ITEC 810 – Information Technology Project

A Survey of Trust Evaluation Methods

Supervisor: Yan Wang

Erden Sacan

(Student ID: 40916332)

Department of Computing,

Faculty of Science,

Macquarie University

Sydney, Australia

erden.sacan@students.mq.edu.au

TABLE OF CONTENTS

1

Abstract

Trust and reputation management systems are essential parts of open networks. For example if requesting A is planning to download files from serving B in a P2P information-sharing network, the reputation management system can calculate and provide information about the trustworthiness of party B based on the collected ratings provided by other peers.

There is a demand for trust evaluation from P2P networks, e-commerce applications, service oriented computing and multi-agent systems with different focuses. Therefore it is not realistic to look for one perfect solution that fits all fields. In this paper we focus on the trust evaluation criteria of these networks to classify different existing trust methods by considering their different trust parameters (e.g. user ratings, temporal dimension, transaction amount, etc.) and mathematical structure, and then analyse the trust evaluation requirements of different fields.

1 Introduction

Many fields of computer applications require trust evaluation and there is a variety of existing trust evaluation methods in these fields. It is even hard to talk about a single definition of trust; it varies from field to field.

There are various definitions of trust. 'Trust represents an agent's estimate of how likely another is to fulfil its commitments.' [Griffiths, 2006]. In another work [Grandison and Sloman, 2000] trust is defined to be the firm belief in the competence of an entity to act dependably, reliably and securely within a specific context.

The high trust value of a party in an open P2P network may be equal to a minimum trust value for an e-commerce transaction. In a P2P file sharing network, a transaction can be ranked as simply successful or not (1 or 0). Nevertheless in an e-commerce transaction we need to know much more than that (e.g. quality of product? delivery time? , etc...) to decide about a trust value. For each field we need the criteria for evaluating trust, to do it efficiently and to be able to offer the right features.

Therefore, in this paper we will discuss different trust evaluation methods for different types of applications. Then we will provide analysis of trust evaluation requirements of these networks.

We decided to analyse four major areas in application, in the scope of this review; e-commerce applications, peer-to-peer networks, service oriented computing and multi-agent systems. Trust issues in social networks will be out of the scope of this review.

In section 2, we will give chosen examples of successful solutions to trust evaluation requirements of these areas and we will explain why they have been successful. In section 3, we will clarify the differences between trust evaluation requirements of these areas.

2 Trust Evaluation Methods

In this section, we will analyse major examples of trust evaluation methods under the areas they are focused in.

2.1 E-commerce

People use Internet more and more to access information about products or services they are interested in or purchase those products and services. With the number of interactions and transactions growing, e-commerce trust evaluation systems are becoming more important.

2.1.1 eBay

One of the most successful online business models, eBay [ebay.com, 2009], is dealing with numerous customers, buyers and sellers, and a huge number of transactions. To deal with this huge traffic eBay keeps its reputation mechanism simple.

After a transaction, both the seller and the buyer can rate each other by selecting +1 (positive), 0 (neutral) or -1 (negative) as the feedback rating. This rating value then is added to the total "Feedback Score" of the seller or buyer. These feedback scores are kept and calculated centrally by eBay. The formula below shows 3 rating types of ratings given by a buyer depending on overall quality of transaction (S) and minimum acceptable quality of transaction (λ).

$$r(S) = \begin{cases} "+" & \text{if } S > 0 \\ "-" & \text{if } S \leq -\lambda \\ \text{no rating} & \text{if } -\lambda < S \leq 0 \end{cases}$$

Formula 1: eBay feedback values [Dellarocas, 2001]

Time factor on the ratings is kept really simple too. Ratings older than 6 months are being ignored by the system. In addition, if the same user rates the same seller more than once in one week time, the average for that week is calculated separately. This average affects seller's rating total by either +1 or -1 only (counts as 1 vote).

The binary reputation system of eBay is dependent on ranking characteristics of parties. If parties rank each other on how much the product's quality meets the advertised quality [Dellarocas, 2001] the reputation mechanism can work well and be accurate. If the binary reputation system is not using the right parameter to rate the parties, it wouldn't be successful (For example, because the seller has little control on the price, it should not be a parameter of rating)

Even though eBay doesn't force the parties to rate each other, empiric studies [Resnick, Zeckhauser & Swanson, 2006] show that around half of the parties in eBay rates the transaction after it is completed: although this provides a good number of feedbacks, a remarkably small number of parties provide negative scores after a transaction (1% to 2%). [Resnick, Zeckhauser & Swanson, 2006] Buyers are afraid of being harassed by e-mails from the seller. In addition, sellers often contact with buyers to solve conflicts and convince them not to give negative rating. eBay hasn't implemented any mechanism to stop these or doesn't have a culture that encourages negative ratings.

This structure might be easy to maintain and cheap to run but it doesn't provide enough information about the seller for many buyers. eBay has an additional detailed seller rating system which takes delivery time and item quality into account, however because it is not enforced, it is not practically functional.

2.1.2 PeerTrust

Reputation mechanism requirements are different within online businesses. Some need more complex systems, such as PeerTrust [Xiong and Liu, 2003], to provide more detailed and more accurate trust information to their users. For example, binary ratings might not be sensitive enough for some applications and they might need parties to rate each other out of 5, in different aspects. In an e-commerce environment some possible trust parameters would be;

Amount of satisfaction: This value would show how much the seller met the promised quality of service or product.

Delivery time: This value shows whether or not the service or product has been delivered on time.

