# Implementing a Trust Overlay Framework for Digital Ecosystems

Paul Malone, Jimmy McGibney, Dmitri Botvich, and Mark McLaughlin

Waterford Institute of Technology, Waterford, Ireland
{pmalone,jmcgibney,dbotvich,mmclaughlin}@tssg.org
http://www.tssg.org

**Abstract.** Digital Ecosystems, being decentralised in nature, are inherently untrustworthy environments. This is due to the fact that these environments lack a centralised gatekeeper and identity provider. In order for businesses to operate in these environments there is a need for security measures to support accountability and traceability. This paper describes a trust overlay network developed in the OPAALS project[1] to allow entities participating in digital ecosystems to share experience through the exchange of trust values and to leverage on this network to determine reputation based trustworthiness of unknown and initially untrusted entities. An overlay network is described together with sample algorithms and a discussion on implementation.

**Keywords:** Trust, digital ecosystem, security.

## 1 Introduction

In the real world, when people interact with each other and provide and consume services, well-evolved social and commercial standards and structures help to provide assurance of orderly behaviour. Our senses are well tuned to the subtleties of real personal contact. Furthermore, in dealing with service providers like banks or shops, physical structures and contact help to convince us that we are really dealing with the service provider (and not an impostor), that the service provider appears to be backed by some assets and will still be there into the future, that the transaction is genuine, that the communication is confidential, and so on. Previous experience is also a major factor for example, a bank is more likely to extend a loan to a customer with a reliable track record. Previous good experience with a service provider assures a consumer of the quality of future transactions. Laws of the land provide penalties that reduce the incentive to behave dishonestly.

Reference is also made to third parties where appropriate. Credit card and other financial transactions need to be correctly authenticated and authorised. Various respected bodies issue credentials to people to help with verification of identification, nationality, permission to drive a car, access to restricted locations, and so on. Service providers may also be certified as to professional

[1] OPAALS, FP6 Network of Excellence, http://oks.opaals.org/website/

F.A. Basile Colugnati et al. (Eds.): OPAALS 2010, LNICST 67, pp. 178–191, 2010.

competence, adherence to health standards, and so on. More fuzzy personal recommendations and reviews are also of value.

The digital world provides opportunities for similar (and in some ways more sophisticated) social and commercial interaction. A great benefit of the Internet is its openness and lack of centralised control. Anyone can provide a service with the minimum of fuss and invite others to make use of it. As well as in the strictly Internet world, mobile telecommunications networks are facilitating more open service provision and consumption. The increasing proliferation of wireless and ad-hoc networks using unlicensed radio spectrum is serving to further loosen control.

This digital world presents a wide variety of risks. Anyone can set up a web site that looks just like your banks site and use various tricks to get you there; online communications can be eavesdropped upon, or even modified; someone else may impersonate you; incorrect or misleading information may be provided. Legal protection is made difficult by the inter-jurisdictional nature of these networks. A major factor is that we can no longer rely on physical structures, social skills or intuition to provide assurance of security, and thus there is a much greater need for reference to third parties for identification of entities and verification of credentials.

Trust is a basic requirement for these interactions. A successful online communication or transaction requires that the parties involved sufficiently trust each other and the infrastructure that connects them. Consider for example the case where I use a web browser to navigate to an online retailers site to purchase a product or service. For this to work, I need to trust several entities: that my computer hardware and software is acting in my interest (trojaned web browser software could direct me to a bogus location); that the website is actually that of the vendor; that the vendor and any other parties collaborating with the vendor (e.g. for payments) will act responsibly with my personal information; that the vendor will deliver the product or service as expected. I may also need to trust, to some extent at least, other parts of the infrastructure. Where there is no strong means to authenticate the other party (e.g. an email correspondent or a non-secured website), I may need to trust that the domain name system (DNS) directs me to the correct IP address and that my Internet service provider and other infrastructure providers correctly route data packets.

The current approach to trust establishment and management on computer networks works in some situations, to some extent, but has significant weaknesses that limit its potential, especially in enabling rich peer-to-peer interactions and transactions.

