A Security Framework for Agent-based Systems

Jamal Bentahar
Concordia Institute for Information Systems Engineering, Concordia University, Canada
Francesca Toni
Department of Computing, Imperial College London, UK
John-Jules Ch. Meyer
Department of Information & Computer Science, Utrecht University, The Netherlands
and **Jihad Labban**
Concordia Institute for Information Systems Engineering, Concordia University, Canada

Accepted: August 2007

Abstract

Purpose *– This paper aims to address some security issues in open systems such as service-oriented applications and grid computing. It proposes a security framework for these systems taking a trust viewpoint. The objective is to equip the entities in these systems with mechanisms allowing them to decide about trusting or not each other before starting transactions.*

Design/methodology/approach *– In this paper, the entities of open systems (web services, virtual organizations, etc.) are designed as software autonomous agents equipped with advanced communication and reasoning capabilities. Agents interact with one another by communicating using public dialogue game-based protocols and strategies on how to use these protocols. These strategies are private to individual agents, and are defined in terms of dialogue games with conditions. Agents use their reasoning capabilities to evaluate these conditions and deploy their strategies. Agents compute the trust they have in other agents, represented as a subjective quantitative value, using direct and indirect interaction histories with these other agents and the notion of social networks.*

Findings *– The paper finds that trust is subject to many parameters such as the number of interactions between agents, the size of the social network, and the timely relevance of information. Combining these parameters provides a comprehensive trust model. The proposed framework is proved to be computationally efficient and simulations show that it can be used to detect malicious entities.*

Originality/value *– The paper proposes different protocols and strategies for trust computation and different parameters to consider when computing this trust. It proposes an efficient algorithm for this computation and a prototype simulating it.*

Keywords – Trust, Multi-Agent Systems, Agent Communication, Dialogue Games
Paper type – Research paper

I. Introduction

Recent years have seen an increasing interest in web services, service-oriented architectures and grid computing, as well as applications of these technologies e.g. for e-science (science that is enabled by the use of distributed computing resources by end-user scientists) and e-business [12], [22], [23]. Security is a fundamental issue for these emerging technologies and applications, due to their open, highly distributed and large scale nature and their need to share services and resources and allow mutual access to them.

In this paper, we propose a trust and agent-based approach to render these systems secure. The idea is to establish a framework allowing entities (web services, virtual organizations, etc.) to evaluate how much trust they have in one another. These entities are abstracted as autonomous agents able to interact with one another by communicating. Inter-agent communication is regulated by protocols (shared amongst agents and thus public) and determined by strategies (internal to agents and thus private). The paper addresses the following questions: 1) What information can agents use when evaluating the trust they have in other agents? 2) Which protocols and strategies can agents use to support this trust evaluation? 3) How can trust be propagated through the system?

Centralized approaches to security fail to adequately address the e-computing challenges posed by open systems. They are mostly based upon mechanisms such as digital certificates, and thus are particularly vulnerable to “attacks”. This is because if some authorities who are implicitly trusted are compromised, then there is no other check available in the system. Instead, in the decentralized approach we propose in this paper, any “intruders” may only cause limited harm before being detected. Indeed, in our approach, agents trust other agents for more reasons than a single certificate, as they interact and reason about trust values using their internal reasoning.

Recently, some decentralized trust models have been proposed (see [19] for a survey). However, these models are purely quantitative and consider agents just as objects interacting by message exchange, without reasoning capabilities.

However, in multi-agent systems, agents reason using their knowledge bases before making decisions, and can thus engage in flexible interactions. In addition, some of these existing models do not consider the case where false information is collected. This paper aims at overcoming these limitations. Some agent-based approaches to security exist in the literature, notably the one proposed in [22], but these focus on authentication and authorization issues whereas we focus on trust evaluation and propagation through a social network.

The remainder of this paper is organized as follows. In Section II, we present the theoretical background, in particular introducing the notions of communication protocol and strategies and sketching the reasoning capabilities of agents. In Section III, we introduce the notion of *direct trust* and show how agents' direct experiences are used to compute this. Section IV focuses on the propagation of trust through a social network. The algorithmic description and computational complexity of such a propagation are stressed. In Section V, we describe and discuss implementation issues. In Section VI, we compare our framework to related work. Section VII concludes the paper.

II. Background

Through this paper, we consider agent-based systems as societies of autonomous interacting agents. These agents could represent different entities depending on the application: web services, virtual organizations, etc. Agents interact with each other using advanced communication techniques based on dialogue game protocols [9], [15], [17]. Dialogue games are interactions between players (agents), in which each player moves by performing utterances according to a pre-defined set of rules. Let us define these notions of protocol and dialogue games.

***Definition** 1 (**Protocol**):* A protocol Pr is a pair $\langle \mathcal{C}, \mathcal{D} \rangle$ with $\mathcal{C}$ a finite set of allowed communicative acts and $\mathcal{D}$ a set of dialogue games.

We will assume that *communicative acts* in $\mathcal{C}$ may be of c different types ($c > 0$) and we will denote by $CA_i(Ag_1, Ag_2, p, t)$ a communicative act of type i performed by some agent Ag_1 and addressed to some other agent Ag_2 at time t about content p. $Inform(Mary, John, Malicious(Bob), 10)$ is an example of a communicative act (of type $Inform$) by which $Mary$ informs $John$ at time 10 that she believes that a certain agent Bob is malicious.

Intuitively, a dialogue game in $\mathcal{D}$ is a rule indicating the possible communicative acts an agent could perform when he receives a communicative act from some other agent. This is specified formally as follows:

***Definition** 2 (**Dialogue Game**):* A dialogue game Dg is either of the form:

$$CA_i(Ag_1, Ag_2, p, t) \Rightarrow \bigvee_{0<j\leq n_i} CA_j(Ag_2, Ag_1, p_j, t_j)$$

where CA_i, CA_j are in $\mathcal{C}$, $t < t_j$ and n_i is the number of allowed communicative acts Ag_2 could perform after receiving a communicative act of type i from Ag_1;
or of the form

$$\Rightarrow \bigvee_{0<j\leq n} CA_j(Ag_1, Ag_2, p_j, t_0)$$

where CA_j are in $\mathcal{C}$, t_0 is some initial time, and n is the number of allowed communicative acts Ag_1 could perform initially.

According to the definition of dialogue game, a dialogue game is in general non-deterministic, in that, for example, given an incoming communicative act of type i, the receiving agent needs to choose amongst n_i possible replies. In order to render agents deterministic, we introduce the conditions within dialogue games, each associated with a single reply.

***Definition** 3 (**Dialogue Game with Conditions**):* A dialogue game with conditions DgC is a conjunction of rules, specified either as follows:

$$\bigwedge_{0<j\leq n_i} \big(CA_i(Ag_1, Ag_2, p, t) \wedge C_j \Rightarrow CA_j(Ag_2, Ag_1, p_j, t_j)\big)$$

where $t < t_j$ and n_i is the number of allowed communicative acts Ag_2 could perform after receiving a communicative act of type i from Ag_1;
or specified as follows:

$$\bigwedge_{0<j\leq n} \big(C_j \Rightarrow CA_j(Ag_1, Ag_2, p_j, t_0)\big)$$

where t_0 is the initial time and n is the number of allowed communicative acts Ag_1 could perform initially.

