Evaluating Trust in Peer-to-Peer Service Provider Communities

Ioan Petri*‡, Omer Rana†, Yacine Rezgui*, and Gheorghe Cosmin Silaghi‡
*School of Engineering Cardiff University,UK
Email: [petrii],[rezguiy]@cardiff.ac.uk
†School of Computer Science & Informatics, UK
Email: o.f.rana@cs.cardiff.ac.uk
‡ Business Information Systems, Babeş-Bolayi University, Romania
Email: gheorghe.silaghi@econ.ubbcluj.ro

***Abstract*—The increasing availability of Internet services has stimulated the development of peer-to-peer markets. These electronic markets have the potential for improving the efficiency of trading by reducing search and transaction costs. They also allow buyers to choose the best possible service for every transaction and interact with different sellers over time. In such markets, the establishment of trust is mandatory in order to ensure reliable market exchanges. We identify how trust can be established within a trading community, making use of pre-established Service Level Agreements between participants. The effectiveness of the approach in selecting trusted participants is evaluated via simulation.**

***Index Terms*—Electronic Markets, Service Level Agreements, Penalties, Rewards, Trust, Prisoner Dilemma**

I. Introduction

Electronic markets can bring together large numbers of buyers and sellers across different communities on significant scales. Hence, electronic markets have the potential for improving the efficiency of trading by reducing search and transaction costs. Online markets also allow buyers to choose the best possible deal for every transaction and interact with different sellers over time. This means that it is necessary to establish trust between buyers and sellers, especially where they may not have directly interacted with each other in the past. For online trading communities, the establishment of trust becomes a requirement for ensuring a secure environment for market exchanges.

Various views about trust exist within electronic communities and transactions. From one perspective, trust may be defined as a *subjective assessment* of influence within one system, changing perceptions about the quality and significance of a service [3]. Trust has an underlying *subjective* nature and may be used by one entity to control and sometimes manipulate other entities or groups. Trust may also be related to previous experiences which one entity has with another (based on previous interactions between the entities). Trust can also involve reciprocity, where one party can be morally obligated to give something in return for something received. However trust implies the necessity of a continuous evaluation in order to identify whether the interacting entity is dealing fairly or the level of reciprocity is adequate.

Trust can have different forms of representation in accordance with the mechanisms involved. For determining trust a variety of clues and past experiences are used to decide when such risk is appropriate. In addition, trust can be obtained by using indirect mechanisms such as social behaviours or third party experiences. Aggregating feedback and opinion about one entity (from a number of other entities) provides the reputation of an entity (considered as a community view about an entity) [7]. Reputation can also have an associated sanctioning role in social groups. When entities violate predefined trust standards they become subject to penalties. In the context of bilateral interactions involving risk, no stage can be performed until one party acquires a certain level of trust that can enable the second party to fulfil its obligations.

However, the study of trust outside formal mechanisms becomes more important in new communities where such mechanisms have yet to be firmly established. This is particularly the case for virtual (or electronic) communities today. These communities have created reputation or rating systems for the express purpose of encouraging trusting and trustworthy behaviours. For electronic markets, Service Level Agreements (SLAs) are efficient instruments for mediating business transactions between interacting entities – especially if such entities have not interacted in the past. An SLA may be used to specify quality of service terms, the measurement criteria, reporting criteria and penalty/reward clauses between participants involved in a transaction. Within an electronic market, an SLA may be used for: (i) an expression/proof of debts as well as credits – debts to the client and credits to the service provider; (ii) as a token of exchange between participants; (iii) as an identification of responsibilities of participants involved (such as the client and service provider). Establishing an SLA between two parties (client & service provider) implies that the service provider has agreed to provide a particular capability to the client within some quality of service. In return, the client must provide a monetary payment (most often) or credit to the provider once the service has been delivered (subject to a penalty, often also monetary, in case the quality of service terms have not been adhered to) [6]. When one party is unknown (provider or client) the level of risk associated with the transaction is considerably increased.

COLLABORATECOM 2011, October 15-18, Orlando, United States
Copyright © 2012 ICST
DOI 10.4108/icst.collaboratecom.2011.247125

Different studies [1], [7], [8] have been investigating how a trusted environment can be developed by using reputation as a metric for monitoring the system.

In our framework we consider both a client and a provider view – where compliance with an SLA for a provider can be measured, whereas clients provide feedback on their previous interactions to other clients (as a means of providing a recommendation). In this particular work we consider clients to have different types of behaviours (both truth telling and deception), whereby feedback about a particular provider may be influenced by particular incentives that a client may have. A key contribution of this work is to identify how malicious intent (based on incorrect feedback) can bias the overall trust establishment within a peer community of clients and service providers, and how trust values change with the number of clients involved in the community and with those providing feedback.