Number of transactions: In eBay structure a seller may keep increasing its reputation by selling more, even if it gets negative feedbacks. If the calculation of trust takes "number of transactions" into account, misleadingly high trust values can be prevented.

Source party credibility: This basically shows how reliable or how biased the source of the feedback is. Sources with higher credibility have bigger effect on the total trust value.

Transaction context: The importance of transactions might be different in a business. For example bigger transactions might value more in terms of trust calculation.

Community context: The community of the environment might have different drivers. In some communities, it would be effective to reward accurate feedbacks.

By taking these factors into account, a reputation system might provide sufficient information and tackle the issue of parties giving intentional incorrect feedback.

The formula below shows how PeerTrust uses these parameters, where;

- $I(u, v)$ = total number of transactions performed by peer *u* with *v*,
- $I(u)$ = the total number of transactions performed by peer *u* with all other peers
- $p(u, i)$= other participating peer in peer u's i^{th} transaction
- $S(u, i)$= the normalized amount of satisfaction peer *u* receives from $p(u, i)$ (in its i^{th} transaction)
- $Cr(v)$= the credibility of the feedback submitted by *v*
- $TF(u, i)$ = the adaptive transaction context factor for p eer u's i^{th} transaction
- $CF(u)$ = adaptive community context factor for peer *u*.
- $T(u)$ =The trust value of peer *u* .

$$T(u) = \alpha * \sum_{i=1}^{I(u)} S(u,i) * Cr(p(u,i)) * TF(u,i) + \beta * CF(u)$$

Formula 2: PeerTrust general trust formula [Xiong and Liu, 2003]

PeerTrust [Xiong and Liu, 2003] doesn't use a central database. Every party has a trust manager which calculates the necessary trust values locally. The distribution of these trust values are controlled by another tool called data locator (See Figure 1 below). This structure distributes the processing work among parties but brings security and privacy concerns.

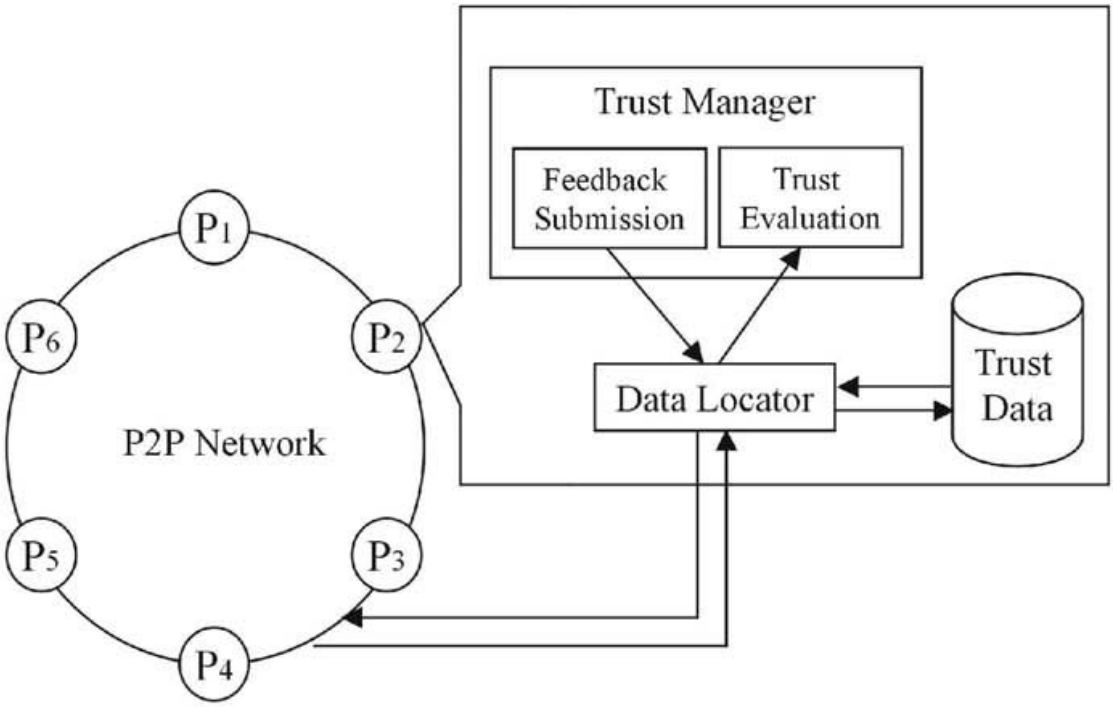

Figure 1: PeerTrust system architecture [Xiong and Liu, 2003]

2.2 P2P

Peer-to-Peer (P2P) information sharing networks are mostly open to public, are big in scale and have no central authorities. Because of this structure, in most cases the requesting party doesn't have any experience with majority of the serving parties. Therefore, a requesting party needs assistance to decide about the reliability of serving parties. Reputation systems collect reputation information about serving parties, then calculate and provide their trust values to the requesting party.

It has been suggested that there are five main features of peer-to-peer reputation systems [Kamvar, Schlosser&Molina, 2003]:

Self-policing: No central management

Anonymity: The reputation value wouldn't give out any information about user's real ID.

No profit to newcomers: The reputation system should value long-term good services of parties; it shouldn't encourage parties with short-term bad reputations to switch to a new account to take advantage of it.

Minimal overhead: The system shouldn't need too much calculation work, bandwidth or storage space.

Robust to malicious collectives: The system should be resistant against fraud attempts, such as creating multiple accounts to vote itself or a group of users positively voting each other continuously.

In such distributed systems, since there is no central decision mechanism, parties collect reputation information and calculate a trust value for the parties they want to interact with, by themselves.