In order to guarantee determinism, the conditions C_j need to be mutually exclusive [21]. Dialogue games with conditions reflect agents' private strategies. How conditions C_j are represented and evaluated is private to each agent. For example, agents may use argumentation to evaluate these conditions [4], [7] and the reasoning capabilities of agents are defined using argumentation. Argumentation can be abstractly defined as a dialectical process for the interaction of different arguments for and against some conclusion [10], [8], [3]. Formally:

***Definition** 4 (**Abstract Argumentation Framework**):* Let Arg be a finite set of elements, representing arguments. An abstract argumentation framework is a pair $\langle Arg, \mathcal{AT} \rangle$, where $\mathcal{AT} \subseteq Arg \times Arg$ is a binary relation over Arg. For two arguments a and b, $\mathcal{AT}(a, b)$ means that a is an attack against b. $\sim \mathcal{AT}(a, b)$ means that a does not attack b.

For example, an argument may be defined as a deduction from a given set of rules of a conclusion, or as a pair (H, h) where h is a sentence in some given language and H a subset of a given knowledge base such that (i) H logically entails

h, (ii) H is consistent and (iii) there is no subset of H with properties (i) and (ii).

Because there may be conflicts between arguments, we need to define what an acceptable argument is. The following is a possible definition (corresponding to the definition of "admissible" set of arguments in [10]):

***Definition** 5 (**Acceptability**):* Let $\langle Arg, \mathcal{AT} \rangle$ be an abstract argumentation framework. A set $S \subseteq Arg$ of arguments is said acceptable iff:
$\forall\ a, b \in S\ \sim \mathcal{AT}(a, b)$ and
$\forall\ a \in Arg$ if $\mathcal{AT}(a, b)$ for some $b \in S$ then
$\exists\ c \in S\ : \mathcal{AT}(c, a)$.

In other words, a *set of arguments* is acceptable iff it does not contain any conflicts and it can counter-attack every attack. Then, a *single argument* may be deemed acceptable, e.g., if it belongs to some acceptable set of arguments or to all maximally acceptable set of arguments [11].

Argumentation is especially useful when only incomplete and/or inconsistent information is available. This is typically the case for our agents, that inhabit an open and dynamic society. Agents can use argumentation to evaluate the conditions in their strategies and thus decide how to react to incoming communicative acts. In particular, each condition C_j can be seen as an argument from the agent's argumentation system supporting the communicative act to be performed.

The idea here is that agents use their reasoning argumentation abilities in order to decide about the next communicative act to perform [6], [18]. For example, if an agent $Mary$ informs an agent $John$ that a proposition P is true, $John$ may accept the proposition if he can build an acceptable argument for P from his knowledge base, he may refuse P if he can build an argument for $\neg P$ and may otherwise challenge P.

III. Direct Trust

We adopt a probabilistic-based approach to compute trust values [5]. We define an agent's trust in other agents as a probability function as follows:

***Definition** 6 (**Trust Function**):* Let $\mathcal{A}$ be a set of agents, and $\mathcal{D}$ be a set of domains or topics. The trust function Tr associates two agents from $\mathcal{A}$ and a domain from $\mathcal{D}$ with a trust value between 0 and 1:

$$Tr : \mathcal{A} \times \mathcal{A} \times \mathcal{D} \longrightarrow [0, 1]$$

Given some concrete agents Ag_a and Ag_b in $\mathcal{A}$ and some concrete domain D, $Tr(Ag_a, Ag_b, D)$ stands for "the trust value associated to Ag_b in domain D by agent Ag_a".

To simplify the notation, in the remainder we will omit the domain from all the formulas. Given agents Ag_a and Ag_b in $\mathcal{A}$, we will represent $Tr(Ag_a, Ag_b)$ in short as $Tr_{Ag_a}^{Ag_b}$.

In this section we consider the case where agents in the system know each other because they had a prior interaction history and can thus compute the trust value of all agents (and thus the Tr function) *directly*. We will assume that agents are equipped with means to evaluate the outcomes of their interactions, e.g. again using an argumentation system. Let us assume that they can evaluate their interactions as "*good*" or "*bad*". A good interaction could be one after which the agent is satisfied because his goal that prompted the interaction is achieved after the interaction (successful outcome). A bad interaction could be the opposite (unsuccessful outcome). The trust value given by Tr can be assessed as the ratio of the "*number of successful outcomes*" to the "*total number of possible outcomes*". Formally, let $NG_{Ag_a}^{Ag_b}$ be the number of good interactions that Ag_a had with Ag_b, and $TN_{Ag_a}^{Ag_b}$ be the total number of interactions between Ag_a and Ag_b. We can define:

$$Tr_{Ag_a}^{Ag_b} = \frac{NG_{Ag_a}^{Ag_b}}{TN_{Ag_a}^{Ag_b}} \tag{1}$$

Because agents are equipped with sophisticated reasoning capabilities, they could evaluate the outcomes of their interactions using more flexible values such as "*very good*", "*good*", "*fair*", "*bad*", and "*very bad*". In the general case, they could evaluate their interactions according to a scale of n types numbered from 1 (the most successful interaction) to n (the less successful interaction), such that the first m interaction types ($m < n$) are successful (for example of type "*very good*", "*good*", and "*fair*"). Let $NI_{i\,Ag_a}^{Ag_b}$ be the number of interactions of type i that Ag_a had with Ag_b. Then Tr can be computed by Equation 2 below, which is a refinement of Equation 1:

$$Tr_{Ag_a}^{Ag_b} = \frac{\sum_{i=1}^{m} w_i NI_{i\,Ag_a}^{Ag_b}}{\sum_{i=1}^{n} w_i NI_{i\,Ag_a}^{Ag_b}} \tag{2}$$

where w_i is the weight associated to the interaction type i.

Agents could use several strategies when weighting the interaction types. For example, to minimize the risk of dealing with untrustworthy agents, the weight of "*very bad*" interactions could be higher than the one of "*very good*" interactions. This means that unsuccessful interactions are more valuable when assessing the agents' trust, and agents should perform well and avoid bad behavior in order to get a better trust value. However, less demanding agents could give the same weight to all interaction types, or give more weight to the "*very good*" and "*good*" interactions.

IV. Indirect Trust

Agents can evaluate directly the trust value of agents they have interacted with extensively. However, if the number of interactions with some agent is low (e.g. because the agents has only recently joined the system), agents are not able to compute their trust value directly, but may need to rely upon information provided to them by other agents (that may have interacted more extensively with the given agent). As proposed in [1], [24], [25], we will assume that each agent has two kinds of beliefs when evaluating the trust of another agent: *local beliefs* and *total beliefs*. Local beliefs are based on the direct interactions among agents. Total beliefs are based on the combination of the different testimonies of other agents that we call *witnesses*. In our model, local beliefs

are given by Equation 2. Total beliefs require studying how different probability measures offered by witnesses can be combined. We deal with this aspect below, by considering two mechanisms, each associated with its own protocols, strategies, and computation of the Tr function. The protocols and strategies allow agents to gather information about an unknown, or not well known, agent from witnesses. They differ as to whether these witnesses are directly known to and trusted by the requesting agents or not.

A. Protocol 1: Trustworthy Agents

Let us suppose that agent Ag_a wants to evaluate the trust of an unknown, or not well known, agent Ag_b. Ag_a then asks agents he knows to be *trustworthy* in judging other agents and Ag_b in particular. We will denote these trustworthy agents (the witnesses) as $Ag_i (i = 1 \dots k)$. Before asking a witness Ag_i about the trustworthiness of the third party Ag_b, Ag_a should check that there is no conflict (of interest of other type) between Ag_i and Ag_b. This is represented by the following strategy:

$$Trustworthy(Ag_a, Ag_i) \wedge NoConflict(Ag_i, Ag_b) \Rightarrow \\ Req\_Inf(Ag_a, Ag_i, Trust(Ag_b), t_0)$$

where $Trustworthy(Ag_a, Ag_i)$ means Ag_i is an agent that Ag_a considers trustworthy, and $Req\_Inf(Ag_a, Ag_i, Trust(Ag_b), t_0)$ is the communicative act by means of which initially Ag_a sends to Ag_i a request for information related to Ag_b's trust. The form of this information and the decision as to whether an agent is to be deemed trustworthy will be discussed later.