II. Related work

Trust Net [11] is designed to evaluate an agents' honesty by using a completely decentralised architecture, utilising a game theoretic approach [7]. In Trust Net each agent announces its strategy in advance and selects another agent based on this strategy. To calculate trust, each agent is evaluated based on the: (i) number of rounds during which the agent has been honest and (ii) number of total rounds. Agents can communicate their trust values (corresponding to trust in other agents) enhancing the process of convergence. A preliminary trust level is calculated in accordance with the values transmitted by other peers. The overall trust level relies on the aggregation of direct experiences and testimonials provided by other peers in the system.

The LIAR model [4] provides a mechanism for controlling the communication in P2P networks, such as support for query routing. The core functionality is based on expected behaviours of peer-nodes which are regulated with the help of social control, using a predefined level of acceptability for evaluating social commitments and trust intentions. A mechanism for detecting violations of social commitments and the associated sanctions enable peer-nodes to provide recommendations. Recommendation, observation and evaluation are used for calculating the trust based on indirect interactions. LIAR makes use of two different approaches: (i) direct interaction based reputation, and (ii) recommendation based reputation.

ReGret [10] is a modular trust system applied in the context of complex e-commerce environments where social relations between users have an important role. ReGreT uses an initial contract to identify the terms and conditions of the transaction. Based on the contract a set of outcomes and impressions are extracted. Impressions are handled as subjective evaluations of an outcome from a specific point of view. The reliability of direct trust is based on the number of outcomes and on the calculation of a deviation. Deviation in the context of ReGreT is the variability of rating values received for the same peer. Within ReGret, users gather beliefs from society members by using two criteria for evaluating the credibility of agents providing feedback: (i) social relations and (ii) past history.ReGret uses fuzzy rules to calculate how the structure of social relations influence the credibility of the information.

In EigenTrust [2] trust information is aggregated across all transactions between peers and a distributed calculation is used for building a trust matrix (identifying "transitive" or recommendation-based trust). All peer-nodes are cooperative and store a global trust vector(eigenvector) containing trust values for other peer-nodes. Each peer-node has a unique global trust value based on the peer's history of operations. Eigentrust works with a set of pre-trusted peer-nodes as the basis for the trust aggregation mechanism. In addition, the inactive peers and malicious collectives are handled as parts of the algorithm. A mechanism for normalising the local and global trust and a probabilistic interpretation reduces the complexity and increases the accuracy of the algorithm. It is proved that malicious collectives do not decisively boost the global values of peers while inactive peers are isolated. In the context of corrupted files, Eigentrust can reduce but cannot completely eliminate corrupted content.

PowerTrust [12] is a trust system for evaluating the trustworthiness of participating peer-nodes. PowerTrust implements a trust overlay network above all peer-nodes for modeling trust relationships and relates the approach to the power distribution of feedbacks from eBay. A dynamic selection of power nodes is used in order to improve the reputation accuracy and aggregation speed. The selection is performed based on a distribution ranking mechanism. The power nodes are represented by nodes which have a good reputation and they ensure the reliability of the scoring process when aggregating and produce global reputation. The PowerTrust process is triggered when a transaction takes place between any pair of peer-nodes. All peer-nodes provide scored for each other based on their transaction. The PowerTrust system aggregates local scores for calculating the global reputation of each participating peer-node.

NICE [16] is a trust model applicable for P2P systems which can establish steady cooperation among peers. NICE model is used to defend the community against malicious peers. Each peer creates a cookie with feedback based on what the other peers assign it at the end of each transaction. When service delivery is successful, the value of the cookie is positive, otherwise, the value is negative. In contract with other trust approaches, the NICE model provides facilities for enhancing interaction between peers. When the requesting peer does not have a direct cookie, it can check the trust path between itself and the providing peer and shows this to the provider [13]. A positive cookie can be exchanged by interacting peers while a negative cookie is retained by the peer that creates it. The NICE model also ensures that negative cookies are untempered and available to other peers in the system. In NICE every peer has a preference list of good peers, and maintains it based on the past interaction history [15].

PeerTrust [9] represents an adaptive trust model for quantifying and comparing the trustworthiness of peers based on a transaction-based feedback system. In PeerTrust, the trustworthiness of one peer node is based on the feedback/evaluation it receives in providing a service to other peers in the past. PeerTrust uses a number of factors when evaluating peer nodes: (i) the feedback a peer obtains from other peers, (ii) the feedback scope, such as the number of transactions that a peers has with other peers, (iii) the credibility factor associated with the feedback source, (iv) the transaction context factor for discriminating mission-critical transactions from less or non-critical ones and (v) the community context factor for addressing community related characteristics and vulnerabilities. Table I provides a comparison between related reputation-based trust models.

III. Approach and Methodology

When a client selects provider(s) for establishing Service Level Agreements (SLAs), especially if clients have not directly interacted with particular providers, it is useful to identify the level of trust that can be placed in the selected provider. Hence, when peer i identifies that peers $\{j_1, ..., j_n\}$ contain the required capability (and consequently the SLA can be established), before establishing an SLA peer i has to determine how much it can trust each peer in this list.

