Modeling Trust for Users and Agents in Ubiquitous Computing

Sebastian Ries*, Jussi Kangasharju, and Max Mühlhäuser

Department of Computer Science, Darmstadt University of Technology,
Hochschulstrasse 10, 64289 Darmstadt, Germany,
{ries, jussi, max}@tk.informatik.tu-darmstadt.de

Abstract Finding reliable partners for interactions is one of the challenges in ubiquitous computing and P2P systems. We believe, that this problem can be solved by assigning trust values to entities and allowing them to state opinions about the trustworthiness of others. In this paper, we introduce our vision of trust-aided computing, and we present a trust model, called CertainTrust, which can easily be interpreted and adjusted by users and software agents. A key feature of CertainTrust is that it is capable of expressing the certainty of a trust opinion depending on the context of use. We show how trust can be expressed using different representations (one for users and one for software agents) and present an automatic mapping to change between the representations.

1 Introduction

In [1], Bhargava et al. point out that "trust [...] is pervasive in social systems" and that "socially based paradigms will play a big role in pervasive-computing environments". Pervasive or ubiquitous computing is characterized by a very large number of smart devices, e.g., PDAs, mobiles, intelligent clothes etc., which come with different communication capabilities, storage, or battery power. Both the basic idea of ubiquitous computing and the heterogeneity of these devices call for interaction with and delegation to other devices.

Ubiquitous computing environments are unstructured and many service providers are available only locally or spontaneously. On the one hand, the interactions with foreign devices include uncertainty and risk, since a safe prediction of the behavior of those devices is not possible. On the other hand, the interactions with reliable partners are the basis for the services ubiquitous computing environments can provide. But how to select reliable interaction partners who behave as expected? Selecting only tamper-proof devices, which belong to the same manufacturer, requires the manufactures to be trusted, and unnecessarily reduces the potential of ubiquitous computing. Due to the great number of interactions with many different partners – some well-known, others not – and

* The author's work was supported by the German National Science Foundation (DFG) as part of the PhD program "Enabling Technologies for Electronic Commerce" at Darmstadt University of Technology

the claim of ubiquitous computing of being a calm technology, we need a non-intrusive way to cope with this challenge. We believe that the concept of trust, which has shown to work well in real life, is a promising solution, since it allows to make well-founded decisions in context of risk and uncertainty. Assuming recognition of entities, trust allows to express an expectation about the future behavior of an entity based on evidence from past engagements. Furthermore, trust needs representations which are meaningful not only to the software agents to enable automatic trust evaluation, but also to the end user, who needs to be able to understand the state of the trust model and to take part in the decision making process, if necessary.

In this paper, we provide a decentralized trust model, named *CertainTrust*, which allows agents to choose trustworthy partners for risky engagements. For our trust model, we propose two representations. The first one serves as a basis for a human trust interface. It represents trust using two independent parameters consisting of an estimate for the probability of trustworthy behavior in a future engagement, and of a parameter expressing the certainty of this estimate. Since we believe that trust is context-dependent, we also enforce the context-dependency of the certainty parameter of a trust value. The second representation is based on the Bayesian approach using beta probability density functions. This approach is well-established to express trust. It serves as a basis for the trust computation and as an interface for evidence-based feedback integration. Finally, we provide a mapping between both representations, and operators for 'consensus' and 'discounting'.

The remainder of this paper is structured as follows. In Sect. 2, we summarize our notion of trust and introduce our concept for the integration of trust in applications. Sect. 3 presents the trust model and the operators for trust propagation. In Sect. 4, we give an example showing how trust is represented and calculated using CertainTrust. Sect. 5 presents a summary of the related work, and Sect. 6 summarizes our contribution and outlines aspects of our future work.

2 Our Notion of Trust

From our point of view, trust is the well-founded willingness for a potentially risky engagement. Trust can be based on direct experience, recommendations, and the reputation assigned to the partner involved in an engagement. We model trust as the subjective probability that an entity behaves as expected based on experiences from past engagements.