When Ag_i receives the Req_Inf communicative act, he uses the following dialogue game (protocol) to reply:

$$Req\_Inf(Ag_a, Ag_i, Trust(Ag_b), t) \Rightarrow \\ Tr\_Inf(Ag_i, Ag_a, Inf(Ag_b), t') \\ \vee \; Tr\_NotHave(Ag_i, Ag_a, *, t') \\ \vee \; Tr\_Ref(Ag_i, Ag_a, *, t')$$

$Tr\_Inf(Ag_i, Ag_a, Inf(Ag_b), t')$ is the communicative act Ag_i uses to provide Ag_a with the relevant information regarding Ag_b's trust. $Tr\_NotHave(Ag_i, Ag_a, *, t')$ means that Ag_i does not have the relevant information about Ag_b's trust (the content of this communicative act is empty, represented by $*$). $Tr\_Ref(Ag_i, Ag_a, *, t')$ is the reply Ag_i uses when he refuses to share information about Ag_b's trust with Ag_a (the content of this communicative act is also empty). The protocol that includes this dialogue game as well as the dialogue game:

$$\Rightarrow Req\_Inf(Ag_a, Ag_i, Trust(Ag_b), t_0)$$

is called *Trustworthy Agents Protocol* and is illustrated by Fig. 1. Here, we adopt a graphical representation of protocols, by means of finite state automata, where communicative acts performed initially label edges from double circles, the reply to communicative acts labelling incoming edges into single circles label outgoing edges from these circles, and no reply is foreseen by the protocol to communicative acts labelling incoming arcs into thick single circles.

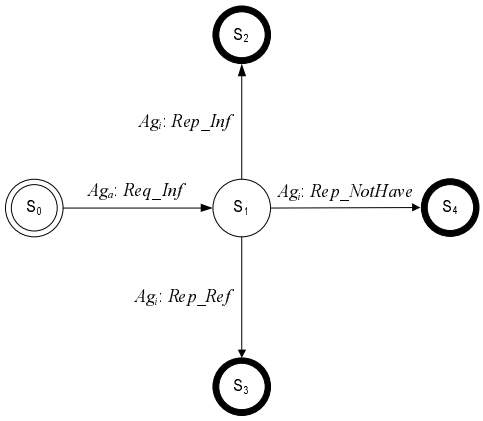

Fig. 1. Protocol 1: Trustworthy Agents

How to select one of the communicative acts allowed in S_1 is based on the argumentative reasoning capabilities of Ag_i, as discussed in Section II. Each of these three choices is associated with a condition $C_{Req\_Inf_j}$ $(j = 1 \dots 3)$ in the strategy below:

$$Req\_Inf(Ag_a, Ag_i, Trust(Ag_b), t) \wedge C_{Req\_Inf_1} \Rightarrow \\ Tr\_Inf(Ag_i, Ag_a, Inf(Ag_b), t') \\ \wedge \; Req\_Inf(Ag_a, Ag_i, Trust(Ag_b), t) \wedge C_{Req\_Inf_2} \Rightarrow \\ Tr\_NotHave(Ag_i, Ag_a, *, t') \\ \wedge \; Req\_Inf(Ag_a, Ag_i, Trust(Ag_b), t) \wedge C_{Req\_Inf_3} \Rightarrow \\ Tr\_Ref(Ag_i, Ag_a, *, t')$$

These conditions could be defined as follows:

$$C_{Req\_Inf_1} : Have(Ag_i, Inf(Ag_b)) \\ \wedge \; Share(Ag_i, Ag_a, Inf(Ag_b)) \\ C_{Req\_Inf_2} : NotHave(Ag_i, Inf(Ag_b)) \\ C_{Req\_Inf_3} : Have(Ag_i, Inf(Ag_b)) \\ \wedge \; \neg Share(Ag_i, Ag_a, Inf(Ag_b))$$

$Have(Ag_i, Inf(Ag_b))$ means that Ag_i has information related to Ag_b's trust, and $Share(Ag_i, Ag_a, Inf(Ag_b))$ means that Ag_i can share these information with Ag_a. The idea here is that if Ag_i has the asked information and he can build an acceptable argument supporting $Share(Ag_i, Ag_a, Inf(Ag_b))$, he will perform the Tr_Inf communicative act. If not, he will perform the Tr_Ref communicative act. If no information is available about Ag_b's trust, he will perform the $Tr\_NotHave$ communicative act. Here, we assume that the Ag_i agents use their local beliefs to assess their trust value in Ag_b as illustrated in Equation 2. This value is given by $Inf(Ag_b)$ ($Inf(Ag_b)$ also gives additional information, as we will discuss later).

B. Trust Evaluation in Protocol 1

Ag_a attributes a trust measure $Tr_{Ag_a}^{Ag_i}$ to each of the agents $Ag_i (i = 1 \dots k)$ he considers *trustworthy*. In general, when

an (*evaluator*) agent assesses the trustworthiness of another (*evaluated*) agent, the former may consider the latter either trustworthy or untrustworthy depending on the trust measure he assigns to this evaluated agent and some *threshold* fixed by the evaluator. The trust measure can be computed using Equation 2. We will define Ag_i trustworthy by Ag_a when the trust measure $Tr_{Ag_a}^{Ag_i}$, given by Equation 2, is greater than a threshold α_a fixed by Ag_a.

We assume that trustworthy agents $Ag_i (i = 1 \dots k', k' \leq k)$ also use Equation 2 to assess the trust value of the agents they know, and in particular Ag_b. Thus, the problem consists in Ag_a evaluating Ag_b's trust measure combining the trust values transmitted by trustworthy agents to Ag_a. Once this value is computed, Ag_a decides to consider Ag_b trustworthy or not depending again on the threshold α_a. In the remainder, for simplicity we will assume that $k' = k$.

Overall, this is a probabilistic problem, and to solve it, we should investigate the distribution of the probabilistic variable X representing the trustworthiness of Ag_b. Because, as we discussed earlier, X may take only one of the two values: 0 (the agent is not trustworthy) or 1 (otherwise), the variable X follows a *Bernoulli distribution* $\beta(1, p)$. Consequently: $E(X) = p$ where $E(X)$ is the expectation of the variable X and p is the probability that the agent is trustworthy. Here, p is the probability we need to compute and it is enough to evaluate the expectation $E(X)$ to find $Tr_{Ag_a}^{Ag_b}$. However, this expectation is a theoretical mean that we must estimate. To this end, we can use the *Central Limit Theorem (CLT) and the law of large numbers*. The CLT states that whenever a sample of size s $(X_1, \dots, X_s)$ is taken from any distribution with mean μ, then the sample mean $(X_1 + \dots + X_s)/n$ will be approximately normally distributed with mean μ. As an application of this theorem, the arithmetic mean (average) $(X_1 + \dots + X_s)/s$ approaches a normal distribution of mean μ, the expectation and standard deviation $\sigma/\sqrt{s}$. Generally, and according to *the law of large numbers*, the expectation can be estimated by the weighted arithmetic mean.