Note that the value of building trust is not limited to commercial interaction. Even if we consider knowledge that is shared for free and without restriction, there are still threats, which can be reduced by having measures of trust, such as:

– The knowledge provided could be deliberately false. For example, free software could contain a Trojan horse or other malicious code.

– The knowledge provided could be erroneous or subject to misinterpretation, due to limitations of its creator or editor.
– The consumer of the knowledge could waste a lot of time on a facet of low value. It takes time to read a document. Software takes time to install. There may also be a learning curve, meaning that significant time and energy needs to be invested in its adoption, making it painful to rollback if it turns out not to be fit for its intended purpose.
– Some knowledge may be undesirable and unwanted, especially if the consumer does not specifically solicit or request it. Spam is an example of bad 'knowledge'.

## 2 The Meaning of Trust

The social concept of trust is very complex and sophisticated, perhaps deceptively so. Trust is closely related to many other social concepts such as belief, confidence, dependence, risk, motivation, intention, competence and reliability [1]. It is also interwoven with the areas of accountability[2] and identity[3].

Trust management has become a well-studied area in recent years, and several recent surveys summarise the state of the art[6][11][7][5]. Much of the work on trust in the computer science domain attempts to provide computational measures of trust so that it can be used in making decisions related to electronic transactions, in a sense mimicking peoples well-evolved forms of social interaction.

One of the difficulties of modelling trust computationally is that social trust is based on quite an intuitive and personalised subjective assessment. As trust is quite an overloaded term, most attempts to model trust start by defining what is meant by trust (at least for the purposes of that particular model). Thus there are currently several different definitions and interpretations of what trust means and how it can be used.

Trust refers to a unidirectional relationship between a trustor and a trustee. A trustor is an autonomous entity that is capable of making an assessment. A trustee can be anything. A frequently cited definition of trust is by Gambetta, a sociologist, as [4]:
*"a particular level of the subjective probability with which an agent assesses that another agent or group of agents will perform a particular action, both before he can monitor such action and in a context in which it affects his own action"*

This definition formulates trust as:

1. a probabilistic measure; i.e. implying that trust can be modelled as a single value between zero (complete distrust) and one (complete trust).
2. defined by the subject; each trustor may have a different view of the same trustee
3. relating to a particular action i.e. a particular service offered by the trustor; you would trust your bank more for financial advise and your doctor more for medical advice.

4. an a priori measure; trust relates to incomplete information and is thus an estimate.
5. relating to context; trust depends on the viewpoint of the trustor on how it might affect his or her action

Jøsang et al. in [6] adapt the work of McKnight and Chervany [9] to define decision trust as:
*"the extent to which one party is willing to depend on something or somebody in a given situation with a feeling of relative security, even though negative consequences are possible"*

In some cases, "to trust" is taken to mean making the decision itself. "I trust you" means that I believe that you will act in a particular way. Trustworthiness is a term closely related to trust, and sometimes used interchangeably with it. Solhaug et al. [12] have made a useful distinction between trust and trustworthiness, by accepting Gambettas definition of trust as a subjective probability and defining trustworthiness as an objective value:
*"the objective probability by which the trustee performs a given action on which the welfare of the trustor depends"*

Reputation is also related to trust, though differing views exist on precisely how they are linked. A common view is that reputation is one of the sources used by a trustor in making a trust assessment. Other sources of information for a trustor, besides reputation, typically include verifiable credentials of the trustee as well as any direct experience the trustor has had of the trustee. One simple definition of reputation is [10]:
*"perception that an agent creates through past actions about its intentions and norms"*

For reputation to be meaningful (distinct from the trustors own experience), it must be the collective perception of a community of observers that is somehow made available to the trustor.