2.2.1 Yu, Sing and Sycara

Many models use polling algorithms to collect information about a party. It simply works by sending a broadcast to neighbours and collecting replies. It is a simple way of doing this but it uses a lot of bandwidth and takes more time to process replies. [Yu, Sing and Sycara, 2004] It is calculating credibility of the advising parties by weighted majority algorithm according to the accuracy of ratings they provided in the past about other parties. This approach balances the effect of exaggerated ratings.

In this model [Yu, Sing and Sycara, 2004], parties help each other to find the witness parties to gather information from more accurate sources. They provide referrals to each other, names and addresses of parties which have experience with the target party. The formula below shows calculation of trust value in this model, where;

- P_j = Target party
- $\{W_1, \ldots, W_L\}$ = Group of witnesses towards target party
- L = Number of witnesses found
- $R(W_k, P_j)$ = W_k's local rating for P_j

- w_k = weight for the credibility of W_k

$$\mathcal{P} = \begin{cases} \sum_{k=1}^{L} w_k * R(W_k, P_j)/L & L \neq 0 \\ 0.5 & L = 0 \end{cases}$$

$$T(P_i, P_j) = \begin{cases} \eta R(P_i, P_j) + (1-\eta)\mathcal{P} & L \neq 0 \\ 0.5 & L = 0 \end{cases}$$

Formula 3 [Yu, Sing and Sycara, 2004]

2.2.2 Wang and Varadharajan

There are many other different approaches to these problems from different reputation system models. In an earlier model, Wang and Varadharajan suggest a system that calculates the probability of achieving a successful transaction with the target party according to binary trust feedbacks about the target party. The system calculates the probability of target party's trust value in a given scope, depending on the feedbacks given by other parties, by using Gauss Distribution [Wang and Varadharajan, 2004]. The formula for the calculation is shown below, where;

$(v_1, v_2]$ $(v_1 < v_2,\ v_1,\ v_2 \in [0,1])$ = Given scope

T = Trust value

$P_\alpha^X(v_1, v_2)$ = Probability of X's trust value is in given scope

$$P_\alpha^X(v_1, v_2) = P(v_1 < T \leq v_2) = \frac{1}{\sqrt{2\pi}\sigma} \int_{\frac{v_1-\mu}{\sigma}}^{\frac{v_2-\mu}{\sigma}} e^{-\frac{x^2}{2}} dx$$

Formula 4 [Wang and Varadharajan, 2004]

2.2.3 Sporas

Because of the dynamic structure of P2P networks, “how old a feedback is” becomes an important variable. Sporas is a decentralized reputation mechanism [Zacharia&Maes, 2000] which can be adapted to focus on the changes in user’s behaviour in time. To do that, it checks the date of ratings and gives a higher value to newer ratings. In addition the new users in the system get the lowest possible rating, which satisfies the requirement in P2P systems mentioned above. If there is more than one rating between two parties, the system keeps record of only the newest. The formulae of the model are shown below, where;

- t = the number of ratings the user has received so far,
- θ = a constant integer greater than 1,
- W_i = rating given by the user i ,
- R^{other} = reputation value of the user giving the rating
- D = the range of the reputation values,
- σ = the acceleration factor of the dumping function Φ

$$R_{t+1} = \frac{1}{\theta} \sum_{1}^{t} \Phi(R_i) \bullet R_{i+1}^{other} \bullet (W_{i+1} - E(R_{i+1}))$$

$$\Phi(R) = 1 - \frac{1}{1 + e^{\frac{-(R-D)}{\sigma}}}$$

$$E(R_{t+1}) = R_t / D$$

Formula 5: Sporas [Zacharia&Maes, 2000]

2.2.4 Trust[2]

As it is mentioned above, because of the scale of P2P networks there is an uncertainty about whom to trust as a serving party. Likewise, it is not clear whom to trust among reputation ranking providers. Therefore, the credibility of a party who ranks the target peer is a major variable that needs to be taken into account. For example, a comprehensive trust model, Trust[2], does that by eliminating or decreasing the values of recommendations from parties with low credibility, provides a less noisy pool of reputation values [Wang and Varadharajan, 2005]. The credibility of a party is measured by calculating the deviation of its recommendations within the mainstream of recommendation values. If this deviation is high, that means a low credibility.

The more a party gets interacted with the target party, the more its recommendations about that peer become accurate and reliable. This is defined as ‘confidence of a party’. It

also considers how old a recommendation is. Fresher recommendations worth more within the calculation of total trust value. .

2.2.5 Griffiths, Chao and Younas

In some P2P environments parties would be interested in not only the success but also the quality of the transaction. Fuzzy logic can be suitable for trust in this manner, because it makes it possible to mathematically deal with unclear terms by converting qualitative comments to quantitative measures by defining value intervals for different aspects such as quality [Griffiths, Chao and Younas, 2006].

2.3 Service Oriented Computing

Service Oriented Computing can be defined as the set of concepts, principles, and methods that represent computing in Service-Oriented Architecture (SOA) in which software applications are constructed based on independent component services with standard interfaces [Tsai&Chen 2006]. It provides a way of building software focusing on open systems where new services, clients and providers may join or leave the system continuously [Billhardt, Hermoso, Ossowski and Centeno, 2007]

SOC is mainly designed to assist distributed software development, within or among companies. In this context, a SOC may include three types of parties [Tsai&Chen 2006];

Service providers: Do the coding part of services and share it through a common interface.

Service brokers: Manage registration and publishment of services among other parties.