Our probabilistic variable X is the weighted average of k independent variables X_i that correspond to Ag_b's trust according to the point of view of trustworthy agents Ag_i. These variables follow the same *Bernoulli distribution*. They are also independent because in the general case, the probability that Ag_b is trustworthy according to an agent Ag_t is independent of the probability that this agent (Ag_b) is trustworthy according to another agent Ag_r (even though sometimes the two agents Ag_t and Ag_r can have the same opinion about Ag_b's trust because considering the same sequence of events, these events are still independent in the general case). Consequently, the variable X could be approximated by a normal distribution whose average is the weighted average of the expectations of the independent variables X_i. The estimation of expectation $E(X)$ can be given by the following equation:

$$M_0 = \frac{\sum_{i=1}^{k} \left(Tr_{Ag_a}^{Ag_i} \times Tr_{Ag_i}^{Ag_b} \right)}{\sum_{i=1}^{k} Tr_{Ag_a}^{Ag_i}} \tag{3}$$

The value M_0 represents a first estimation of $Tr_{Ag_a}^{Ag_b}$. This estimation, however, does not take into account the *number of interactions* between the trustworthy agents and Ag_b. This number is an important factor because it allows promoting information coming from agents that are more knowledgeable about Ag_b. Another factor might be used to reflect the *timely relevance* of transmitted information. This is because the agent's environment is dynamic and may change quickly. The idea is to promote recent information and to deal with out-of-date information with less emphasis.

The timely relevance could be represented as a coefficient when computing the agent's trust. We use the following function to estimate this factor:

$$TR(\Delta t_{Ag_i}^{Ag_b}) = N(\Delta t_{Ag_i}^{Ag_b})\ e^{-\lambda\ ln(\Delta t_{Ag_i}^{Ag_b})} \tag{4}$$

The parameter Δt is the time difference between the current time and the time at which Ag_i updates his information about Ag_b's trust. The parameter $N(\Delta t_{Ag_i}^{Ag_b})$ is the number of interactions between Ag_i and Ag_b during the time difference Δt. λ is an application-dependent coefficient. The intuition behind this formula is to use a function decreasing with the time difference. Consequently, the more recent the information, the higher is the timely relevance coefficient. The logarithm function is used for computational reasons when dealing with large numbers. Intuitively, this function, which is similar to the well known *reliability function for systems engineering* ($R(t) = e^{-\lambda t}$), reflects the reliability of the transmitted information.

The number of interactions during Δt ($N(\Delta t_{Ag_i}^{Ag_b})$) is an important trust parameter, because in order to get accurate information, it is not only important to evaluate how recent the information is, but also how much recent information the agents use. Indeed, a trust function should consider not only the total number of interactions, but also the number of recent interactions. Fig. 2. illustrates the behavior of the function $TR(\Delta t_{Ag_i}^{Ag_b})$ depending on the number of interactions when $\lambda = -0.5$.

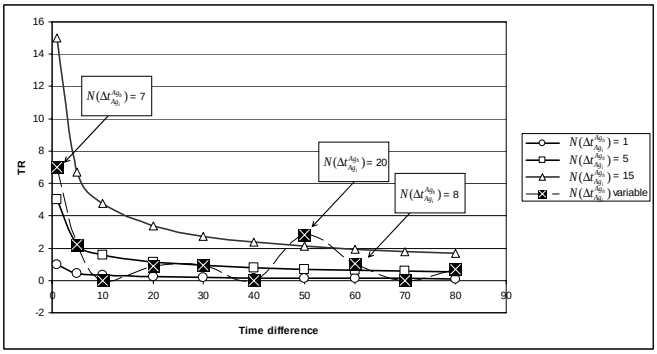

Fig. 2. Timely relevance behavior with interaction numbers.

This figure shows that the timely relevance is more important when the number of interactions during Δt is large. For example the difference between recent and out-of-date information is more important when the number of interactions during Δt is 15 compared to the case where $N(\Delta t_{Ag_i}^{Ag_b}) = 1$. We note that there is no significant difference between recent and out-of-date data when the number of interactions during Δt is ignored ($N(\Delta t_{Ag_i}^{Ag_b}) = 1$). When this number is variable

(variable curve), the figure shows that the timely relevance factor of old interactions could be more important than the one of recent interactions if the number of old interactions is much bigger than the one of recent interactions. However, what is important when $N(\Delta t^{Ag_b}_{Ag_i})$ is variable, is that a small number of recent interactions (the case of $N(\Delta t^{Ag_b}_{Ag_i}) = 7$ in the figure) are more important than a large number of out-of-date interactions (the case $N(\Delta t^{Ag_b}_{Ag_i}) = 20$ in the figure). This issue is made clear in Fig. 3. This figure illustrates two trustworthy agents Ag_1 and Ag_2 who interacted with Ag_b recently (1 unit of time ago), and 80 units of time ago. The total number of interactions between these two agents and Ag_b is the same (20 recent interactions + 2 old interactions for Ag_1 and 2 recent interactions + 20 old interactions for Ag_2). Although the total number of interactions is the same, the difference D in the timely relevance coefficient between these two trustworthy agents is more important in the recent time ($D = 20-2 = 18$) compared to the old time ($D = 2.23 - 0.22 = 2.01$). In other words, the difference in terms of timely relevance between 20 and 2 recent interactions is much more important than the difference between the same number of interactions in the old time. The idea ultimately is that what is important is not only the total number of interactions, but also the number of recent interactions.

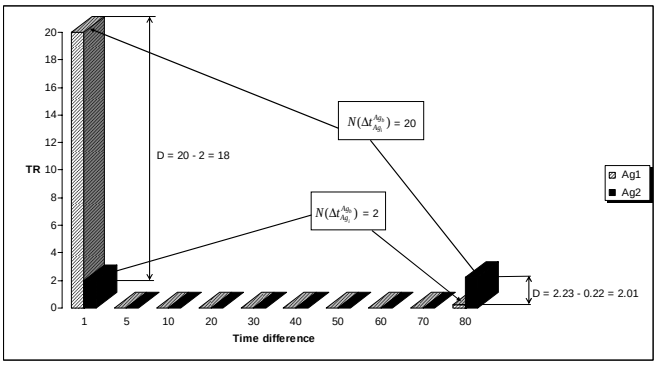

Fig. 3. Difference between recent and out-of-date interactions.

Fig. 4 illustrates the role of the factor λ in the behavior of the timely relevance function. Time is more relevant when λ is close to -1. However, if λ is close to 0, the difference between recent and out-of-date data is not significant. Indeed, how to select this factor and whether to consider or not the number of interactions $N(\Delta t^{Ag_b}_{Ag_i})$ depend on the agent policy. To promote recent data, λ should be high and $N(\Delta t^{Ag_b}_{Ag_i})$ should be considered. If data are not time sensitive, λ should be small and the number of recent interactions could be ignored ($N(\Delta t^{Ag_b}_{Ag_i}) = 1$).

Equation 5 gives a richer estimation of $Tr^{Ag_b}_{Ag_a}$ by taking into account the factors discussed above: (1) the trust of trustworthy agents according to the point of view of Ag_a ($Tr^{Ag_i}_{Ag_a}$); (2) the number of interactions between these trustworthy agents and Ag_b ($TN^{Ag_i}_{Ag_b}$); (3) the timely relevance of information transmitted by trustworthy agents ($TR(\Delta t^{Ag_b}_{Ag_i})$); and (4) Ag_b's trust according to the trustworthy agents ($Tr^{Ag_b}_{Ag_i}$), as communicated

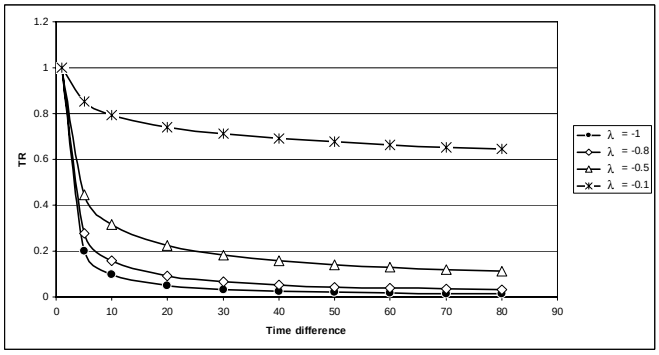

Fig. 4. Timely relevance behavior.