## 3 Trust Overlay Model

For the purposes of this work, we adopt a two layer model for communications between peers(see Fig. 1). Peers can either interact for a transaction or to exchange trust information. For modelling purposes, each service usage interaction is a discrete event. A logically separate trust management layer handles threat notifications and other pertinent information. We mimic social trust by setting a fuzzy trust level. Each different service can then make an appropriate decision based on this trust level e.g. certain actions may be allowed and others not. In our system, we model trust as a vector. In the simplest case, at least if there is just one service, this can be viewed as a simple number in the range (0,1). Each peer may then maintain a local trust score relating to each other peer of which it is aware. If peer As trust in peer B is 1, then peer A completely trusts peer B. A score of 0 indicates no trust. Note that this is consistent with Gambettas widely accepted definition of trust as cited above. If trust is to be a probability measure, then the (0,1) range is natural.

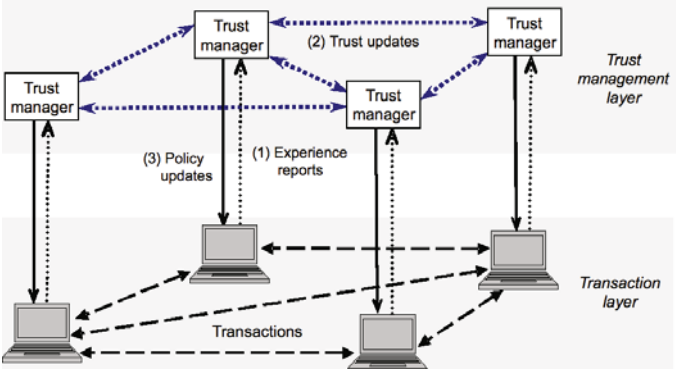


**Fig. 1.** Trust overlay helps to secure transactions in a digital ecosystem.

### 3.1 Algorithms

The way trust is updated based on both experience and recommendations has a profound impact on the usefulness of this kind of overlay system. Note that peers are autonomous in our model and each might implement a different algorithm for updating and using trust. It can also be expected that a peers behaviour in terms of handling trust may change if it is hijacked. Potential trust update algorithms include:

*Moving average:* Advanced moving averages are possible, where old data is remembered using data reduction and layered windowing techniques.

*Exponential average:* Exponential averaging is a natural way to update trust as recent experience is given greater weight than old values, and no memory is required in the system, making it more attractive than using a moving average.

*No forgiveness:* This in a draconian policy where a peer behaving badly has its trust set to zero forever. Even more extreme is where a peer that is reported by a third party as behaving badly has its trust set to zero forever. This could perhaps be used if a particularly sensitive service is misused.

*Second chance (generally, nth chance):* Draconian policies are generally not a good idea. IDS and other security systems are prone to false alarms. A variation on the no forgiveness approach is to allow some bounded number of misdemeanours.

*Hard to gain trust; easy to lose it:* To discourage collusion, there is a case for making trust hard to gain and easy to lose.

*Use of corroboration:* To prevent an attack by up to $k$ colluding bad peers, we could require positive recommendations from at least $k + 1$ different peers.

*Use of trust threshold for accepting recommendations:* It is possible to model the ability to issue a recommendation as a kind of service usage on that receiving peer. Thus the receiving peer can apply a trust threshold to decide whether to accept that recommendation in the same way as any transaction attempt is adjudicated.

Next, we examine a simple exponential average algorithm in the context of both direct experience and referral.

**Exponential Average Direct Experience Trust Algorithm.** In the case of direct experience the exponential average algorithm, published in [8], will be written as:

$$T_{i,j(n)} = \alpha E + (1 - \alpha) T_{i,j(n-1)}$$

Where:

$T_{i,j(n)}$ is the $n$th value of trust placed by $i$ in $j$
$E$ is the latest direct experience report ($0 \leq E \leq 1$)
$\alpha$ is the rate of adoption of trust ($0 \leq \alpha \leq 1$)

The value $\alpha$ determines the rate of adoption. $\alpha = 0$ means that the trust value is not affected by experience. If $\alpha = 1$, the trust value is always defined as the latest experience and no memory is maintained. The algorithm behaviour is illustrated ihn Figs. 2 through 4. Each of these illustrations considers the case where experience is a measure of the dependability of a service and that dependability is measured is measured as a binary value (1 or 0).