Application builders: Bring the services together by using basic high level specification language according to end user requirements.

Besides software development, Service Oriented Computing is used in a variety of sectors such as travel agencies (for selection and combination of hotel, car rental, travel insurance and airline services) and e-retailers. A figure of services oriented example is shown below, by Billhardt, Hermoso, Ossowski and Centeno;

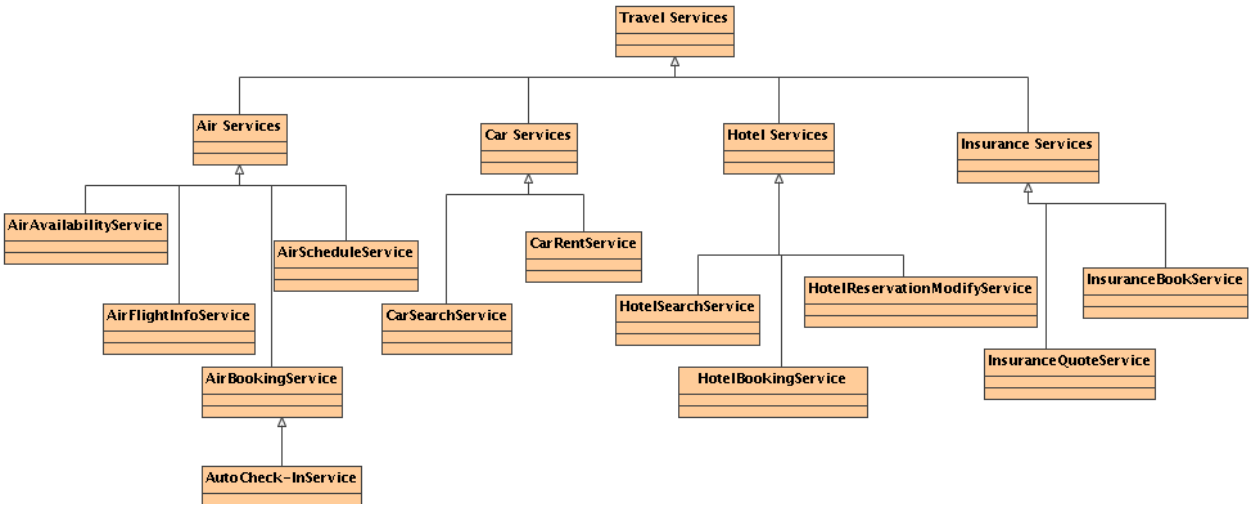

Figure 2: Services oriented example for a travel agency [Billhardt, Hermoso, Ossowski and Centeno, 2007]

A service in service oriented computing can be defined as a behaviour that is provided by a party to be used by any other party based on a network-addressable interface contract (generally identifying some capability provided by the service), it stresses interoperability and may be dynamically discovered and used[Cardiff University, 2008]. Therefore, this service [Dustdar, 2008];

- Has a standardised interface
- Is self-contained with no dependencies to other services
- Is available to other parties
- Needs little integration to be used by other parties
- Is context-independent
- Allows for service composition
- Has measurable Quality of Service attributes.

2.3.1 QoS Reputation

Quality of service (QoS) is the major component of decision making in service oriented computing. Some models use trust and reputation techniques to decide between offered services within the network and foresee the future quality of service.

When a simple ranking mechanism that doesn't take credibility of information providers into account is used in SOC, it is possible for malicious parties to exploit the system by distributing false ratings. Thus, a trust mechanism should check the trustworthiness of information providers as well.

QoS-based service selection calculates a trust value called "QoS reputation" for each party in the network. This reputation value is calculated [Vu, Hauswirth & Aberer, 2005] by:

1. Finding trusted parties by looking at the accuracy of the prior reports they provided

2. Marking them as honest

3. Marking all reports provided by them as honest

4. Comparing unmarked reports to find dissimilar reports to honest reports and marking them cheating

5. Comparing unmarked reports to find similar reports to cheating reports and mark them cheating.

6. Calculating QoS reputation by looking at the number of honest and cheating reports.

It improves the accuracy of trust evaluation on the information provided by parties. Nevertheless, it doesn't solve the problem of having new parties which has just a few reports about them or the parties with a big number of incomparable reports about them.

2.3.2 Billhardt, Hermoso, Ossowski and Centeno

When there is no prior transaction for a particular service between two parties, the requesting party needs to find another source of reputation information. In many trust evaluation methods, this problem is solved by using recommendations of parties. One model [Billhardt, Hermoso, Ossowski and Centeno, 2007] suggests prior experiences between these two parties, which are actually about different services, can be used to estimate a trust value for the target service of target party. By doing this, this model prevents from malicious parties' misleading and/or biased rankings. This model builds its structure on these assumptions:

- Various services from the same party would have similar quality.
- If they are provided by the same party, similar services would have more similar quality.

