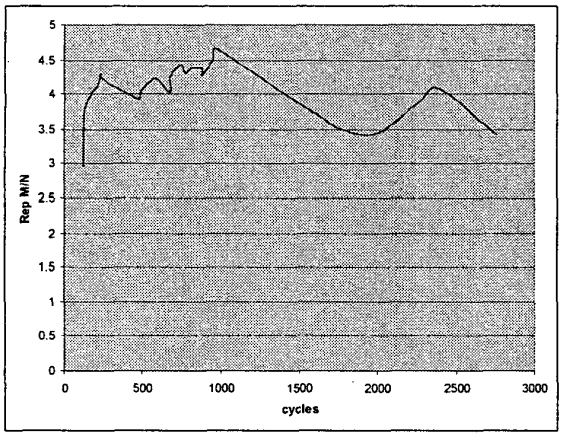

Figure 4. Behavior of a good host's reputation (τ = 1000 cycles)

We can see from Figure 4 that as the level of interaction diminishes after 1000 cycles, the reputation value drops below 4, which is the absolute trust threshold. This is due to the choice of the time constant τ. In fact, τ plays an important role in determining how short or how long the reputation remains stable. For large values of τ, the reputation values change more slowly with time compared to smaller values of τ. This can be clearly seen by comparing Figure 5 to Figure 4. The value of τ in Figure 5 was increased ten times to 10,000 cycles.

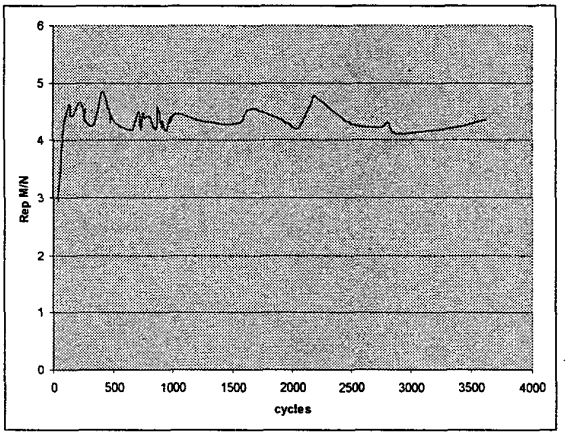

Figure 5. Behavior of a good host's reputation (τ = 10,000 cycles)

As for the reputation of malicious hosts with respect to others, this drops from an initial value of 2.5 to a minimum value less than 1, when interactions are taking place (below 1000 cycles). For the case where little or no interactions (after 1000 cycles) are occurring, a malicious host's reputation increases towards 2.5 as shown in Figure 6. The state of being "not trustworthy" is not a permanent state for a host, according to (2). When the value of τ is increased to 10,000 cycles, the same effect that was observed for good hosts, namely the stability of the reputation values, is also observed for bad hosts, as shown in Figure 7.

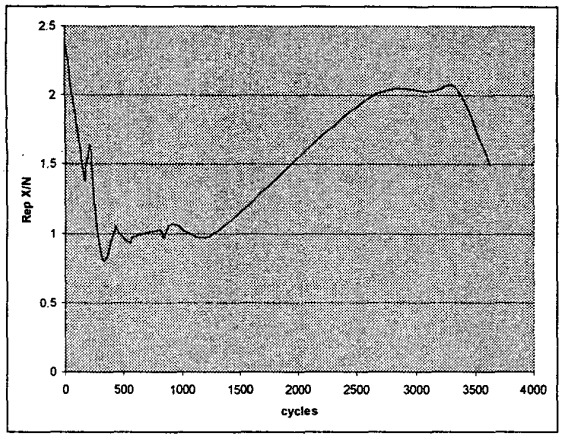

Figure 6. Behavior of a bad host's reputation (τ = 1000 cycles)

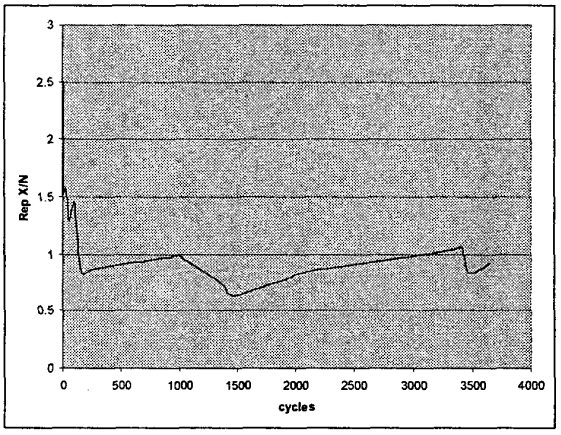

Figure 7. Behavior of a bad host's reputation (τ = 10,000 cycles)

Varying the values of the other parameters such as FI, ξ, ϕ, θ, and/or the coefficients A and B will result in different system dynamics. However, in all cases, we reach a trust decision that is consistent with the expected behavior of TRUMMAR.