2.1 Trust-Aided Computing

The integration of trust in ubiquitous computing applications seems to be a promising concept to cope with the new challenges of unstructured environments and dynamically changing interaction partners. In our vision of trust-aided computing (TAC) the applications are enabled to explicitly reason about trust. TAC

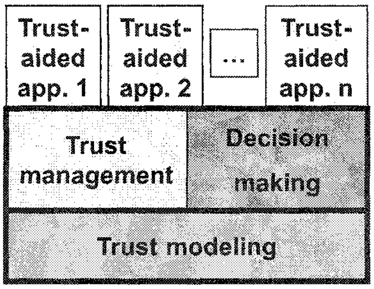

Figure 1. Trust-aided computing architecture

keeps track of the available entities, it collects information about direct experiences with other entities and information as recommendations and reputation. Thus, TAC allows applications to adapt to changes in the infrastructure and keep user preferences for interaction with entities which are already trusted. Not only relying on direct experience, but also on recommendations and reputation information, allows for building trust in entities with which no or only little direct experience is available. Since trust is subjective, this allows not only automated, but personalized decision making.

TAC disburdens the user from constantly being asked for the same decision in the same context. Furthermore, it disburdens the application developers from providing hard-coded strategies, which allow automated decision making, but cannot or can only hardly be personalized and do not take advantage of information collected in past engagements.

Trust-aided computing can be integrated in applications like P2P file sharing or packet routing in ad hoc networks to cope with malicious nodes. For ubiquitous computing trust-aided computing allows to find trustworthy devices in smart environments. On a higher level of abstraction trust-aided applications can help user to (semi-)automatically recognize trustworthy partners in virtual communities, e.g., to find reliable sellers in online auction platforms, or to evaluate recommendations provided by members of social networks (e.g., FilmTrust [4]). The trust-aided computing architecture has four main components (see Fig. 1):

- The *trust-aided applications* are able to reason about trust in a uniform way. They support the users since TAC allows to autonomously select trustworthy interaction partners. Yet, in critical cases, which can be identified, when reasoning about trust and risk, the user can be informed about the state of the system and asked for interaction.
- The *trust management component* focuses on the collection and filtering of evidence. Therefore, it is necessary to monitor the devices which are available, and to collect evidence from those devices. Furthermore, it is necessary to filter the collected evidences and recommendations based on policies to increase the level of attack-resistance. Other aspects of trust management are the evaluation of the context and the risk, which is associated to an engagement. Current trust management approaches are usually based on policies,

in which users can state which entities they consider as trustworthy. The approaches in [3, 5] already allow for the integration of trust levels.
- The *trust modeling component* concentrates on the reasoning about trustworthiness based on the available evidence. It provides the representational and computational models of trust. Since the users need to be able to set up, control and adjust the trust parameters, there is the need for a user interface, which can be used intuitively. Furthermore, there is the necessity for an interface, which is suitable for software agents allowing for automated integration of feedback and for autonomous evaluation of trust-relevant information. The computational model defines the aggregation of recommendations, reputation information and direct experience to a single opinion. For highly dynamic environments, which may contain a very large number of entities, e.g., ubiquitous computing environments, we cannot expect the end user to set up policies for each entity and for each context. Thus, the computational trust model itself has to provide some robustness towards attacks.
- The *decision making component* is often treated as a part of trust management. Since it is a very important aspect, and the users probably will judge the performance of a trust-based system, in the quality of its decisions, we like to mention it separately. The decision making has to consider the collected information about trust as well as the information about the expected risk. Although, we like to automate the decision process as far as possible, there must be the possibility of user interaction, to support the user to get used to TAC, and to be able to interact with the system in critical cases.