A. Trust calculation

Consider a collection of peer-nodes P=$[p_1, p_2, p_3, ..., p_n]$, some of which can provide the required resource (service). Considering p_j as the selected peer-node, we use a trust metric L_{ij} for defining the trust rating assigned by p_i to node p_j as a consequence of previous interactions.

The SLA subsequently established between these peers contains a set of attributes
A=$[A_1, A_2, A_3, ..., A_n]$, such as availability, response time, integrity, latency, etc. The trust calculation is based on one or more such attributes, where each attribute can have an associated weighting (indicating its importance). The local trust between peer-nodes i & j can be specified as:

$$T_{ij} = \frac{L_{ij}}{\sum_{k=1}^{n}(max(0, L_{ij}))} \tag{1}$$

where $L_{ij}^{A_i} = \sum_{k=1}^{n}(P_{ij} - N_{ij})$ represents the local trust between p_i and p_j, P_{ij} represents the number of positive interactions between p_i and p_j, while N_{ij} identifies the number of negative interactions between p_i and p_j from the context of peer node p_i. A_i represents the attributes used to rate the interactions as being either positive or negative. When multiple attributes are used $i > 1$, $L_{ij}^{A_i}$ = $\phi[L_{ij}^{A_i}]$ = $\alpha_1[L_{ij}^{A_1}]$ + $\alpha_2[L_{ij}^{A_2}]$ + ... + $\alpha_n[L_{ij}^{A_n}]$.

It is important to note that the positive interactions P_{ij} are calculated in relation with all the service level agreements previously established between p_i and p_j – P_{ij}: $SLA_i^j(pen \leqq 0, rew \geqq 0)$, where pen represents the penalty as an SLA parameter and rew identifies a reward. One positive interaction identifies the case when an SLA has been completed without violations(regular SLA or rewards). On the other hand the negative interactions N_{ij} are identified in the context of associated penalties caused by SLA violations – N_{ij}: $SLA_i^j((pen \geqq 0, rew \leqq 0))$.

B. Global trust

When rating peer-nodes, local trust can be subjective. In order to ensure an objective rating we calculate the global trust of peer-node i by combining direct interactions from immediate (generally one hop) neighbours $[j_1, j_2, ..., j_n]$ with indirect interaction of $[k_1, k_2, ..., k_n]$ with each peer-node j_x. This is the same as the Eigentrust model [2] described in section II

$$T_{jk} = \frac{L_{jk}}{\sum_{p=1}^{n}(max(0, L_{jk}))} \tag{2}$$

where $L_{jk}^{A_j} = \sum_{p=1}^{n}(P_{jk} - N_{jk})$ is the local trust between p_k and p_j, P_{jk} represents the number of positive interactions between p_k and p_j, while N_{jk} identifies the number of negative interactions between p_k and p_j. Hence, $T_{ik} = T_{ij} * T_{jk}$ represents the trust level of node i calculated as the product between the local trust T_{ij} and the indirect interaction rating T_{jk} obtained from those peer-nodes in the neighbourhood of node j. After phases of feedback processing, each peer-node is evaluated based on the aggregated value between local trust T_{ij} and global trust T_{jk}. We use a threshold δ for deciding whether a peer-node is trusted or untrusted. The protocol works with a predefined $view$ parameter which determines the number of immediate (i.e. one hop) neighbours for each peer node.

Hence, each peer-node uses an associated *view of peers* parameter. The associated $view$ of one peer-node i is formed by a set of peer-nodes j_1, j_2,...,j_n, whereas each j_1, j_2, ... ,j_3 peer-node has aother associated $view$ such as $view(j_1)$ = [$k_{j_{11}}$, $k_{j_{12}}$,..., $k_{j_{1n}}$], $view(j_2)$ = [$k_{j_{21}}$, $k_{j_{22}}$, ..., $k_{j_{2n}}$] and $view(j_n)$ = [$k_{j_{31}}$, $k_{j_{32}}$, ... , $k_{j_{3n}}$]. The local trust between i and j_i is specified by L_{ij_m} while the local trust between j_i and $k_{j_{mp}}$ is specified by $L_{j_m k_p}$.

C. Trust implications

Aggregate trust values are therefore based on feedback provided by neighbouring peers, based on their prior interactions and opinions. It is useful to note that a neighbouring peer only sends a single value to the requesting peer – and does not reveal the mechanism it has used to calculate this trust value. Hence, if $T_{ik} = T_{ij} * T_{jk}$ is the global trust of peer j, where peer j represents a potential SLA partner for i, peer k can "behave" in a number of possible ways when asked for feedback about j. In such a context the value of T_{ik} can be inaccurate as T_{jk} may not reflect the real interaction history between k and j. We use a game theoretic approach to capture

Trust model	Centralised	Metric	Type of feedback	SLA or QoS negotiation
REGRET	no	[-1,1]	continuous	QoS verification
EigenTrust	no	[0,1]	binary	na
PeerTrust	no	[0,1]	continuous	na
NICE	no	[0,1]	continuous	na

TABLE I
COMPARISON OF REPUTATION-BASED TRUST SYSTEMS [1]

the perceived payoff that a peer sees when returning feedback to a requesting peer – based on the Prisoners Dilemma utility mechanism. Using this approach, each peer-node can perform different behaviours according its own decision function – which is used to determine the expected utility (or payoff) that a peer sees when returning this information to a requesting neighbour. In many other trust-based systems, a truth telling feedback mechanism is assumed.