by Ag_i to Ag_a following the strategies previously indicated.

$$Tr^{Ag_b}_{Ag_a}(\Delta t) = \frac{\sum_{i=1}^{n}\left(Tr^{Ag_i}_{Ag_a} \times TN^{Ag_i}_{Ag_b} \times TR(\Delta t^{Ag_b}_{Ag_i}) \times Tr^{Ag_b}_{Ag_i}\right)}{\sum_{i=1}^{n}\left(Tr^{Ag_i}_{Ag_a} \times TN^{Ag_i}_{Ag_b} \times TR(\Delta t^{Ag_b}_{Ag_i})\right)} \quad (5)$$

This Equation shows how trust can be obtained by merging the trust values transmitted by trustworthy agents. This merging method takes into account the proportional relevance of each trust value, rather than treating them equally. To compute this trust, the relevant information a trustworthy agent Ag_i should provide to Ag_a (i.e. the content of Tr_Inf) are: 1) the total number of interactions Ag_i had with Ag_b ($TN^{Ag_i}_{Ag_b}$); 2) the number and time of recent interactions between them ($N(\Delta t^{Ag_b}_{Ag_i})$ and $\Delta t^{Ag_b}_{Ag_i}$); 3) and the overall Ag_i's evaluation of Ag_b's trust ($Tr^{Ag_b}_{Ag_i}$).

C. Protocol 2: Social Network

In the previous section, we developed a *Trustworthy Agents* Protocol and Strategy that allows agents (Ag_a) to ask others (Ag_i) information about the trust value of a third party (Ag_b). If these agents themselves do not have information about Ag_b, they reply by performing the communicative act $Tr\_NotHave$. In this section, we render the earlier mechanism more sophisticate by allowing trustworthy agents to provide additional agents serving as referees who may know the third party. These referees could also suggest other referees in turn if they do not know this third party either. The resulting protocol and strategy, called *Social Network* Protocol and Strategy extend the *Trustworthy Agents* Protocol and Strategy by using the following communicative acts: Req_Refer by which Ag_a asks for referees, and Tr_Refer by which Ag_i provides a set of referees with the relevant information about their trust as discussed in the previous section. Two new dialogue games are included in the *Social Network Protocol*. They are specified as follows:

$DG1$

$$Tr\_NotHave(Ag_i, Ag_a, *, t) \Rightarrow$$
$$DoNothing(Ag_a)$$
$$\vee \; Req\_Refer(Ag_a, Ag_i, Refer(Ag_b), t')$$

$DoNothing(Ag_a)$ is a special act that means Ag_a will do

nothing. This is because the achieved state could be final.

$DG2$

$$Req\_Refer(Ag_a, Ag_i, Refer(Ag_b), t) \Rightarrow$$
$$Tr\_Refer(Ag_i, Ag_a, Inf\_Refer(Ag_b), t')$$
$$\vee\ Tr\_NotHave(Ag_i, Ag_a, *, t')$$
$$\vee\ Tr\_Ref(Ag_i, Ag_a, *, t')$$

According to $DG1$, after receiving $Tr\_NotHave$, Ag_a can do nothing, or ask the agent who sends this communicative act to provide a set of referees. $DG2$ indicates that after receiving this request, Ag_i can provide a set of referees with the relevant information about their trust (the total number of interactions Ag_i had with each referee, the number and time of recent interactions between them, and the overall Ag_i's evaluation of the trust of each referee), or reply by indicating that he does not have this type of information ($Tr\_NotHave$), or reply by refusing to give his list of referees (Tr_Ref). The whole protocol is illustrated by Fig. 5.

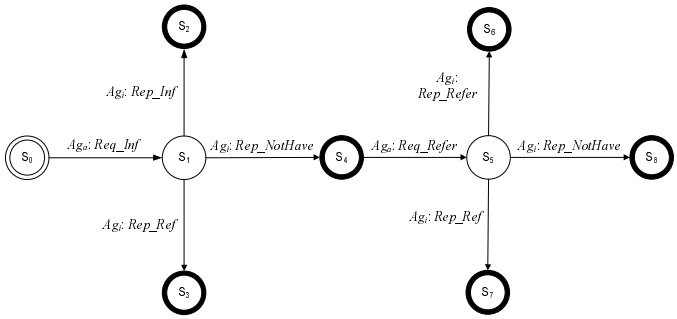

Fig. 5. Protocol 2: Social Network.

As for the *Trustworthy Agent Protocol*, Ag_a may use his argumentation system to select one of the two choices specified in $DG1$. Each choice is associated with a strategy ($C_{Tr\_NotHave_j}$ ($j \in \{1, 2\}$)). An example of these conditions are:

$$C_{Tr\_NotHave_1} : ReceivedEnough(Ag_a, Inf(Ag_b))$$
$$C_{Tr\_NotHave_2} : \neg ReceivedEnough(Ag_a, Inf(Ag_b))$$

Informally, if Ag_a has received enough relevant information about Ag_b's trust, he will not ask for any more referees, otherwise, he will. The three conditions for the strategy associated to $DG2$ may be identical to the ones associated to the dialogue game of the *Trustworthy Agent Protocol*.

D. Trust Evaluation in Protocol 2

Ag_a should assess the trust value for each referee using the relevant information transmitted by the trustworthy agents or by other referees. This is done by applying Equation 5, in which Ag_b represents now the referee rather than the third party and Ag_i represent the agents who refer this referee. The new computed values will be used to evaluate the third party's trust. We can build a trust graph in order to deal with this issue. We define such a graph as follows:

Definition 7 (Trust Graph): A trust graph of a given agent Ag_a is a directed and weighted graph. The nodes are agents and an edge (Ag_j, Ag_w) means that agent Ag_j knows agent Ag_w. The weight of the edge (Ag_j, Ag_w) is a tuple (v, x, y, z) where v is the Ag_w's trust according to the point of view of Ag_j, x is the total number of interactions between Ag_j and Ag_w, y is the time of the recent interactions, z is the number of recent interactions. The weight of a node is the agent's trust value according to the point of view of Ag_a.

According to this definition, in order to determine the trust of the target agent Ag_b, it is necessary to find the weight of the node representing this agent in the graph. The graph is constructed while Ag_a receives answers from the consulted agents. The evaluation process of the nodes starts when the entire graph is built. This means that this process only starts when Ag_a has received all the answers from the consulted agents. The process terminates when the node representing Ag_b is evaluated. The termination is guaranteed since the number of consulted agents is finite.

The graph construction algorithm is as follows:

1) Ag_a sends a request about Ag_b's trust to all the trustworthy agents Ag_i. The nodes representing these agents (denoted $Node(Ag_i)$) are added to the graph. Since the trust values of these agents are known, the weights of these nodes (denoted $Weight(Node(Ag_i))$) can be evaluated. These weights are represented by $Tr_{Ag_a}^{Ag_i}$ (i.e. by Ag_i's trust according to the point of view of Ag_a).
2) Ag_a asks Ag_i to provide relevant information associated with Ag_b's trust. The answers by Ag_i are recovered when they are offered in a variable denoted Str. $Str.Agents$ represents the set of agents referred by Ag_i. $Str.Tr_{Ag_i}^{Ag_j}$ is the trust value of an agent Ag_j (belonging to the set $Str.Agents$) from the point of view of the agent who referred him (i.e. Ag_i). $Str.TN_{Ag_i}^{Ag_j}$ is the total number of interactions between Ag_i and Ag_j. $Str.N(\Delta t_{Ag_i}^{Ag_j})$ is the number of recent interactions between Ag_i and Ag_j. $Str.\Delta t_{Ag_i}^{Ag_j}$ is the time of these recent interactions.
3) When a consulted agent answers by indicating a set of agents, these agents will also be consulted. They can be regarded as *potential witnesses*. These witnesses are added to a set called: $Potential\_Witnesses$. When a potential witness is consulted, he is removed from the set.
4) To ensure that the evaluation process terminates, two limits are used: the maximum number of agents to be consulted ($Limit\_Nbr\_Visited\_Agents$) and the maximum number of witnesses who must offer an answer ($Limit\_Nbr\_Witnesses$).