2.2 Properties of Trust

Having introduced an architecture which shows how to integrate trust in software applications, we focus on trust modeling for the rest of the paper. Our model expresses the following properties of trust: Trust is subjective, i.e., the trust of an agent A in an agent C does not need to be the same as trust of any other agent B in C. Furthermore, we cannot expect the behavior of A towards C to be the same as the behavior of C towards A, thus trust is asymmetric. Trust is context-dependent. Obviously, there is a difference in trusting in another agent as provider of mp3-files or as provider of an online banking service. It also is a difference in trusting in someone as service provider or as recommendation provider. If A trusts B in the context of providing recommendations about a good service provider, e.g., for file-storing, this does not necessarily imply that A trusts in B as a good peer to store files at, and vice versa. Trust is non-monotonic, i.e., experience can increase as well as decrease trust. Thus, we need to model both positive and negative evidence. Trust is not transitive in a mathematical sense, but the concept of recommendations is very important. Recommendations are necessary to introduce trust in agents with which no or only little direct experience is available. Moreover, we do not think of trust as finite resource, e.g., as done in flow-based approaches like EigenTrust [9]. It should be possible to increase trust in one entity without decreasing trust in another one.

2.3 Trust & Certainty

As in [6,10,12], we believe that it is necessary to express the (un-)certainty or reliability of an opinion stating trustworthiness. We also believe that the certainty of an opinion increases with the number of evidence, on which an opinion is based. Modeling the certainty of an opinion allows us to provide information on how much evidence an opinion is based, or to state that there is not any evidence available. Furthermore, it is possible to express that one opinion might be supported by more evidence than another one. We believe that the certainty value needs to be context-dependent because of the following reasons.

- In ubiquitous computing environments trust models can be used to automate decision making in many different contexts. In some contexts, there might be a great number of encounters, other contexts might be related to high risk, considering legal or financial implications. In these contexts, it seems reasonable, that users want to collect a great number of evidence, before they would think about an opinion to be certain. If forced to make a decision about an engagement involving high risk, one might choose to reject the engagement, although there is positive but too little evidence.
- In contexts in which the number of encounters is lower, or the associated risk is lower, users may be satisfied with a lower number of evidence to come to a well-founded decision.

To model the context-dependency for the certainty of an opinion, we assume there is a *maximal number of expected evidence* per context, which corresponds to the maximal level of certainty. For example, the maximal number of expected evidence can be defined as 5, 10, 100, or 1000.

3 Trust Model - CertainTrust

Let the contexts be denoted by con_i $i \in \{1, 2, \dots\}$, e.g., $con_1 = file\_sharing$. For providing recommendations for a context con_i, we define a special context, which is denoted as rec_i. Let agents be denoted by capital letters $A, B, \dots$, and propositions by small letters $x, y, \dots$. The opinion of agent A about the truth of a proposition x in context con_i is denoted as $o_x^A(con_i)$. For example, in Fig. 2 the proposition x can be interpreted as $x =$ "Agent C behaves trustworthily in context con_i". The opinion of agent A about B's trustworthiness for providing recommendations for a context con_i is denoted as $o_B^A(rec_i)$. If the context is clear or not relevant, we use o_x^A and o_B^A. The maximal number of expected evidence (see Sect. 2.3) is denoted as $e(con_i)$ or e. Since the evidence model is partly derived from ideas presented in [6], we use the same terminology when possible.

We assume that evidence is collected and stored locally, and that recommendations are provided on request. The propagation of recommendations is done based on chains of recommendations, as shown in Fig. 2. We propose special operators for consensus (aggregation of opinions) and discounting (weighting of recommendations). For simplicity, opinions are assumed to be independent.

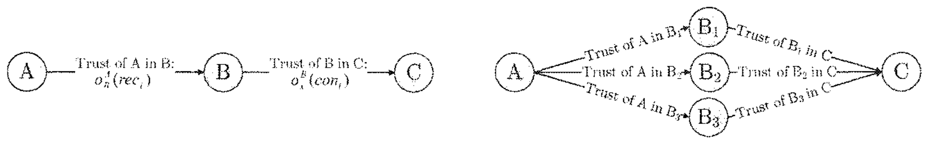

Figure 2. Trust chains

Our model provides two representations for opinions to express trust. The first representation is a pair of *trust value* and *certainty value* which serves as a base for a human trust interface. The second representation is based on the number of collected evidence and allows us to easily integrate feedback and forms the base of the computational model.