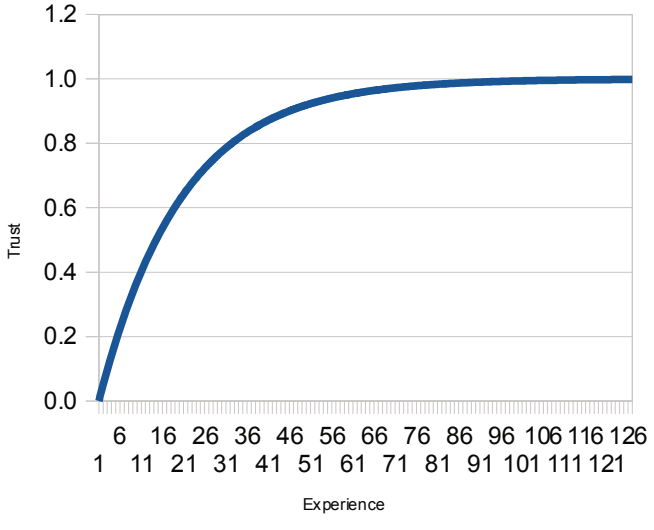


**Fig. 2.** Exponential Average Experiential Trust (100% Positive Experience)

When the experience $E$ is always 1 (i.e. the service is always dependable) this trust evolves as in Fig. 2. Fig. 3 shows how this trust evolves where every

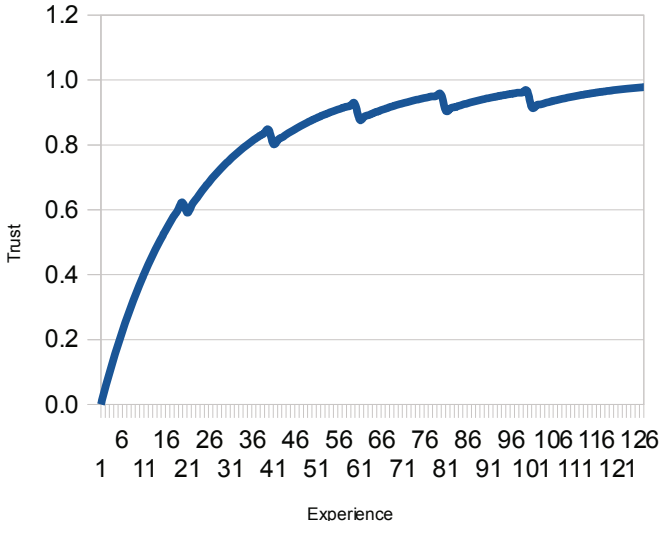


**Fig. 3.** Exponential Average Experiential Trust (95% Positive Experience)

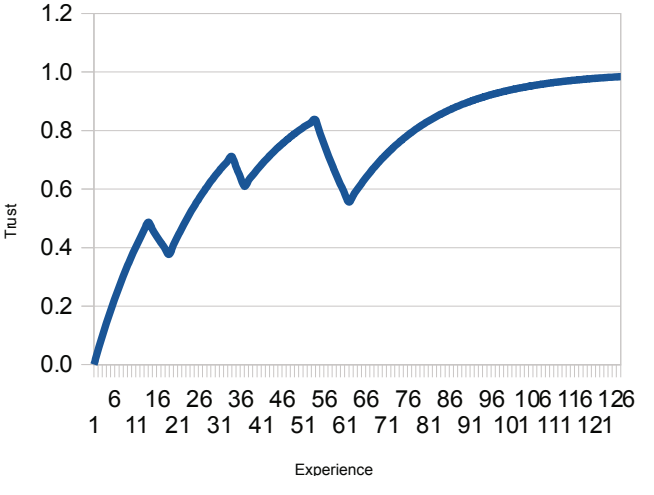


**Fig. 4.** Exponential Average Experiential Trust (with failure clusters)

20th experience is reported as a failure. There remains a trend of exponential recovery but the failures impact on the trust value while allowing for recovery in subsequent success reports. Similarly, Fig. 4 demonstrates how the algorithm performs when clusters of failures occur. The rate of adoption and reduction in trust in this algorithm is thevalue. In some cases it might be desirable that this value varies depending on the latest experience. For example in some cases it can be argued that while it is difficult to gain trust (slow adoption of trust) it is

easy to lose it (fast reduction) based on bad experience. To model this we can vary the value as follows:

$$\begin{aligned} &\text{if } (T_{i,j(n-1)} < E) \\ &\text{then} \\ &\quad \alpha = 0.1 \\ &\text{else} \\ &\quad \alpha = 0.4 \end{aligned}$$