The trust combination formula (Equation 5) is used to evaluate the graph nodes. The weight of each node indicates the trust value of the agent represented by the node. Such a weight is assessed using the weights of the adjacent nodes. For example, let $Arc(Ag_x, Ag_y)$ be an arc in the graph, Ag_x should be evaluated before evaluating Ag_y. Consequently, the evaluation algorithm is recursive. This algorithm terminates

because the nodes of the set $Trustworthy(Ag_a)$ are already evaluated by the construction graph algorithm. Since the evaluation is done recursively, the call of this algorithm in the main program has as parameter the agent Ag_b. This algorithm is specified in Algorithm 1.

Complexity

Our trust model is based on the construction of a trust graph and on a recursive call to the function Evaluate-Node(Ag_y) to assess the weight of all the nodes. Since each node is visited exactly once, there are n recursive calls, where n is the number of nodes in the graph. To assess the weight of a node we need the weights of its neighboring nodes and the weights of the input edges. Thus, the algorithm takes a time in $O(n)$ for the recursive calls and a time in $O(e)$ to assess the agents' trust where e is the number of edges. The run time of the trust algorithm is therefore in $O(max(e, n))$ i.e. linear in the size of the graph.

***Algorithm** 1:* **Node Evaluation**

Evaluate-Node(Ag_y**)** {
  $\forall Arc(Ag_x, Ag_y)$
    If $Node(Ag_x)$ is not evaluated Then
      Evaluate-Node(Ag_x)

  $m1 := 0,\ m2 := 0$
  $\forall Arc(Ag_x, Ag_y)$ {
    $m1 = m1 +$
      $Weight(Node(Ag_x)) * Weight(Arc(Ag_x, Ag_y))$
    $m2 = m2 + Weight(Node(Ag_x))$
  }
  $Weight(Node(Agy)) = m1/m2$
}

V. Trust Automation

A. Prototype

We have implemented a prototype allowing agents to communicate using the protocols and strategies given in this paper, reason using argumentation, and evaluate trust using the mechanisms we have provided above. The prototype is designed as a society of interacting agents equipped with knowledge bases and argumentation systems. It is implemented using the $Jack^{©TM}$ platform (The Agent-Oriented Software Group, 2004). Agents' knowledge bases contain propositional formulae (under the form of Horn clauses) and arguments. These knowledge bases are designed and implemented as $Jack^{©TM}$ data structures called $beliefsets$, which are used to maintain an agent's beliefs about the world. The meaning of the propositional formulae (i.e. the ontology) is recorded in a $beliefset$ whose access is shared among the agents. Agent communication is done by sending and receiving messages, which are events extending the basic $Jack^{©TM}$ $MessageEvent$ class. Dialogue games are implemented as a set of events and plans. A plan describes a sequence of actions that an agent can perform when an event occurs. Whenever an event is posted and an agent chooses a task to handle it, the first thing the agent does is to search a plan to handle the event. Dialogue games are not implemented within the agents' program, but as event and plan classes that are external to agents. An agent Ag_1 starts a dialogue game by generating an event and by sending it to the addressee Ag_2. Ag_2 executes the plan corresponding to the received event and answers by generating another event and sending it to Ag_1.

In the implemented prototype, agents inherit from the basic class $Jack^{©TM}$ $Agent$. The argumentation systems are implemented as Java modules using logic programming techniques. These modules use agents' $beliefsets$ to build arguments for or against certain propositional formulae. The trust model is implemented using events and plans. The requests sent by an agent about the trust of another agent are events and the evaluations of agents' trust are programmed in plans. The trust graph is implemented as a graph data structure. As Java classes, agents have private data called *Belief Data*. Fig. 6 illustrates the different data structures used in the prototype (agents' belief data, plans, and events).

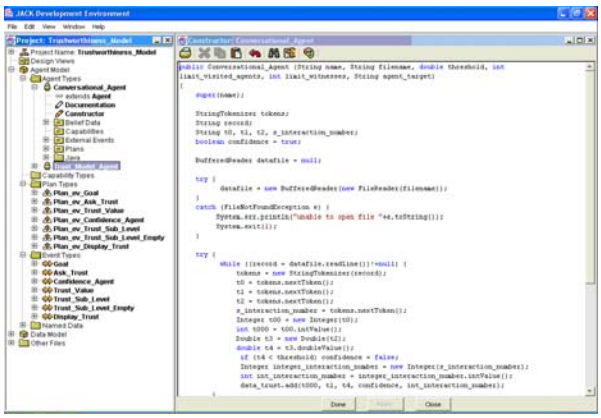

Fig. 6. The data structures of the prototype

The main steps of the evaluation process of Ag_b's trust are implemented as follows:

1) By respecting the two limits discussed in the graph construction algorithm, Ag_a consults his knowledge base and sends a request to his trustworthy agents Ag_i $(i = 1, .., k)$ about Ag_b's trust. Ag_a sends this request starting by agents having the highest trust value.
2) In order to answer the Ag_a's request, each agent Ag_i executes a plan instance. Thus, using his knowledge base, each agent Ag_i offers to Ag_a an Ag_b's trust value if Ag_b is known by Ag_i. If not, Ag_i proposes a set of trustworthy agents from his point of view, with the relevant information discussed above.
3) When Ag_a receives the information he asked from an agent Ag_i, he executes a plan in which he adds to the trust graph: **1)** the agent Ag_i and his trust value as graph node; **2)** The trust value that Ag_i offers for Ag_b, the number of times that Ag_i interacted with Ag_b, the time period of the recent interactions, and the number of these interactions as arc relating the node Ag_i and the node Ag_b. This first part of the trust graph is recorded until the end of the evaluation process of Ag_b's trust.
   When Ag_i offers a set of referees to Ag_a, this latter

executes another plan according to which he adds to another graph three pieces of information for each Ag_i agent: **1)** the agent Ag_i and his trust value as a sub-graph node; **2)** the nodes Ag_j representing the agents proposed by Ag_i; **3)** for each agent Ag_j, the relevant information related to his trust. This sub-graph is used to evaluate Ag_j's trust values using Equation 5.

4) Steps 1, 2, and 3 are applied again until all the consulted agents offer a trust value for Ag_b or one of the two fixed limits is reached.
5) Equation 5 is used to evaluate Ag_b's trust value using the recorded information during the previous steps.

B. Example

In order to evaluate our trust model, we used the developed prototype to run simulations. The purpose was to test how a given agent (Ag_1 in the example) determines if some agents are trustworthy or not by interacting with other agents in his social network. We created 300 agents in the simulated environment, 50 of them are malicious agents and 250 are trustworthy, and the average number of interactions between agents is 140 interactions. In this example, Ag_1 uses Protocol 2 (Section IV-C) to ask other agents in his network about the trustworthiness of 70 agents (the 50 malicious agents and 20 trustworthy agents). We are interested in this example in the number of detected malicious agents. The simulations show that this number increases with the size of the social network, which automatically increases the number of interactions. When the network achieves a given size (70 in the example), the number of detected malicious agents becomes constant and very close to the real number of malicious agents. Fig. 7 depicts these results.