In our approach, peer-nodes are scheduled to perform different behaviours. Each peer-node has an associated decision function $f_{dec}(p) : P \rightarrow M$, where P represents the set of peer and M the set of behaviours each peer node can perform. We assume that peer p_i has an associated behaviour m_i which can change over time and is not constant.

In our Prisoner Dilemma approach, each peer has an associated incentive i_i for each behaviour m_i, defined within I_{PD}=$[(m_1, i_1), (m_2, i_2), (m_3, i_3), ..., (m_n, i_n)]$. Hence, a peer node i can *decide* to perform $m_i \in M$ because according to its subjective decision function $f_{dec}(p_i)$, m_i enables it to maximise its utility.

From the set of behaviours M=$[m_1, m_2, m_3, ..., m_n]$, $\forall$ $M_m \subseteq M$, where M_m=$[m_1, m_2, m_3, ..., m_p]$, $p < n$, represents the set of behaviours that are malicious. Over time, each peer-node can perform one malicious behavior $m_i \in M_m$ or a set of malicious behaviours such as m_k=$m_i \wedge m_j$, where $m_i \in M_m$. We consider two types of malicious behaviours, (i) p_i provides incorrect feedback when queried; (ii) $m_j \in M_m$ identifies the case when p_j stops interacting with one peer and joins another (i.e. it removes itself from the immediate neighbourhood of one peer and joins the neighbourhood of another) – we refer to this as a *migration*. Therefore, $m_k \in M_m$ would represent a sum of behaviours such as migrating to the view of other peers (m_j) and providing incorrect feedback (m_i). The extent to which these behaviours exist in our simulation framework is controlled by two probability values:

1) p_{ml}(single malicious behaviour probability) – the probability for simulating a single behaviour(providing incorrect feedback) such as m_i.
2) p_{mi}(multiple malicious behaviour probability) as the probability for simulating sets of behaviours such as m_k=$m_i \wedge m_j$ – peer-nodes which can migrate over the network and join different views of peers (m_i) and provide incorrect feedback in different $views$ of peers (m_j).

IV. SIMULATOR

We use PeerSim [5] for simulation, as components may be 'plugged in' and used as a simple ASCII file based configuration. PeerSim consists of modules that can be used to construct and initialize the underlying P2P network, handle various protocols, control and modify the network. It is an open source, Java-based simulation framework for developing and testing P2P algorithms in a dynamic environment which can work in two different modes: cycle-based or event-based. The cycle based engine relies on a time scheduling algorithm.

PeerSim identifies the following components:

- *protocols* – used to define the behaviour of the different peers. They can be of different types such as handling and simulating the overlay network, or implementing a distributed algorithm
- *nodes* – represented as entities of the P2P network. Each peer-node has a stack of protocols, one protocol simulating the behaviour of peer-nodes.
- *controls* – used to control the simulation, either at regular intervals or during the initialization. They can be simple observers which gather statistics or they can modify the simulation changing different protocols.

Simulation in PeerSim is controlled through Java objects that can be scheduled for execution at certain points. These `controllers` typically initialise, observe or modify the simulation. The initialization phase is carried out by control objects that are scheduled to run only at the beginning of each experiment. In the configuration file, the initialization is undertaken using a Scheduler object which identifies the simulation cycles, and when they are executed. This object can also be used to configure a protocol or be executed in specific simulation cycles. It is also possible to control the order of execution of the components within each cycle.

V. RESULTS

We conduct a number of experiments to analyse the effect of different malicious behaviours on the overall trust distribution of a P2P network. The system uses different types of behaviours with different associated payoffs. Each experiment presents the trust distribution in the context of different types of behaviours assigned to each peer-node. For simulating behaviours we use an unstructured P2P architecture with peers providing feedback. We use a cycle based simulation process with 2000 peer-nodes scheduled to perform different behaviours according to the execution probabilities – p_{ml}, p_{mi}.

Experiment 1 : The level of trust during 1000 execution cycles, when the malicious behaviour probability p_{ml} is varied within the interval [0.01,0.1].