Based on these assumptions, the trust value is calculated as shown in the formulae below, where;

- $c_{C\to\langle P,S\rangle}$ = confidence of party (by taking similarities of offered services "S" into account)
- $t_{C\to\langle P,S\rangle}$ =trust value [0…1]
- $(r_{C\to\langle P,S\rangle})$ =reliability

$$c_{C\to\langle P,S\rangle} = \frac{\sum_{\langle X,Y\rangle\in LIT_C} c_{C\to\langle X,Y\rangle}\cdot r_{C\to\langle X,Y\rangle}\cdot sim(\langle X,Y\rangle,\langle P,S\rangle)}{\sum_{\langle X,Y\rangle\in LIT_C} r_{C\to\langle X,Y\rangle}\cdot sim(\langle X,Y\rangle,\langle P,S\rangle)}$$

$$t_{C\to\langle P,S\rangle} = \begin{cases} c_{C\to\langle P,S\rangle}, & if \quad r_{C\to\langle P,S\rangle} > \theta \\ \dfrac{\sum_{X\in NC\cup\{C\}} c_{X\to\langle P,S\rangle}\cdot w_{X\to\langle P,S\rangle}}{\sum_{X\in NC\cup\{C\}} w_{X\to\langle P,S\rangle}} \end{cases}$$

Formula 6 [Billhardt, Hermoso, Ossowski and Centeno, 2007]

Cooperative interactions include more aspects than transaction success. For example, one party may promise to deliver a certain service at a given quality, in the given timeline for a given price. In this situation, while judging this party, all these 3 dimensions should be taken into account. Therefore the trust evaluation system in such environment should provide these dimensions.

2.3.3 Wang, Lin, Wong, Varadharajan

Fuzzy logic helps us to define classes such as “the class of tall men”, “low quality” or “very high price in a mathematical manner and make them a part of calculations [Zadeh, 1965]. It works as a bridge between linguistic way of thinking of human and maths equations. It gives a system ability to deal with uncertainty and imprecision effectively such as this qualitative formula;

If T=new *and* Z=expensive *then* x=very good

A later framework uses fuzzy logic in a more comprehensive way [Wang, Lin, Wong, Varadharajan, 2008]. The model is built up on rules. Such rules can change the formula to be used, dependent on the type of event occurring. It doesn’t allow a new user to gain high trust values even if they provide extremely good services in the beginning. The formulae below provides a faster increase in the early stages of memberhip.

$$bf(x) = \frac{e^{\alpha x} - e^{-\alpha x}}{(e^{\alpha x} + e^{-\alpha x})\beta} \qquad (\alpha \geq 1 \quad \beta \geq 1)$$

Formula 7 [Wang, Lin, Wong, Varadharajan, 2008]

The figure below shows the trust improvement curve in time in this model ($\alpha = 3, \quad \beta = 1$):

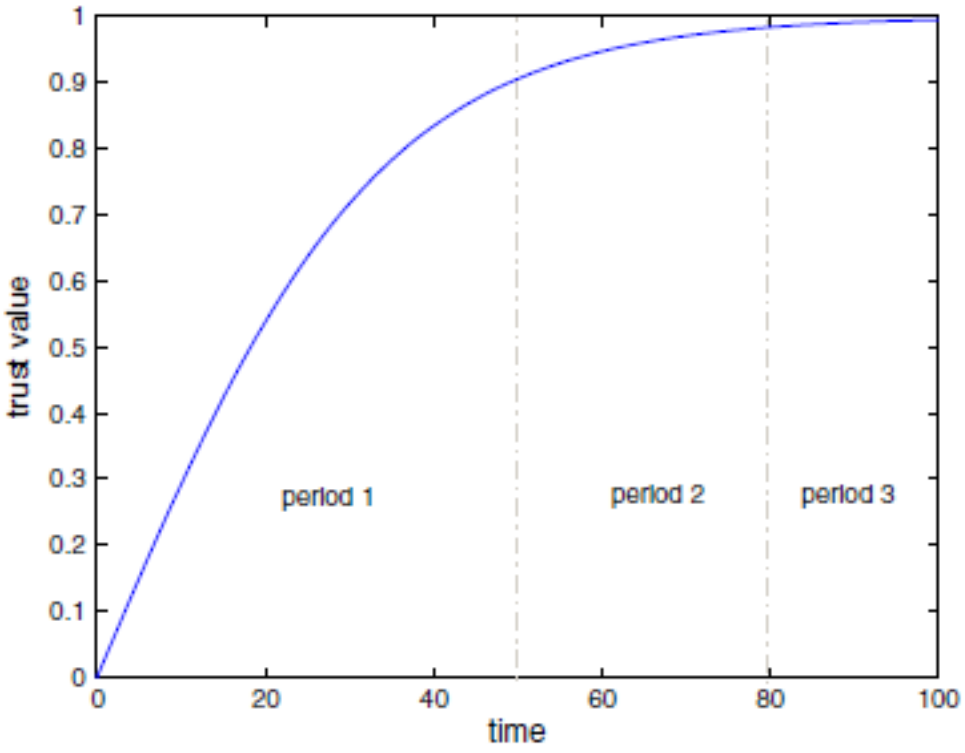

Figure 3: Trust improvement curve [Wang, Lin, Wong, Varadharajan, 2008]

Even if two parties have exactly the same trust value, it's still possible that they would have different reputation levels (one "very good", the other "good") For example; two service providers: one "very new", the other "very old" in the network, have the same reputation value, which is "very low"(see Table 1 below). With the same reputation score, the new party gets the trust value of 2; however, the old one gets the trust value of 0.