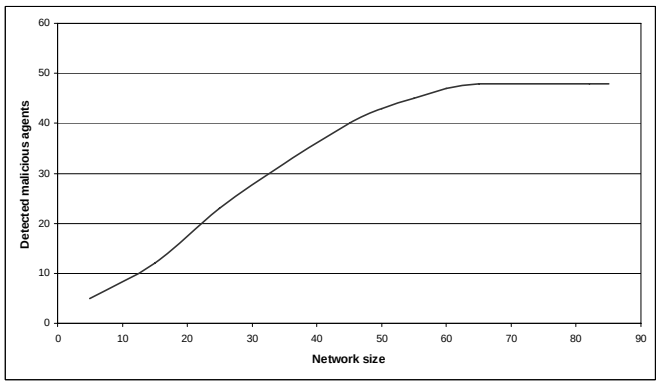

Fig. 7. Performance according to the network size.

VI. Related Work

Several Trust model systems have been proposed in recent years. In this section, we will discuss the most important of them: *PGP Trust Model* [2], *SPORAS* [28], *FIRE* [14], *ReGreT* [20], *Referral Model* [24], [25], and *PRUNES* [27].

PGP (Pretty Good Privacy) is an algorithm used to encrypt files and protect them from unauthorized access using public and private keys [29]. Each private key is paired with a public key that is available to anyone. At the level of trust, *PGP* adopts the *web of trust approach* [2]. There are no central authority which everybody trusts, but instead, individuals sign each other's keys and progressively build a web of individual public keys interconnected by links formed by these signatures. For example, let us assume that *Carol* requires some data from *Bob* whom *Carol* have never met before. *Alice*, one of *Carol*s colleagues, signs *Bob*'s public-key certificate which she knows is authentic. *Bob* then forwards his signed certificate to *Carol* who wishes to communicate with *Bob* privately. *Carol*, who knows and trusts *Alice*, finds out, after verification, that *Alice* is among *Bob*'s certificate signer (*Bob* could have more than one signature on his certificate to make it more widely acceptable). Therefore, *Carol* can be confident that *Bob*'s public key is authentic. In this case, *Carol* regards *Alice* to be an introducer for her. This approach is close to the social network adopted in this paper. However, the *PGP* trust model is only proposed to trust public keys, and does not explain how introducers are trusted and evaluated. In addition, unlike our model, the *PGP* trust model does not consider communication protocols between agents and introducers.

Recently, some online trust models have been developed (see [13] for a detailed survey). The most widely used are those on *eBay* and Amazon Auctions. Both of these are implemented as a centralized trust system so that their users can rate and learn about each other's reputation. For example, on *eBay*, trust values (or ratings) are +1, 0, or -1 and users, after an interaction, can rate their partners. The ratings are stored centrally and summed up to give an overall rating. Thus, reputation in these models is a global single value. However, the model can be unreliable, particularly when some buyers do not return ratings. In addition, these models are not suitable for applications in open (multi-agent) systems. e.g. for negotiation, because they are too simple in terms of their trust rating values and the way they are aggregated.

Another centralized approach called *SPORAS* has been proposed in [28]. *SPORAS* does not store all the trust values, but rather updates the global reputation value of an agent according to his most recent rating. The model uses a learning function for the updating process so that the reputation value can reflect an agent's trust. It introduces a reliability measure based on the standard deviations of the trust values. However, unlike our models, *SPORAS* deal with all ratings equally without considering the different trust degrees. Consequently, it suffers from rating noise. In addition, like *eBay*, *SPORAS* is a centralized approach, so it is not suitable for open systems.

Broadly speaking, there are three main approaches to trust in open multi-agent systems. The first approach is built on an agent's direct experience of an interaction partner. The second approach uses information provided by other agents [24], [25], [26]. The third approach uses certified information provided by referees [14], [23]. In the first approach, methods by which agents can learn and make decisions to deal with trustworthy or untrustworthy agents should be considered. In the models based on the second and the third approaches, agents should be able to reliably acquire and reason about the transmitted information. In the third approach, agents should provide third-party referees to witness about their previous performance.

Because the first approaches are only based on a history of interactions, the resulting models are poor because agents with no prior interaction histories could trust dishonest agents until a sufficient number of interactions is built.

Sabater [20] proposes a decentralized trust model called *Regret*. Unlike the first approach models, Regret uses an evaluation technique not only based on an agent's direct experience of his partners' reliability, but it also uses a witness reputation component. In addition, trust values (called *ratings*) are dealt with according to their recency relevance. Thus, old ratings are given less importance compared to new ones. However, unlike our model, Regret does not show how witnesses can be located. In addition, this model does not deal with the possibility that an agent may lie about his rating of another agent, and because the ratings are simply equally summed, the technique can be sensitive to noise. In our model, this issue is managed by considering the witnesses' trust and because our merging method takes into account the proportional relevance of each reputation value, rather than treating them equally (see Equation 5).

Yu and Singh [24], [25], [26] propose an approach called *Referral Model* based on social networks in which agents, acting as witnesses, can transmit information about each other. The purpose is to tackle the problem of retrieving ratings from a social network through the use of referrals. Referrals are pointers to other sources of information similar to links that a search engine would plough through to obtain a web page. Through referrals, an agent can provide another agent with alternative sources of information about a potential interaction partner. The social network is presented using a referral network called *TrustNet*. The trust graph we propose in this paper is similar to *TrustNet*, however there are several differences between our approach and Yu and Singh's approach. Unlike Yu and Singh's approach in which agents do not use any particular reasoning, our approach uses argumentation-based negotiation in which agents use argumentation-based reasoning. In addition, Yu and Singh do not consider the possibility that an agent may lie about his rating of another agent. They assume all witnesses are totally honest. However, this problem of inaccurate reports is considered in our approach by taking into account the trust of all the agents in the trust graph, particularly the witnesses. Also, unlike our model, Yu and Singh's model do not treat the timely relevance information and all ratings are dealt with equally. Consequently, this approach cannot manage the situation where the agents' behavior changes over time.

Huynh, Jennings, and Shadbot [14] propose a model called *FIRE* in order to tackle the problem of collecting the required information by the evaluator itself to assess the trust of his partner, called the target. The problem is due to the fact that the models based on witness implicitly assume that witnesses are willing to share their experiences. For this reason, they propose an approach, called certified reputation, based not only on direct and indirect experiences, but also on third-party references provided by the target agent itself. The idea is that the target agent can present arguments about his reputation. These arguments are references produced by the agents that have interacted with the target agents certifying his credibility (the model proposed by Maximilien and Singh [16] uses the same idea). This approach has the advantage of quickly producing an assessment of the target's trust because it only needs a small number of interactions and it does not require the construction of a trust graph. However, this approach has some limitations. In particular, because the referees are proposed by the target agent, this agent can provide only referees that will give positive ratings about him and avoid other referees, probably more credible than the provided ones. Moreover, even if the provided agents are credible, their witness could not reflect the real picture of the target's honesty. This approach can privilege opportunistic agents, which are agents only credible with potential referees. In addition, in this approach, the evaluator agent should be able to evaluate the honesty of the referees using a witness-based model. Consequently, a trust graph like the one proposed in this paper could be used. This means that, in some situations, the target's trust might not be assessed without asking for witness agents.