3.1 Human Trust Interface

The *human trust interface* (HTI) is used to represent trust as opinions. In the HTI an opinion o is a 2-tuple $o = (t, c)^{HTI} \in [0, 1] \times [0, 1]$, where HTI refers to the representational model. The opinion $o_B^A(rec_i) = (t_B^A(rec_i), c_B^A(rec_i))^{HTI}$ expresses the opinion of A about the trustworthiness of B in the context rec_i. The value of $t_B^A(rec_i)$ represents the probability that A considers the proposition "I believe, B to be trustworthy for providing recommendations for the context con_i" to be true. This value is the *trust value*. The value $c_B^A(rec_i)$ is the *certainty* or *certainty value*. This value expresses, which certainty the provider of an opinion assigns to the trust value. A low certainty value expresses that the trust value can easily change, and a high certainty expresses that the trust value is rather fixed. The values for trust and certainty can be assigned independently of each other. For example, an opinion $o_B^A(rec_i) = (1, 0.1)^{HTI}$ states that A expects B to be trustworthy in providing recommendations for con_i, but that A is not at all certain about this opinion.

For the moment, we express both the values for trust and certainty as continuous values in $[0, 1]$. Since humans are better in assigning discrete (verbal) values than continuous ones, as stated in [4,8], we want to point out, that both values can easily be mapped to a discrete set of values, e.g., to the natural numbers in $[0, 10]$, or to set of labels, as "very trusted" (vt), "trusted" (t), "undecided" (ud), "untrusted" (u), and "very untrusted" (vu). We assume that the trust and certainty values are independent, since the HTI then allows the users to easily express and interpret an opinion based on labels (see Fig. 3 (right side)). This is important since it allows users to check the current state of the model, and to set up and adjust those values according to personal preferences.

3.2 Evidence Model

The second representation, the evidence model, is based on beta probability density functions (pdf). The beta distribution $Beta(\alpha, \beta)$ can be used to model the posteriori probabilities of binary events. The beta pdf is defined by:

$$f(p \mid \alpha, \beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} p^{\alpha-1}(1-p)^{\beta-1} \ , \\ \text{where } 0 \le p \le 1, \alpha > 0, \beta > 0 \ . \tag{1}$$

Furthermore, we use $r = \alpha + 1$ and $s = \beta + 1$, where $r \geq 0$ and $s \geq 0$ represent the number of positive and negative evidence, respectively. The *number of collected evidence* is represented by $r + s$.

In the evidence model, an opinion o can be modeled using the parameters α and β. We denote this representation as $o = (\alpha, \beta)^{\alpha\beta}$. If the opinion is represented by the parameters r and s, we use the notation $o = (r, s)^{rs}$.

For $r + s \neq 0$ the mode t of the distribution $Beta(\alpha, \beta)$ is given as:

$$t = mode(\alpha, \beta) = \frac{\alpha - 1}{\alpha + \beta - 2} = \frac{r}{r+s} \tag{2}$$

For any $c \in \mathbb{R} \setminus \{0\}$ holds

$$mode((r,s)^{rs}) = mode((c \cdot r, c \cdot s)^{rs}) \ . \tag{3}$$

The main feature of this model is the easy integration of feedback in the trust model. Assuming that feedback fb can be expressed as real number in $[-1;1]$, where '-1'is a negative experience and '1' is positive, the update of an opinion is done by recalculating the parameters $r_{new} = r_{old} + 0.5 * (1 + fb)$ and $s_{new} = s_{old} + 0.5 * (1 - fb)$ (cf. [7]). If the feedback is generated automatically, we can update the trust model without user interaction. Furthermore, the software agents can use all statistical information from the beta distribution, e.g., mean value and variance, as basis for decision making.