T	Very low	Low
t		
Very new	Low as new ($\acute{2}$)	Medium as new ($\acute{3}$)
New	Low as new ($\acute{2}$)	Medium as new ($\acute{3}$)
Medium	Low ($\grave{2}$)	Low ($\grave{2}$)
Old	Very low ($\grave{1}$)	Low ($\grave{2}$)
Very old	Extremely low ($\grave{0}$)	Very low ($\grave{1}$)

Table 1 [Wang, Lin, Wong, Varadharajan, 2008]

These linguistics terms are defined by membership functions. For example, trust rank is categorised by five fuzzy sets: "very low", "low", "medium", "high" and "very high". Some samples for these membership functions are below;

$$\mu_{\text{low}}(T) = \begin{cases} \frac{T-T_1}{T_3-T_1} & \text{if } T_1 \le T \le T_3 \\ \frac{T_5-T}{T_5-T_3} & \text{if } T_3 < T \le T_5 \\ 0 & \text{otherwise} \end{cases}$$

$$\mu_{\text{medium}}(T) = \begin{cases} \frac{T-T_4}{T_6-T_4} & \text{if } T_4 \le T \le T_6 \\ \frac{T_8-T}{T_8-T_6} & \text{if } T_6 < T \le T_8 \\ 0 & \text{otherwise} \end{cases}$$

Formula 8: Trust rank membership functions [Wang, Lin, Wong, Varadharajan, 2008]

Likewise, service period is categorised by five fuzzy sets as well: "very new", "new", "medium"," old" and "very old". Some samples for these membership functions are below;

$$\lambda_{\text{very new}}(t) = \begin{cases} \frac{t_2-t}{t_2} & \text{if } 0 \le t \le t_2 \\ 0 & \text{otherwise} \end{cases}$$

$$\lambda_{\text{new}}(t) = \begin{cases} \frac{t-t_1}{t_3-t_1} & \text{if } t_1 \le t \le t_3 \\ \frac{t_5-t}{t_5-t_3} & \text{if } t_3 \le t \le t_5 \\ 0 & \text{otherwise} \end{cases}$$

Formula 9: Service period membership functions [Wang, Lin, Wong, Varadharajan, 2008]

$$\phi_{i-j}(t, T) = \lambda_i(t) \times \mu_j(T)$$

Formula 10: Satisfaction degree function (calculated from membership functions) [Wang, Lin, Wong, Varadharajan, 2008]

The final reputation score value is calculated by using satisfaction degree function by the formula below: where;

S_{i-j} = Rank score

$$\mathrm{val}(S) = \frac{\sum_{\phi_{i-j} \in \Phi} \phi_{i-j} \times \mathrm{val}(S_{i-j})}{\sum_{\phi_{i-j} \in \Phi} \phi_{i-j}}$$

Formula 11: Service period membership functions [Wang, Lin, Wong, Varadharajan, 2008]

As it is mentioned above, the curve of reputation value doesn't allow new users to gain reputation quickly. Whereas for the old one an "average" value means whether the provider has been at that level for long time or started to struggle in the market. Thus "average" for a new provider actually means a better reputation value than the old one has. Therefore this model calculates reputation by using level of trust and time variables.

2.4 Multi-Agent Systems

MAS (Multi-Agent Systems) use mostly automated agents to solve problems (software design, trade) distributed among different domains. Because of this multi-domain structure, trust methods may have problems accessing sources for feedback.

2.4.1 FIRE

FIRE [Huynh, Jennings&Shadbolt, 2006] model proposes a system that combines four different sources of trust information;

Interaction trust: Prior direct experiences with the target party

Role based trust: What are the relationships of target party with other parties, is it a member of a trustworthy group?

Witness reputation: Experiences of other parties with the target party

Certified reputation: The target can collect third-party references to provide information about itself

Therefore if a change happens in the network at the time of trust evaluation requesting, the impact on the accuracy of trust value would be tolerable by the support of other sources.

The formulae that this model uses for trust calculation is below, where;

- $\mathcal{T}_K(a,b,c)$ = the trust value that agent a has in agent b with respect to term c
- $\mathcal{T}(a,b,c)$ = overall trust value
- $\mathcal{R}_K(a,b,c)$ = Set of ratings collected
- $\omega_K(r_i)$ = the rating weight function that calculates the relevance or the reliability of the rating r_i
- W_I = Interaction Trust
- W_R = Role Based Trust
- W_R = Witness Reputation
- W_R = Certified Reputation

$$\mathcal{T}_K(a,b,c) = \frac{\sum_{r_i \in \mathcal{R}_K(a,b,c)} \omega_K(r_i) \cdot v_i}{\sum_{r_i \in \mathcal{R}_K(a,b,c)} \omega_K(r_i)}$$

$$\mathcal{T}(a,b,c) = \frac{\sum_{K \in \{I,R,W,C\}} w_K \cdot \mathcal{T}_K(a,b,c)}{\sum_{K \in \{I,R,W,C\}} w_K}$$

$$w_K = W_K \cdot \rho_K(a,b,c)$$

Formula 8: FIRE [Huynh, Jennings&Shadbolt, 2006]

2.4.2 M_{DT-R}

Another model, M_{DT-R} [Griffiths, 2006] uses 'risk of cooperating' feedbacks to calculate a reputation value. In such network, it is assumed that all parties are allowed to be a member of the network, offer services and resources and do connections. Thus, M_{DT-R} does not use trust values to evaluate how secure to connect to other parties. Instead, the model uses trust to evaluate the cost, quality, following timeline ability and success of cooperation with that particular party.

It takes the amount of changes in trust value into account as well. If a new trust value from a party is much greater or much less than the existing trust value of target peer, it affects the total trust more.

3 Trust Evaluation Requirements

In this section, we will list our findings about trust evaluation requirements of E-commerce, P2P, SOC and Multi-Agent networks. We will provide the differences between these fields, in terms of reputation mechanism success criteria.

3.1 E-commerce

To validate the information on a service or product from a particular party, it is necessary to use opinions of a number of other parties about the target party. In addition, it is necessary to validate the trustworthiness of those opinions in the same way.