**Exponential Average Referral Based Trust Algorithm.** In the case of referral based trust the exponential average algorithm, published in [8], can be written as:

$$T_{i,j(n)} = \beta T_{i,k} T_{k,j} + (1 - \beta T_{i,k}) T_{i,j(n-1)}$$

Where:

$T_{i,j(n)}$ is the $n$th value of trust placed by $i$ in $j$
$T_{i,k}$ is the trust $i$ has in $k$'s referral
$T_{k,j}$ is the trust $k$ has in $j$
$\beta$ is the influence that referrals have on local trust $(0 \leq \beta \leq 1)$

The value of $T_{i,j}$ represents an indirect functional trust that $i$ has in $j$. In this case $i$ has received a recommendation of direct functional trust,$T_{k,j}$. $T_{i,k}$ is the trust that $i$ has in $k$'s referral. The larger the value of $T_{i,k}$, (i.e. the more $i$ trusts $k$'s referral), the greater the influence of $k$s trust in $j$ on the newly updated value of $T_{i,j}$. Note that, if $T_{i,k} = 0$, this causes $T_{i,j}$ to be unchanged.

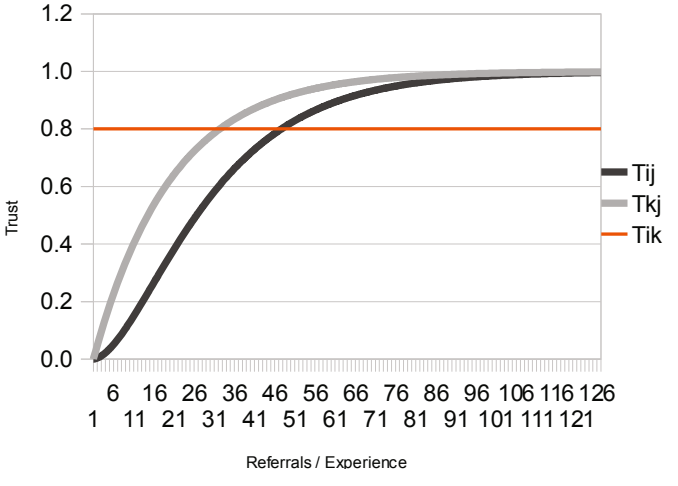


**Fig. 5.** Exponential Average Referral Trust (100% Positive Experience, $\beta = 0.1$)

Fig. 5 shows how this algorithm behaves for a fixed value for $T_{i,k}$ and $\beta = 0.1$. $T_{k,j}$ is a direct functional trust derived as described above and shown in Fig. 2. $T_{i,j}$ indicates how the indirect functional trust evolves. Fig. 6 shows how this is affected when the direct functional trust experiences one failure in every 20 experience reports. Fig. 7 shows what happens when the direct functional experience reports yield one failure in 20 experiences followed by a general failure after which only negative reports are received. Each of these examples consider the referral trust to be constant. The following figure (Fig. 8) show how the indirect functional trust evolves when the referral trust varies, in this case a linear increase.