3.3 Mapping Between Both Representations

Trust value t of an opinion $o = (\alpha, \beta)^{\alpha\beta}$ is defined as the mode of the corresponding beta distribution. The certainty value c of an opinion $o = (\alpha, \beta)^{\alpha\beta}$ in a context con_i is defined as follows: The maximal number of expected evidence can be denoted by $e(con_i) = \alpha_{max} + \beta_{max} - 2$, where α_{max} and β_{max} fulfill:

$$mean_{coll} := \frac{\alpha}{\alpha + \beta} = \frac{\alpha_{max}}{\alpha_{max} + \beta_{max}} =: mean_{max} \tag{4}$$

Then the certainty c is calculated as:

$$c = \frac{f(mean_{coll} \mid \alpha, \beta) - 1}{f(mean_{max} \mid \alpha_{max}, \beta_{max}) - 1} \tag{5}$$

Definition 1 (Mapping) *$(\alpha, \beta)^{\alpha\beta} = (t, c)^{HTI}$, iff $t = mode(\alpha, \beta)$ and the certainty c fulfills Eq. 5.*

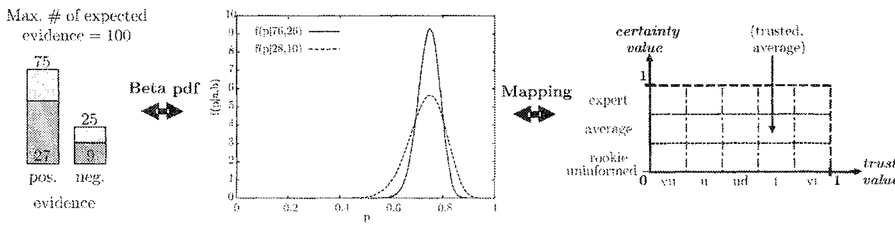

Figure 3. Mapping: Evidence - Agent Interface (pdf) - User Interface (HTI)

This mapping provides the translation between both representations (see Fig. 3). The interpretation of an opinion in the HTI by users has to be as close as possible to the interpretation of the same opinion in the evidence model by a software agent. This way, a user can interpret and adjust opinions based on evidence collected by a software agent and vice versa. Fig. 3 shows an opinion which is based on 27 positive and 9 negative pieces of evidence. The maximal number of expected evidence is 100. The mapping between the collected evidence and the agent interface is done using beta density functions. The mapping between the agent interface and the user interface is the one defined by Def. 1.

Intuitively, a human would set the trust value close to the observed relative frequency. Since the mode of a pdf is equal to the relative frequency of the observed event, the trust value t is close to the intuitive value set by the user. When no evidence is available, the mode of the pdf is not defined. In this case it can be reasonable to assume either a trust value of 0.5, or to infer the trust value based on the past experiences with formerly unknown entities.

The certainty value is intuitively linked to the number of collected evidence [6, 10,12]. A greater number of collected evidence leads to higher confidence, and to a higher certainty value. The maximal number of expected evidence $e(con_i)$ (see Sect. 2.3) is the maximal certainty value. Similar to [12], we want the certainty to increase adaptively with the number of collected evidence, i. e., the first pieces of evidence increase the certainty value more than later ones. As shown in Fig. 4, our certainty value fulfills these properties. In the absence of information ($r+s=0$), the certainty value is $c=0$, and $c=1$ if the number of collected evidence is equal to the expected number of evidence. Between the two extremes, the certainty value increases adaptively. If the number of collected evidence is greater than the number of expected evidence, there is a normalization, which preserves the trust value and scales the certainty to $c=1$ (see Eq. 6).