PRUNES (*Prudent Negotiation Strategy*) [27] introduced a negotiation strategy based on the use of *Digital Credentials*. It represents a complete automated trust negotiation strategy based on backtracking, which guarantees to find a successful negotiation whenever the credential strategies of the service requester and provider allow. PRUNES ensure that no irrelevant credentials are disclosed in the resulting negotiation. It guarantees that trust is established if allowed by the credential disclosure strategies. In the proposed model, credentials are treated as propositional symbols. It is also assumed that the credential sets of the two negotiation parties are disjoint. A credential disclosure strategy for credential C is defined to be in the form:

$$C \leftarrow F_c(S_1, \ldots, S_n)$$

where $F_c(S_1, \ldots, S_n)$ is an expression involving only credentials from other parties. S_i is satisfied if and only if the other party has shown credential S_i. Credential C can be shown to the other party only if $F_c(S_1, \ldots, S_n)$ is satisfied. A successful negotiation finds a credential exchange sequence $G = L_1, \ldots, L_j, S$, where each L_i is a credential, for $1 \leq i \leq j$, such that when two parties exchange credentials in the order defined by G, FL_i evaluates to true when it is time for L_i to be shown to the other party. Note that the requester can get the service S only if the requester proves his qualifications by showing a combination of credentials that satisfies S's access strategy, which can be expressed in the same manner as a credential disclosure strategy. Unlike our proposal, *PRUNES* does not consider parties as autonomous agents, but just entities exchanging simple messages. Also, the problem in *PRUNES* is not how to trust entities, but rather how to negotiate trust by exchanging Digital Credentials.

VII. Conclusion

The contribution of this paper is the definition and implementation of a new model to secure agent-oriented systems in which agents communicate with each other and reason using advanced decision making techniques. Two communicating protocols and different agent strategies have been presented, as well as several models, of increasing sophistication, for agents to make use of the information communicated to them

by other agents they consider trustworthy to determine the trust of further target agents. Our model has the advantage of being computationally efficient and of taking into account four important factors: (1) the trust (from the viewpoint of the evaluator agents) of the trustworthy agents; (2) the trust value assigned to target agents according to the point of view of trustworthy agents; (3) the number of interactions between trustworthy agents and the target agents; and (4) the timely relevance of information transmitted by trustworthy agents. The resulting model allows us to produce a comprehensive assessment of the agents' credibility in a software system.

ACKNOWLEDGMENT

This research was partially funded by The Royal Society, UK, under the International SHORT VISIT / FELLOWSHIP programme. The first and fourth authors are also supported by the Natural Science and Engineering Research Council, Canada and Le Fonds Québécois de la Recherche sur la Nature et les Technologies. The second author was also supported by the Sixth Framework IST programme of the EC, under the 035200 ARGUGRID project, and by a UK Royal Academy of Engineering/Leverhulme Trust senior fellowship.

REFERENCES

[1] A. Abdul-Rahman, and S. Hailes. Supporting trust in virtual communities. In Proc. of the 33rd Hawaii Int. Conf. on System Sciences, IEEE Computer Society Press. 2000.

[2] A. Abdul-Rahman. The PGP trust model. In EDI Forum: The journal of Electronic Commerce, April, 1997.

[3] L. Amgoud, N. Maudet, S. Parsons. Modelling dialogues using argumentation. In Proc. of the 4th Int. Conf. on Multi-Agent Systems, pp. 31-38, 2000.

[4] J. Bentahar, B. Moulin, J-.J. Ch. Meyer, and B. Chaib-draa. A computational model for conversation policies for agent communication. In J. Leite and P. Torroni editors, Computational Logic in Multi-Agent Systems. Lecture Notes in Artificial Intelligence (3487): 178-195, 2005.

[5] J. Bentahar, and J-J. Ch. Meyer. A New Quantitative trust model for negotiating agents using argumentation. The Int. Journal of Computer Science and Applications, 4(2), 1-19, 2007.

[6] J. Bentahar, M. Mbarki, and B. Moulin. Strategic and tactic reasoning for Communicating agents. Argumentation in Multi-Agent Systems, Springer LNAI 4766, 2007, 142–160

[7] J. Bentahar, Z. Maamar, D. Benslimane, and Ph. Thiran. An Argumentation framework for communities of web services, IEEE Intelligent Systems, 22(6), 2007

[8] A. Bondarenko, P.M. Dung, R. A. Kowalski, and F. Toni. An abstract, argumentation-theoretic approach to default reasoning. Artificial Intelligence 93(1–2), 63–101, 1997.

[9] M. Dastani, J. Hulstijn, and L. V. der Torre. Negotiation protocols and dialogue games. In Proc. of Belgium/Dutch Artificial Intelligence Conf., pp. 13-20, 2000.

[10] P.M. Dung. The acceptability of arguments and its fundamental role in non-monotonic reasoning and logic programming and n-person game. Artificial Intelligence 77, 321–357, 1995.

[11] P.M. Dung, P. Mancarella and F. Toni. Computing ideal sceptical argumentation. Artificial Intelligence, Special Issue on Argumentation in Artificial Intelligence, 2007.

[12] I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the grid: enabling the scalable virtual organization. The Int. Journal of High Performance Computing Applications, 15(3), 200-222, 2001.

[13] T. Grandison, and M. Sloman. A survey of trust in internet applications. IEEE Communication Surveys & Tutorials, 3(4), 2000.

[14] T. D. Huynh, N. R. Jennings, and N. R. Shadbolt. An integrated trust and reputation model for open multi-agent systems. Journal of Autonomous Agents and Multi-Agent Systems AAMAS, 2006, 119-154.

[15] N. Maudet, and B. Chaib-draa, Commitment-based and dialogue-game based protocols, new trends in agent communication languages. In Knowledge Engineering Review. Cambridge University Press, 17(2):157-179, 2002.

[16] E. M. Maximilien, and M. P. Singh. Reputation and endorsement for web services. ACM SIGEcom Exchanges, 3(1):24-31, 2002.

[17] P. McBurney, and S. Parsons, S. Games that agents play: A formal framework for dialogues between autonomous agents. In Journal of Logic, Language, and Information, 11(3):1-22, 2002.

[18] H. Prakken. Relating protocols for dynamic dispute with logics for defeasible argumentation. In Synthese (127):187-219, 2001.

[19] S. D. Ramchurn, T. D. Huynh, and N. R. Jennings. Trust in multi-agent systems. The Knowledge Engineering Review, 19(1):1-25, March 2004.

[20] J. Sabater. Trust and reputation for agent societies. Ph.D. Thesis, Universitat Autonoma de Barcelona, 2003.

[21] F. Sadri, F. Toni, and P. Torroni. Dialogues for negotiation: agent varieties and dialogue sequences. In Proceedings of the International workshop on Agents, Theories, Architectures and Languages. Lecture Notes in Artificial Intelligence (2333):405-421, 2001.

[22] E. Shakshuki, L. Zhonghai, and G. Jing. An agent-based approach to security service. International Journal of Network and Computer Applications. Elsevier, 28(3): 183-208, 2005.

[23] H. Skogsrud, B. Benatallah, and F. Casati. Model-driven trust negotiation for web services. IEEE Internet Computing, 7(6):45-52, 2003.

[24] B. Yu, and M. P. Singh. An evidential model of distributed reputation management. In Proc. of the First Int. Conference on AAMAS. ACM Press, pp. 294-301, 2002.

[25] B. Yu, and M. P. Singh. Detecting deception in reputation management. In Proc. of the 2nd Int. Conf. on AAMAS. ACM Press, pp. 73-80, 2003.

[26] B. Yu, and M. P. Singh. Searching social networks. In Proc. of the second Int. Conf. on AAMAS, pp. 65-72, 2003.

[27] T. Yu, M. Winslett, X. Ma. PRUNES: An efficient and complete strategy for automated trust negotiation over the internet. ACM Conference on Computer and Communications Security, Athens, Greece, November 2000.

[28] G. Zacharia, and P. Maes. Trust management through reputation mechanisms. Applied Artificial Intelligence, 14(9):881-908, 2000.

[29] P. Zimmermann. Pretty Good Privacy Users's Guide. Volume I and II, 1993.