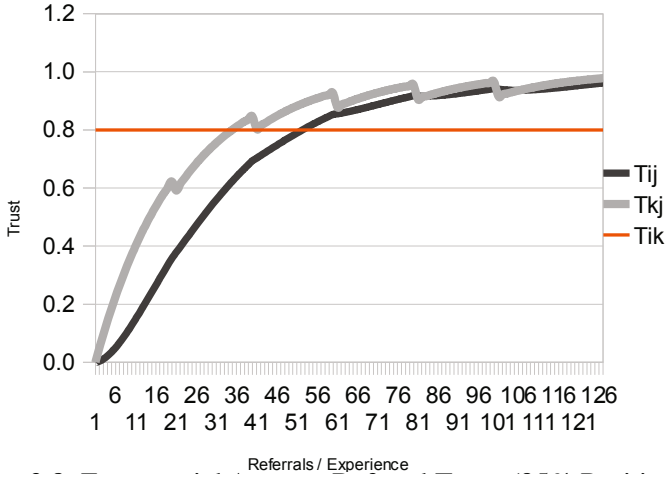


**Fig. 6.** Exponential Average Referral Trust (95% Positive Experience, $\beta = 0.1$)

## 4 Implementation

A project called *TrustFlow*[2] has been established on sourceforge to house the development of the OPAALS trust model. The project contains the source code as well as standalone examples showing how the framework is used. Sample trust algorithms are provided together with documentation on how to create and use further algorithms.

The OPAALS Trust Model for digital ecosystems is described in Section 3. The approach is to use a trust overlay network for providing a community based approach to trustworthiness based on the reputation of entities. A UML Class Diagram of this model is provided in Fig. 9. The Entity can represent a node, service, resource, a service provider or a service consumer. Each entity has a

[2] TrustFlow, `http://trust ow.sourceforge.net`

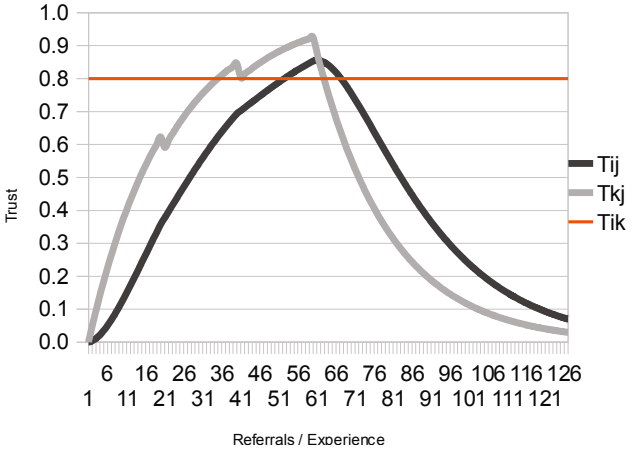


**Fig. 7.** Exponential Average Referral Trust (95% Positive Experience, general failure after 60 experiences, $\beta = 0.1$)

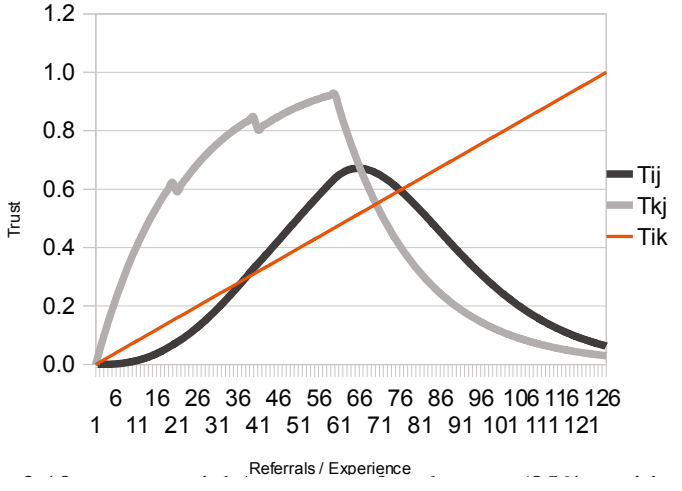


**Fig. 8.** Exponential Average Referral Trust (95% Positive Experience, general failure after 60 experiences, $\beta = 0.1$, linear increase in referral trust)

Trust Manager associated with it. The entities gain experience from interacting with other entities and publish reports of these experiences to the Trust Manager. Using a pre-defined context dependent algorithm the Trust Manager updates the entities' local trust and based on a policy of sharing trust information provides trust updates to other Trust Managers in the overlay network.