Normalization If an opinion $o=(r,s)^{rs}$ is based on more than the maximal number of expected evidence, it will be scaled to this maximum. The normalization preserves the mode of the pdf (see Eq. 3), and therefore, does not change the trust value. The normalized opinion $norm(o)$ will be used as input for the discounting described above.

$$norm((r,s)^{rs}) = \begin{cases} (r,s)^{rs} & \text{if } r+s \leq e \ , \\ (\frac{r}{r+s} \cdot e, \frac{s}{r+s} \cdot e)^{rs} & \text{else } . \end{cases} \tag{6}$$

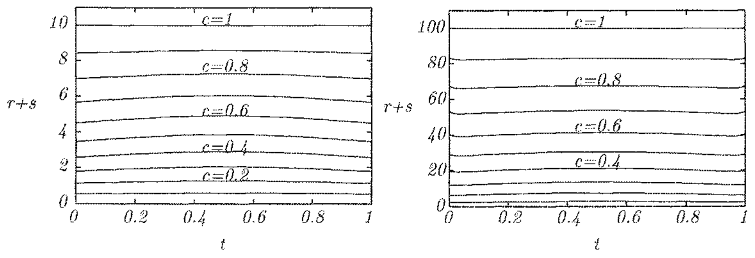

Figure 4. Iso-certainty lines: max. no. of exp. evidence $e = 10$ (l), $e = 100$ (r)

3.4 Trust Propagation

For trust propagation we define two operators, similar to the ones defined by Jøsang in [6]. We also call our operators 'consensus' for the aggregation of opinions and 'discounting' for the recommendation of an opinion. The consensus operator is identical with the one presented in [6]. We define a new discounting operator based on our evidence model.

Definition 2 (Consensus) *Let $o_x^A = (r_x^A, s_x^A)^{rs}$ and $o_x^B = (r_x^B, s_x^B)^{rs}$ be the opinions of A and B about truth of the proposition x. The opinion $o_x^{A,B} = (r_x^{A,B}, s_x^{A,B})^{rs}$ which combined the experiences of A and B, is defined as:*

$$o_x^{A,B} = o_x^A \oplus o_x^B = (r_x^A + r_x^B, s_x^A + s_x^B)^{rs} \tag{7}$$

The '$\oplus$' symbol denotes the consensus operator. The operator can easily be extended for the consensus between multiple opinions.

Definition 3 (Discounting) *Let $o_B^A = (r_B^A, s_B^A)^{rs}$ and $o_x^B = (r_x^B, s_x^B)^{rs}$. We denote the opinion of A about x based on the recommendation of B as o_x^{AB} and define it as:*

$$o_x^{AB} = o_B^A \otimes o_x^B = (d_B^A r_x^B, d_B^A s_x^B)^{rs} \text{ , where } d_B^A = t_B^A c_B^A \ . \tag{8}$$

The '$\otimes$' symbol denotes the discounting operator. In a chain of recommendations, we start at the end of the chain, e.g., $o_x^{ABC} = o_B^A \otimes (o_C^B \otimes o_x^C)$.

Discounting reduces the number of evidence taken into account, since $d_B^A \in [0; 1]$. The discounting factor d_B^A increases with positive evidence. That is, if A and C have the same amount of total evidence with B, but A has more positive evidence, then A gives a stronger weight to the recommendation of B than C.

Furthermore, the discounting factor increases with the number of collected evidence. That is, if A and C have the same ratio of positive and negative evidence with B, but A has more evidence in total, then A gives the opinion of B a stronger weight than C does.