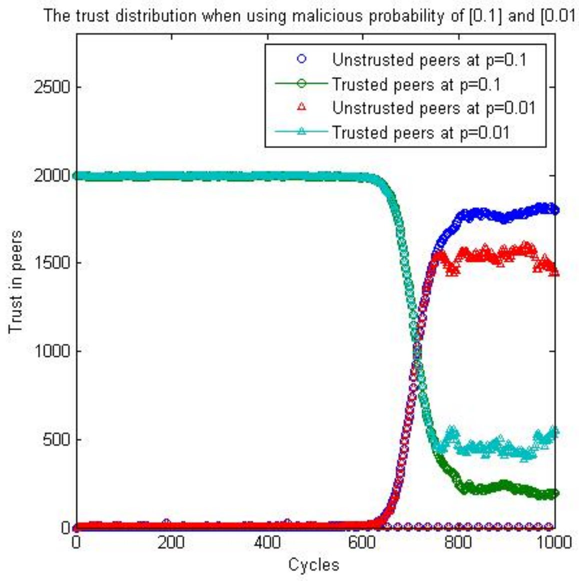

Fig. 1. A cycle based representation of trust at different malicious probabilities

In the context of a malicious probability of p_{ml}=0.01 we observe a continuous variation in terms of trusted/untrusted peers over the interval [600-700]. The trust level becomes stable after cycle 750. When using a malicious probability of p_{ml}=0.1 the decay in terms of trusted peers is associated with the interval [700-800]. This behavior is influenced by the availability of feedback at particular points in the simulation. When peer-nodes start sending feedback the trust level decreases because of malicious behaviour involving incorrect feedback. The processing of feedback provides a method of altering the level of trust within the system. Within the experiment an increase in untrusted peers is induced by feedback which can be malicious and which are aggregated for the trust calculation(see section III-B). We can conclude from the experiment that a malicious probability of p_{ml}=0.1 has a higher impact over the system in term of trust distribution than p_{ml}=0.01 because p_{ml}=0.1 leads to greater malicious behavior over the simulation time.

Experiment 2 : The trust distribution in the context of 2000 peer-nodes where the probability of malicious behaviour is varied in accordance with the set p_{ml}=[0.05,0.01,0.1].

Figure 2 illustrates the level of trust expressed in terms of trusted and untrusted peers after 1000 simulation cycles. It is important to note that this experiment presents the distribution of trust at the end of the simulation process. This experiment illustrates the impact of the trust distribution when the simulator is configured to use a malicious behaviour probability of p_{ml}=0.05. With this probability, the number of peers which can provide malicious feedback is high and therefore the distribution of trust is significantly affected.

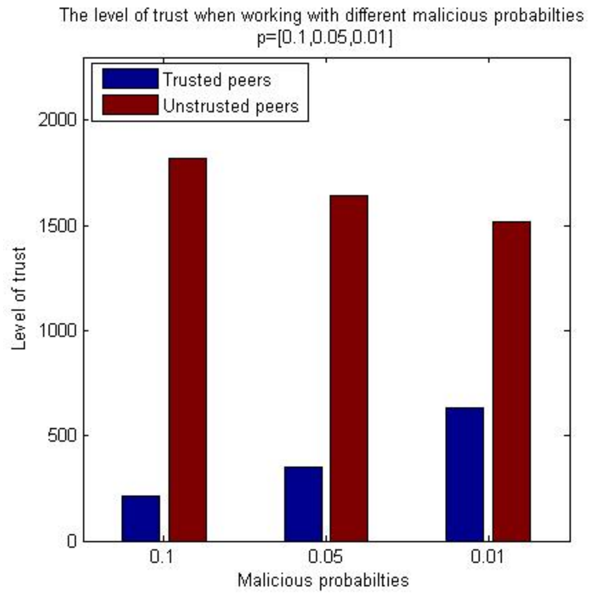

Fig. 2. Trust level at different malicious probabilities

Similar distribution is observed for a malicious probability of p_{ml}=0.01. A significant change of trust can be observed when the system uses p_{ml}=0.1 – recording the highest number of malicious behaviours and therefore the trust level identifies the highest decrease in terms of trusted peers. It is important to note that within this experiment we keep the migrating probability fixed.

Experiment 3 : A cycled based representation of trust when the system works with different migration probabilities p_{mi}=[0.1,0.01]. The network size is set to 2000 peer-nodes and the probability of malicious behaviours is fixed.

Figure 3 illustrates how the level of trust evolves during 1000 simulation cycles. It can be observed that migration of peers as a complex malicious behaviour destabilises the level of trust within the system. Within this experiment, a migration probability of p_{mi}=0.1 produces a higher decrease in terms of trusted peers than p_{mi}=0.01. This confirms that when more peers are scheduled to migrate, a decay is induced within the overall trust level. The trust equilibrium in terms of trusted and untrusted peers is reached after 800 execution cycles in the context of p_{mi}=0.1. An equilibrium is reached when the number of trusted and untrusted peers are equally distributed within the system.