5. Conclusions

TRUMMAR is a reputation-based trust model that mobile agent systems can use to protect agents from malicious hosts. Even though numerous trust models have been previously proposed in the literature, TRUMMAR is unique in being truly comprehensive since it accounts, in a unified framework, for a multitude of concepts such as prior-derived reputation, first impression, loss of reputation information with time, hierarchy of host systems (neighbors, friends, and strangers), and the inclusion of interaction results in reputation calculation. In this respect, TRUMMAR is also general enough to be applied to any distributed infrastructure, whether a mobile agent system, a computational grid system, or otherwise.

While TRUMMAR is ready to accept the concepts of sociability, rank, and cooperation, we are currently in the process of implementing such concepts in the model and its simulation environment.

In addition to simulation, we are also working on the implementation of TRUMMAR using Sun's Java Web Services. This platform supports heterogeneity through Java APIs and XML communication. The initial implementation has been successfully deployed on a network with forty hosts, with future plans to increase the number of hosts to hundreds.

6. Acknowledgements

The authors would like to thank Mohammad Eid and Samer Hannah for their help in reviewing this paper. This work was supported by the University Research Board of the American University of Beirut.

7. References

[1] L. Mui, M. Mohtashemi, and A. Halberstadt, "A Computational Model of Trust and Reputation", *Proceedings of the 35th Hawaii International Conference on System Sciences*, Big Island, Hawaii, January 2002.

[2] L. Mui, A. Halberstadt & M. Mohtashemi, "Evaluating Reputation in Multi-agents Systems", *Trust, Reputation, and Security: Theories and Practices*, Springer-Verlag, Berlin, 2003.

[3] A. Abdul-Rahman and S. Hailes, "Supporting Trust in Virtual Communities", *Proceedings of the 33rd Hawaii International Conference on System Sciences*, Maui, Hawaii, January 2000.

[4] J. M. Pujol, R. Sangüesa, and J. Delgado, "Extracting Reputation in Multi Agent Systems by Means of Social Network Topology", *Proceedings of the First International Joint Conference on Autonomous Agents and MultiAgent Systems*, Bologna, Italy, July 2002.

[5] F. Azzedin and M. Maheswaran, "Evolving and Managing Trust in Grid Computing Systems", *IEEE Canadian Conference on Electrical & Computer Engineering (CCECE '02)*, Manitoba, Canada, May 2002.

[6] F. Azzedin and M. Maheswaran, "Towards Trust-Aware Resource Management in Grid Computing Systems", *2nd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID2002)*, Berlin, Germany, May 2002.

[7] F. Azzedin and M. Maheswaran, "Integrating Trust into Grid Resource Management Systems", *International Conference on Parallel Processing (ICPP 2002)*, Vancouver, Canada, August 2002.

[8] B.Cubaleska and M.Scheider, "Applying Trust Policies for Protecting Mobile Agents Against DoS", *Third International Workshop on Policies for Distributed Systems and Networks (POLICY)*, Moterey, California, June 2002.

[9] M. Venkatraman, B. Yu and M. P. Singh, "Trust and Reputation Management in a Small-World Network", *Proceedings of the Fourth International Conference on MultiAgent Systems*, Boston, Massachusetts, July 2000.

[10] R. Falcone, K. S. Barber, L. Korba, and M. Singh, "Challenges for Trust, Fraud, and Deception Research in Multi-agent Systems", *Trust, Reputation, and Security: Theories and Practice*, Lecture Notes in Artificial Intelligence: Springer, 2003, 8-14.

[11] K. S. Barber and J. Kim, "Soft Security: Isolating Unreliable Agents", *Proceedings of the AAMAS 2002 Workshop on Deception, Fraud and Trust in Agent Societies,* Bologna, Italy, July 2002.

[12] K. S. Barber and J. Kim, "Belief Revision Process Based on Trust: Agents Evaluating Reputation of Information Sources", *Workshop on Trust in Cyber-societies*, Barcelona, Spain, June 2000.

[13] R. Jurca and B. Faltings, "Towards Incentive-Compatible Reputation Management", *Proceedings of the AAMAS 2002 Workshop on Deception, Fraud and Trust in Agent Societies*, Bologna, Italy, July 2002.

[14] T. Tran and R. Cohen, "Modeling Reputation in Agent-Based Marketplaces to Improve the Performance of Buying Agents", *Proceedings of the Ninth International Conference on User Modelling (UM-03)*, PA, USA, June 2003.