In e-commerce networks the parties generally concern the detailed information about a particular party before getting involved in a transaction with it, such as:

- The quality of products it is selling
- How timely the delivery is
- In terms of replying customer questions, how agile it is
- How many successful transactions it had in a given time period
- How big the amount of these transactions are

3.2 P2P

P2P file sharing networks are very big in size and have a very dynamic structure. Therefore in general, a transaction history between two specific parties does not exist.

Binary rating systems work very well for file sharing systems. Because in this type of systems the major concern is whether the file being shared is the completely correct the version or not.

In P2P networks there are parties sharing resources, sending queries to each other to check the availabilities, if there are available connections, they start transferring the resource. In a typical file sharing network, such as Gnutella or BitTorrent, there are good and bad parties. Good ones join the network to contribute to the resource sharing; they download and upload related resources. Nevertheless, because of the open structure of P2P networks it is easy to join for malicious entrants. These bad ones join the network to upload malicious resources, advertising purposes or simply slow down the network.

In such environment there is a need of a reputation mechanism that would assist distinguishing good parties from bad ones. In a nutshell, trust methods that are focusing in P2P networks can consider offering these features;

- Decentralised (distributed) reputation feedback collection about target party, no central storage
- Use minimum possible bandwidth and system resources
- Collect, calculate and provide trust values only when there is a demand for them.
- Taking credibility of feedback provider parties into account
- Assigning absolute minimum reputation value to new entrants
- New (fresh) feedbacks should weigh more than old ones

3.3 Service Oriented Computing

Service oriented computing provides integration or development of business systems using multiple sources through web services architecture. For example, the development of the application software can be distributed among different service providers through a service oriented network.

Service selection mechanisms are being used in service oriented computing (such as cooperation design processes, multi-supplier supply chains) to find the most relevant provider based on the requirements of searcher party.

In such systems, each party in the network is an independent decision maker. To get the most out of offered services, a party in such environment must select a suitable service provider for its requirements. Each party has its own requirements, in terms of quality, cost and timeline of the service or resource. Every party in the network may offer a service or resource to other parties. These services and resources have a variety of quality and cost. To keep the system up and running these parties must work together.

Therefore we can say that in a service oriented computing environment, after a party starts a search for a service provider for specific requirements, firstly the pool of service providers are filtered to a smaller pool of providers who are offering the requested service, then the trust method takes place and selects the most appropriate service provider.

Such trust method assisted service provider selection environment is shown in the figure below, by Billhardt, Hermoso, Ossowski and Centeno;

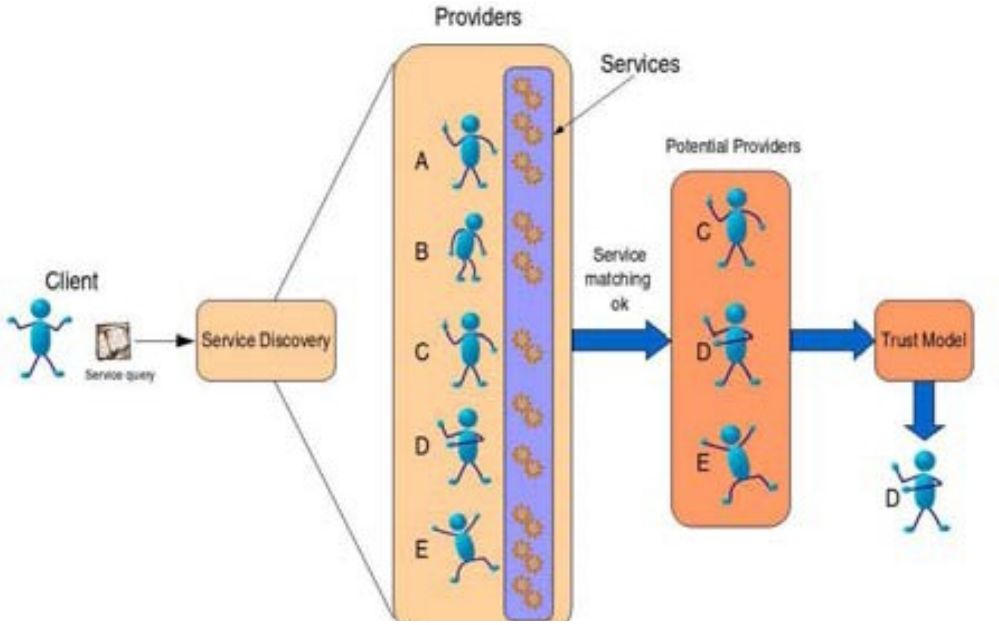

Figure 4: Provider Selection Using Trust Methods in SOC Environments

[Billhardt, Hermoso, Ossowski and Centeno, 2007]

Since every party is independent about the service or resource it provides, every transaction, purchase, connection or transfer of data has a risk of unsuccessfulness. Such as not meeting the requirements of buyer party by bad quality resource, highly priced resource, not providing the service in promised timeline or failing to provide the service at all. In addition, these two parties generally are members of different security domains. Therefore these features should be considered in SOC trust methods;

- The cost of service
- The quality of service
- Timely delivery of service
- Flexibility to work in different domains, under different rules

4 Multi-Agent Systems

In such networks because parties are owned by different stakeholders, in theory each rating would present a different stakeholder point of view. In addition, because of the size of such networks, it is not practically possible to calculate the trust of a party by searching through entire network. Therefore trust evaluating methods that are using composite techniques and sources (such as trustworthy group memberships, certificates) at the same time appears to be an effective solution for open multi-agent systems.