Within the experiment, two different equilibrium states are identified at approximately cycle 750 for p_{mi}=0.1. For an associated migration probability of p_{mi}=0.01 the equilibrium is reached within the interval [950-100] cycles.

As mentioned before, the migration rate of peers represents the cause of the trust distribution during the simulation. This explains how a migrating probability of p_{mi}=0.1 produces a higher impact in terms of untrusted peers than p_{mi}=0.01.

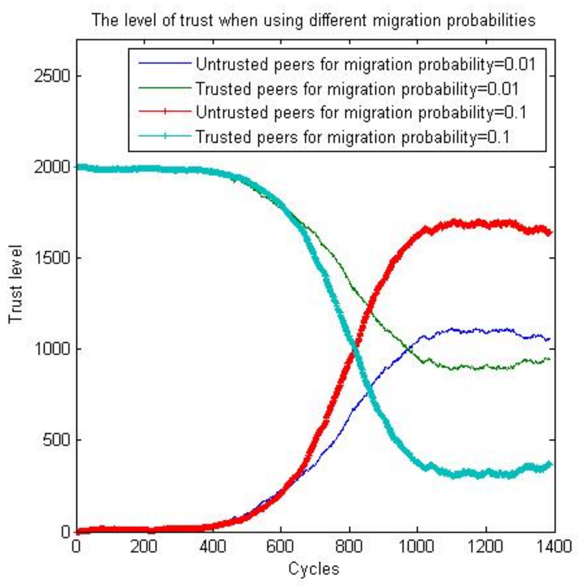

Fig. 3. Cycle based representation of trust at different migrating probabilities

Decrease and increase in terms of trusted/untrusted peers is induced by the process of feedback provision. When peers start providing feedback within the system, malicious behaviours are spread among peers, altering the level of trust. The level of trust stabilises when the processing of feedback is terminated.

Experiment 4 : The level of trust in the context of different migration probabilities p_{mi}=[0.5,0.1,0.01,0.001] after 1000 execution cycles. This experiment uses 2000 peer-nodes in network size.

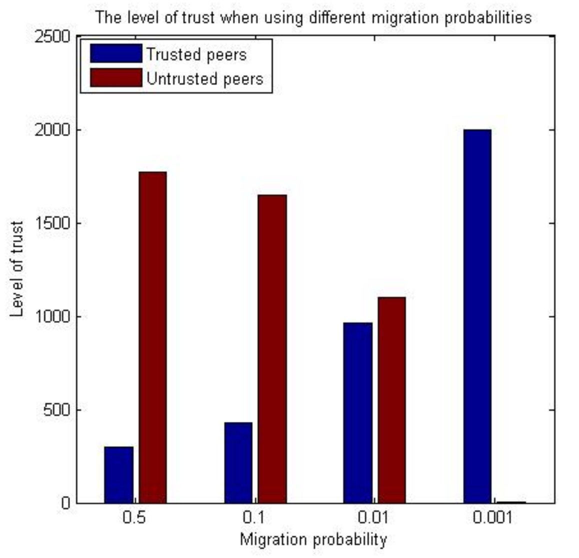

Fig. 4. Trust level at different migrating probabilities

This experiment presents the distribution of trust when the system experiments different migrating probabilities(see section III-C). From figure 4 is observed how the malicious behavior of migrating to different $view$ of peers affects the overall distribution of trust. For a migration behaviour probability p_{mi} of 0.5 the proportion between trusted and untrusted peers is significantly affected as p_{mi}=0.5 identifies a high number of migrating behaviours among peer-nodes. Similar distribution of trust is recorded for the intervalp_{mi}=[0.1, 0.01]. It is important to note that for the probability of migrating p_{mi}=0.5 the system records the highest level of untrusted peers. When using a probability of migrating of p_{mi}=0.001 the system records the highest level of trusted peers because p_{mi}=0.001 identifies a low level of migrating behaviours. In this case, as figure 4 illustrates, the number of untrusted peers is very low($\gg 0$).

Experiment 5: This experiment investigates how the level of trust is distributed across the P2P community when expanding the size of the network. The experiment uses a variable network size [4000, 8000] with a migration probability of p_{mi}=0.01 and a probability of malicious behaviours set to p_{ml}=0.05. Figure 5 presents the trust distribution in the context of large scale networks. In accordance with previous experiments, the trust level maintains similar tendency even when dealing with a large scale system. It is interesting to observe that the trust equilibrium is touched at an earlier stage when the system uses a network size of 4000 peer-nodes. This takes place after [750-800] simulation cycles while for a large scale network of 8000 peer-nodes the experiment would need more simulation cycles in order to reach an equilibrium value. On the other hand, the fixed migration and malicious behavior probabilities seem to have higher impact on trust in the context of smaller scale systems.