Opinion	HTI	rs	Interpretation
A's opinions about B_i as recommenders			
$o^A_{B_1}$	(1,1)	(100,0)	(vt,expert)
$o^A_{B_2}$	(0.7,0.5)	(21.9,9.04)	(t,average)
$o^A_{B_3}$	(0.5,0.2)	(3.80,3.80)	(ud,rookie)
B_i's opinion about C as service provider			
$o^{B_1}_x$	(1,0.5)	(15.02,0)	(vt,average)
$o^{B_2}_x$	(0.7,0.5)	(11.19,4.80)	(t,average)
$o^{B_3}_x$	(0,0.5)	(0,15.02)	(vu, average)
Discounted opinions			
$o^{AB_1}_x$	(1,0.5)	(15.02,0)	(vt,average)
$o^{AB_2}_x$	(0.7,0.24)	(3.90,1.68)	(t,rookie)
$o^{AB_3}_x$	(0,0.10)	(0,1.50)	(vu,rookie)
Consensus: A's opinion about C as service provider			
o^{AB_1,AB_2,AB_3}_x	(0.86,0.62)	(18.92,3.18)	(vt,average)

Table 1. Example: Trust calculation

4 Example

Consider a file sharing scenario, where peers offer files for others to download. The risk in a file download is that the file might be corrupted or contain viruses. A simple corrupted file carries only a small risk, since we have only lost some CPU cycles and bandwidth, but a virus could potentially be extremely damaging. The trust is based on direct experience (i.e., downloads) and recommendations from other peers. In this example, we assume high risk and a high frequency of encounters. Therefore, we choose $e(con_i) = 50$ for the context of providing files, and $e(rec_i) = 100$ for the contexts of providing recommendations.

Peer A wants to download a file from peer C. Peer A has no direct experience with C, but A receives 3 recommendations about C's behavior (see right side of Fig. 2). Table 1 shows the collected and the calculated opinions.

This example shows how trust is represented and calculated in our model. Comparing the opinions $o^{B_i}_x$ and $o^{AB_i}_x$ shows that the discounting operation only manipulates the certainty values and preserves trust values. The consensus operation increases the certainty and adapts the trust value according to the given opinions. The resulting opinion o^{AB_1,AB_2,AB_3}_x seems to be reasonable, when considering the input. Especially, we can see that the recommendation by B_3, which states C should be considered as "very untrustworthy", has only little impact on the resulting opinion, although all peers B_i claim to have a similar amount of experience with C. The reason is that the opinion $o^A_{B_3}$, stating that A has only little experience with B_3 with varying results, corresponds to lower discounting factor than the one of $o^A_{B_1}$ and $o^A_{B_2}$.

If A decides to download the file from C, then A can generate some feedback information based on the file's quality. This information can be used in an opinion as direct experience with C and to update the trust in the recommendations of the B_i's. If additional reputation information was available, it would have been integrated and discounted in the same way as the opinion of an agent B_i.

5 Related Work

The modeling of trust is addressed by a growing group of researchers [11]. There are several approaches which try to model trust one-dimensionally, e.g., Tidal-Trust [4] and EigenTrust [9]. A single trust value does not allow to express the certainty or the reliability of this trust value. Thus, it is impossible to express if an opinion is based on single evidence or on multiple pieces of evidence.

Other approaches model trust with two or three dimensions. Two dimensional trust models are often based on the Bayesian approach, e.g., [6, 10]. As stated in [8], those models are often too complicated to be understood by average users. The belief model approaches, e.g., [2, 6], use the triple belief b, disbelief d, and uncertainty u to represent trust. The problem with such models is that the three parameters cannot be assigned independently (e.g., $b + d + u = 1$ [6]) and hence the presence of uncertainty influences both belief and disbelief. Therefore, it is non-trivial to express, e.g., a medium belief with different levels of uncertainty. Our model allows to independently choose the values for trust and certainty.

Approaches like Subjective Logic [6] are not capable of expressing uncertainty context-dependently. In [6], the uncertainty u is defined as $u = 2/(r + s + 2)$. Therefore, uncertainty depends only on the number of collected evidence, but not on the context.

Other approaches presented in [10,12] introduce *reliability* as a concept which is similar to our concept of *certainty*. They also define a context-dependent value similar to the maximal number of expected evidence.