Conclusion

In this paper we discuss trust evaluation requirements of four major fields of computer applications; e-commerce, peer-to-peer networks and service oriented computing. We provided major examples of trust evaluation methods, we discussed their features and we categorised them. Then we listed our findings about trust requirements of different computer applications.

E-commerce networks are in need of centralised reputation systems, which would allow administrators to have control over the reputation information traffic. In addition such systems provide a higher availability and stability for the users who are constantly in need of reputation information. Such systems appear to be costly to run but if the solution can be kept simple like the eBay example in this review this cost can be limited.

Whereas P2P networks are in need of decentralized reputation systems, which would give them the required flexibility, they need. In service oriented systems it is hard to setup certain rules and expect all parties to follow them. In addition this area needs more detailed ranking levels (for example: to compare cost, timing, availability – out of 5), therefore binary ranking methods are not suitable.

Peer-to-peer file sharing networks are too big to control centrally. In addition, because of the number of transfers at a given time, it is highly probable to find online and available sources to provide reputation information about target peers. Decentralised systems appear to be more open to fraud attempts but reputation mechanisms have precautions such as credibility and references.

To conclude, it is not possible to have a trust evaluation method that would suit every type of computer network. Simple methods, such as eBay's, are far from answering specific requirements of other applications. Nevertheless, when the solution starts using different sources of information and becomes sophisticated, as a result it becomes more and more dependent on the specific variables of the application. For effective solutions, trust evaluation methods should have application-specific features to answer varying requirements.

Reference

1. H. Billhardt, R. Hermoso, S. Ossowski, and R. Centeno, (2007). `Trust-based service provider selection in open environments.` In *Proceedings of the 2007 ACM Symposium on Applied Computing* (Seoul, Korea, March 11 - 15, 2007). SAC '07. ACM, New York, NY, 1375-1380.

2. Cardiff University (2008), `Service Oriented Computing`, last viewed 1st May 2009, http://www.wesc.ac.uk/technology/soc/index.html

3. C. Dellarocas (2001). `Analyzing the economic efficiency of eBay-like online reputation reporting mechanisms'. In EC '01: Proceedings of the 3rd ACM conference on Electronic Commerce, pp. 171-179, New York, NY, USA. ACM Press.

4. Schahram Dustdar (2007), Service-Oriented Computing and Web Services, last viewed 24th April 2009, https://www.infosys.tuwien.ac.at/teaching/courses/IntAppl/1_SOA+WebServices.pdf

5. Tyrone Grandison, Morris Sloman (2001). `A Survey of Trust in Internet Applications'. IEEE Communications Surveys and Tutorials, Fourth Quarter 2000 :3-4

6. Nathan Griffiths (2006). `Enhancing peer-to-peer collaboration using trust.` Expert Systems with Applications :849–858

7. Nathan Griffiths, Kuo-Ming Chao, and Muhammad Younas. (2006). `Fuzzy Trust for Peer-to-Peer Systems.` In Proceedings of the 26th IEEE international Conference workshops on Distributed Computing Systems (July 04 - 07, 2006). ICDCSW. IEEE Computer Society, Washington

8. T.D. Huynh, N.R. Jennings, and N.R. Shadbolt 2006. `An integrated trust and reputation model for open multi-agent systems.` Autonomous Agents and Multi-Agent Systems 13, 2 (Sep. 2006)

9. How feedback works, eBay, viewed 21st March 2009, http://pages.ebay.com/help/feedback/howitworks.html

10. S. D. Kamvar, et al. (2003). `The Eigentrust algorithm for reputation management in P2P networks'. In WWW '03: Proceedings of the 12th international conference on World Wide Web, pp. 640-651, New York, NY, USA. ACM Press.

11. P Resnick, R Zeckhauser, J Swanson, K Lockwood (2003). `The Value of Reputation on eBay: A Controlled Experiment'.

12. W.T. Tsai and Yinong Chen (2007) `Introduction to Service-Oriented Computing`, last viewed 2nd April 2009, http://www.public.asu.edu/~ychen10/activities/SOAWorkshop/Background.pdf

13. Le-Hung Vu, M. Hauswirth , K. Aberer (2005). `QoS-based Service Selection and Ranking with Trust and Reputation Management`, 13th International Conference on Cooperative Information Systems, Agia Napa, Cyprus.

14. Yan Wang, Kwei-Jay Lin, Duncan S. Wong, Vijay Varadharajan. (2008). `Trust management towards service-oriented applications'. Service Oriented Computing and Applications.

15. Yan Wang and Vijay Varadharajan. (2005). `Trust2: Developing Trust in Peer-to-Peer Environments.` In Proceedings of the 2005 IEEE international Conference on Services Computing - Volume 01 (July 11 - 15, 2005). IEEE Computer Society, Washington

16. Yan Wang and Vijay Varadharajan. (2004) `Interaction Trust Evaluation in Decentralized Environments`. E-Commerce and Web Technologies :144-153. Springer Berlin, Heidelberg

17. L. Xiong & L. Liu (2004). `PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities'. IEEE Transaction On Knowledge and Data Engineering 16(7):843-857.

18. Bin Yu, Munindar P. Singh, Katia Sycara. (2004). `Developing Trust in Large-Scale Peer-to-Peer Systems'. In 1st IEEE International Conference on Mobile Ad-hoc and Sensor Systems.

19. G. Zacharia & P. Maes (2000). `Trust Management Through Reputation Mechanisms'. Applied Artificial Intelligence 14(9):881-907.

20. L. A. Zadeh (1965). `Fuzzy sets'. Information and Control 8(3):338-353.