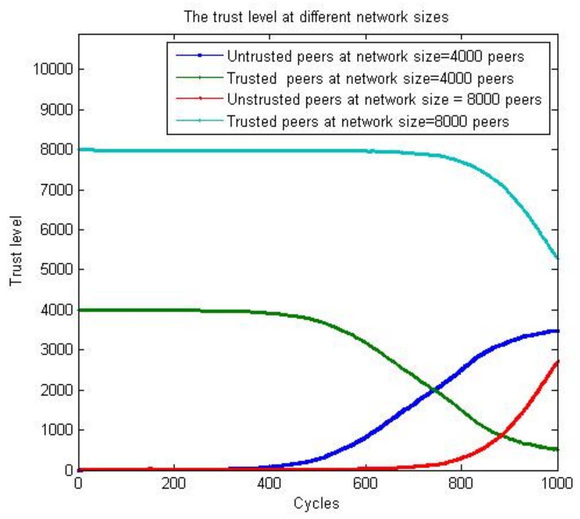

Fig. 5. Cycle based trust level at different network sizes

The equilibrium states along the experiments are determined by the process of feedback provision. The processing of feedback starts at cycle 400 for a network of 4000 peers while for a network of 8000 peers the processing of feedback is scheduled to take place at cycle 800. For small scale systems (network size of 4000), the trust equilibrium is reached at cycle

750 which identifies an equal distribution of trusted/untrusted peers. For large scale systems the simulator needs more cycles in order to reach a trust equilibrium.

Experiment 6: This experiment provides a representation of trust when the simulator works with different *views* of peers. The view of peers is a parameter identifying the number of links (connective paths) one peer-node has within the network – representing the number of peers in the immediate neigbourhood (one hop connection) of each peer. In the context of fixed execution probabilities this experiment investigates how the size of peer communities interferes with the distribution of trust. With a network size of 2000 peer-nodes, the experiment presents the level of trust when view=[20] and view=[5].

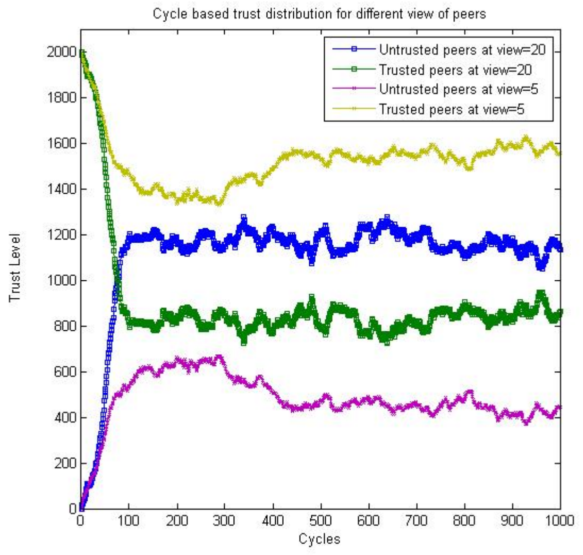

Fig. 6. Cycle based trust level at different views of peers

From figure 6 it can be observed that large communities are more affected by malicious behaviours than small communities. A community of peers is identified by the number of peers in view. It is important to note that a high number of peers in view (large community) amplifies the effect of malicious behaviours because the number of processed feedback increase. When using lower values for the view parameter, the effect of malicious behaviours is reduced because the number of feedback decrease. Therefore, in the context of our calculation the number of processed feedback represents an important parameter for the distribution of trust. The equilibrium state is reached for view=20 after approximately 100 execution cycles as well as the decrease/increase in terms of trusted/untrusted peers relies on the process of feedback provision.

Experiment 7: Trust distribution when working with different views of peers. The execution probabilities are fixed and the view of peers is varied according to the view interval of [20,15,10,5]. The network size is set to 2000 peer-nodes.

This experiment presents how the size of peer-communities (simulated with the view parameter) affect the distribution of trust in terms of trusted/untrusted peers. This experiment

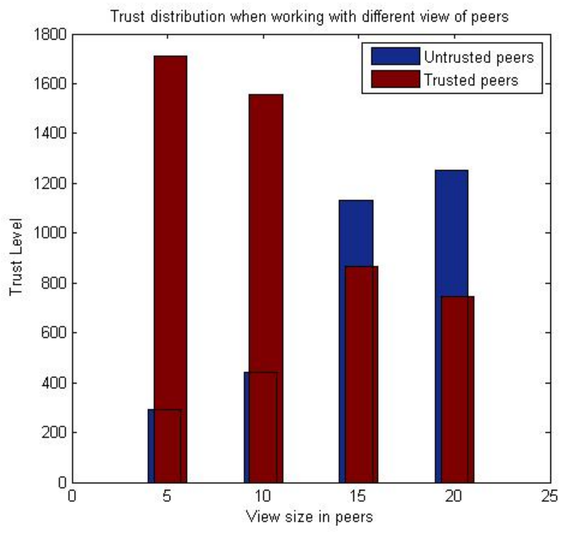

Fig. 7. The trust level at different views of peers

illustrates the distribution of trust after the simulator runs 1000 cycles. It is observer from figure 7 that a view of 20 peer-nodes has a significant impact over the distribution of trust. This is illustrated by the decrease in terms of trusted peers taking place within the system. When using 15 respectively 10 peers in view the impact over the trust distribution is lower than for 20 peers in view. This happens because 20 peers in view(large communities) represent a higher number of feedback processed during the simulation. The lowest impact over trust is identified when working with a view size of 5 peers. In this case the number of processed feedback is limited therefore the impact is reduced.