The model in [10] is based on the Bayesian approach. The maximal number of expected evidence e corresponds to m, which is described as the "minimal number of encounters necessary to achieve a given level of confidence". The reliability w is defined to increase linearly with the number of collected evidence from 0 (no evidence) to 1 (collected evidence at least m). But this linear approach is stated to be an first order approximation.

In the model in [12], the *intimate* level of interactions, is close to the concept of a maximal number of expected evidence. The *number of outcomes factor* (No) $\in [0, 1]$, increases with the number of collected evidence. To achieve the adaptive behavior as described in 3.3, the ratio between the collected and the expected number of evidence is used in a *sinus*-function.

6 Conclusion and Future Work

In this paper we introduced our vision of trust-aided computing, which allows for an explicit integration of trust information in applications. Furthermore, we provided a trust model, which allows to represent trust in a way, which can be interpreted and updated by software agents as well as by users.

We have shown, how to express the certainty of an opinion in contexts which are associated with different levels of risk, or frequency in interaction. In the HTI the values for trust and certainty can be interpreted independently, which allows to introduce the semantics of an opinion based on labels. Therefore, the user is able to easily control the state of the trust model and to adjust opinions, if necessary. The evidence model enables software agents to update an opinion, when new evidence is available, and to reason about the trustworthiness of an interaction partner. The mapping between the HTI and the evidence model has an intuitive interpretation and it is mathematically founded. The two operators for trust propagation are based on the evidence model.

Our future work will include the development of trust management and decision making strategies. Those are necessary to be able to evaluate the trust model in a simulation, and to enhance the attack-resistance of the model. By first selecting recommendations from entities, which have repeatedly shown to provide good recommendations (high trust value and high certainty), and limiting the number of collected evidence to the maximal expected number of evidence, we believe to reduce the impact of sybil attacks to our trust model. Furthermore, we will refine the discrete representation for the human trust interface.

References

1. B. Bhargava, L. Lilien, A. Rosenthal, and M. Winslet. Pervasive Trust. *IEEE Intelligent Systems*, 19(5):74–88, 2004.
2. V. Cahill et al. Using Trust for Secure Collaboration in Uncertain Environments. *IEEE Pervasive Computing*, 2/3:52–61, July 2003.
3. M. Carbone, M. Nielsen, and V. Sassone. A Formal Model for Trust in Dynamic Networks. In *Proc. of IEEE Int. Conf. on Software Engineering and Formal Methods*, Brisbane, Australia, September 2003. IEEE Computer Society.
4. J. Golbeck. *Computing and Applying Trust in Web-Based Social Networks.* PhD thesis, University of Maryland, College Park, 2005.
5. T. Grandison. *Trust Management for Internet Applications.* PhD thesis, Imperial College London, 2003.
6. A. Jøsang. A Logic for Uncertain Probabilities. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 9(3):279–212, 2001.
7. A. Jøsang and R. Ismail. The Beta Reputation System. In *Proceedings of the 15th Bled Conf. on Electronic Commerce*, June 2002.
8. A. Jøsang, R. Ismail, and C. Boyd. A Survey of Trust and Reputation Systems for Online Service Provision. In *Decision Support Systems*, 2005.
9. S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The Eigentrust Algorithm for Reputation Management in P2P Networks. In *Proc. of the 12th Int. Conf. on World Wide Web*, pages 640–651, New York, USA, 2003. ACM Press.
10. L. Mui et al. A Computational Model of Trust and Reputation for e-Businesses. In *Proceedings of the 35th HICSS*, 2002. IEEE Computer Society.
11. S. Ries, J. Kangasharju, and M. Mühlhäuser. A Classification of Trust Systems. In *On the Move to Meaningful Internet Systems 2006: OTM Workshops*, pages 894 – 903, Montpellier, France, 2006.
12. J. Sabater and C. Sierra. Reputation and Social Network Analysis in Multi-Agent Systems. In *Proceedings of the 1st Int. Joint Conf. on Autonomous Agents and Multiagent Systems*, pages 475–482, New York, NY, USA, 2002. ACM Press.