VI. Conclusions

This paper aims to provide a reliable mechanism for the trustworthy selection of partners for establishing Service Level Agreements.To address the limitation of untrusted environments we propose a model of selecting partners based on their trust level. We use an algorithm that enables peers to provide feedback about their direct (with a distance of one hop) or indirect (with a distance of multiple hops) neighbours. The algorithm uses as input the associated penalties and rewards from previous SLA exchanges and provides how the trust level changes as new participants enter the market (offering different types of services).

We also demonstrate how changes in behaviour (mailicious and truth telling) impacts the overall trust within the system. Malicious behaviour is controlled through two probability values that causes a peer to either provide incorrect feedback or alter its connectivity (referred to as migration). Simulating different scenarios by varying the probability values, we show that the level of trust within a system is closely related to the types of malicious behaviours existing within the service provider community. It was demonstrated that a complex malicious behaviour can significantly alter the level of trust whereas a simple malicious behaviour can only alter the trust level of a specific community. When simulating different malicious behaviours for dynamic systems, where the size of the network can vary, it is observed that large scale networks are more affected by malicious behaviours than small scale networks.

References

[1] Arenas, Alvaro E. and Aziz, Benjamin and Silaghi, Gheorghe Cosmin, Reputation management in collaborative computing systems, Security and Communication Networks, John Wiley & Sons, Ltd., pp. 546–564, (2010).

[2] Sepandar D. Kamvar and Mario T. Schlosser and Hector Garcia-molina, The EigenTrust Algorithm for Reputation Management in P2P Networks, in Proceedings of the 12th International World Wide Web Conference, (2003).

[3] Kim, S., Societies, A., & Simulation, S., Effects of a Trust Mechanism on Complex Adaptive Supply Networks: An Agent-Based Social Simulation Study, Order A Journal On The Theory Of Ordered Sets And Its Applications, 12(3), 4, 2009.

[4] Guillaume Muller and Laurent Vercouter, L.I.A.R.: Achieving Social Control in Open and Decentralised Multi-Agent Systems, Applied Artificial Intelligence, pp. 723-768, (2010).

[5] Márk Jelasity, Alberto Montresor, Gian Paolo Jesi and Spyros Voulgaris. “The Peersim Simulator”. Downloadable at: http://peersim.sf.net.

[6] Petri, Ioan and Rana, Omer and Cosmin Silaghi, Gheorghe, SLA as a Complementary Currency in Peer-2-Peer Markets, Economics of Grids, Clouds, Systems, and Services, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, pp. 141-152, (2010).

[7] Lik-Mui, Computational Models of Trust and Reputation: Agents, Evolutionary Games, and Social Networks, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, PhD thesis, (2002).

[8] Roger Dingledine and Nick Mathewson and Paul Syverson, Reputation in P2P Anonymity Systems, In Workshop on Economics of Peer-to-Peer Systems, (2003).

[9] Li Xiong and Ling Liu, Reputation management in collaborative computing systems, PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities, IEEE Transactions On Knowledge and Data Engineering, pp. 843–857, (2004).

[10] Jordi Sabater and Carles Sierra, Social ReGreT, A reputation model based on social relations, SIGecom Exchanges, pp. 44-56, (2002).

[11] Schillo, M. and Funk, P. and Rovatsos, M., Using Trust for Detecting Deceitful Agents in Artificial Societies, Applied Artificial Intelligence, Special Issue on Trust, Deception and Fraud in Agent Societies, pp. 825–848, (2000).

[12] Runfang Zhou and Kai Hwang, PowerTrust: A Robust and Scalable Reputation System for Trusted Peer-to-Peer Computing, IEEE Trans. Parallel Distrib. Syst., pp. 460-473, (2007).

[13] G. Suryanarayana, M. H. Diallo, J. R. Erenkrantz, and R. N. Taylor. Architectural Support for Trust Models in Decentralized Applications. In ICSE06, Shanghai, China, May, (2006).

[14] Yan Wang and Vijay Varadharajan. Dynamicnust: The Trust Development in Peer-to-Peer Environments, In IEEE Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTCO6), (2006).

[15] Hai Ren. Comparision of Trust Model in Peer to Peer System. TTK T-110.5290 Seminar on Network Security. 2006.

[16] Rob Sherwood, L. Seungjoon and B. Bhattacharjee, Cooperative peer groups in NICE, Computer Networks, 50(4):523–544, (2006).