Entities are responsible for creating trust algorithms. Each Entity has an associated TrustManager. A TrustAlgorithm is associated on a per entity basis with an trustor and a context. Entities create these algorithms and publish them to the TrustManager. The TrustManager maintains a set of these algorithms and

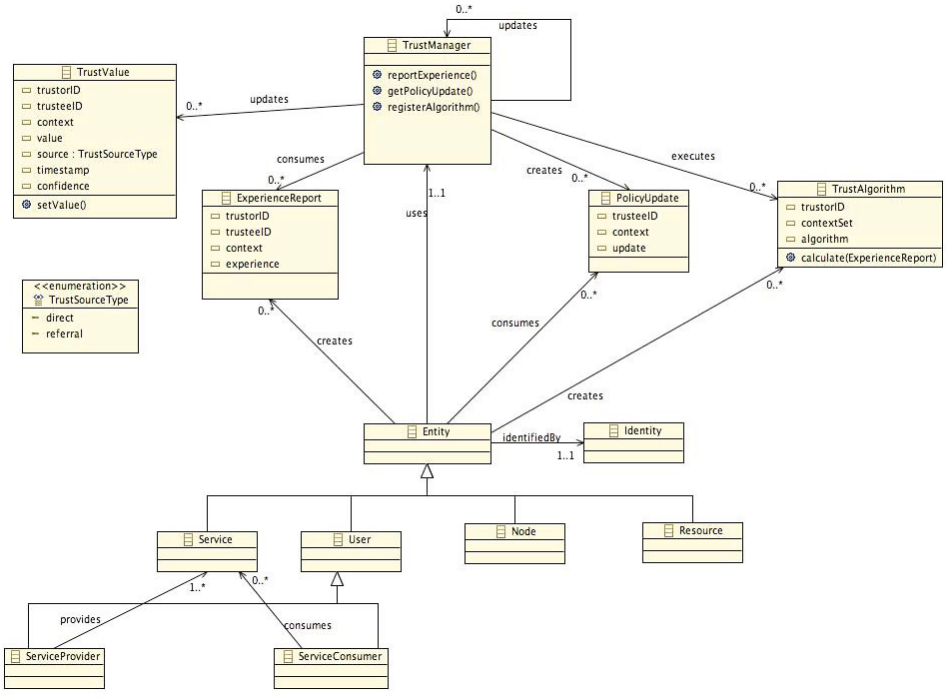


**Fig. 9.** UML Class Diagram of TrustManager and associated classes

when it receives an ExperienceReport, it uses the appropriate TrustAlgorithm by performing a lookup on trustor-context tuple.

The TrustManager also maintains a set of TrustValue objects which it updates after performing the algorithm on the ExperienceReport. The Entity can request PolicyUpdates from the TrustManager. These policy updates are derived from current TrustValues and are used by the entity in making choices about future interactions with other entities. The TrustValue class includes a 'source' element indicating whether the value was derived from direct experience or based on referrals. The class also has a timestamp attribute, as recently gathered trust might be of more value than older values. Also there is a confidence attribute, used to denote the confidence in this Trust Value (e.g. this value can be increased each time the Trust Value is updated, i.e. 100 updates to the Trust Value implies more confidence in the Trust Value than 10). The rest of this section describes the primary elements of this model and their role and implementation.

### 4.1 Trust Manager

The TrustManager acts as the core of the trust model. It has the following set of responsibilities:

1. Maintaining a set of algorithms for connected Trustor entities for specific contexts. Each context can have distinct algorithms for determining trust based on direct experience, referrals and reputation.
2. Receiving ExperienceReports from the Trustor entities and generating updated trust values according to the appropriate algorithms.
3. Providing updated TrustValues to other TrustManagers which can be used as inputs to referral or reputation algorithms. It is only appropriate that these published TrustValues have been derived from direct experience.
4. Providing PolicyUpdates to the Trustor entities based on current TrustValues derived from direct experience and referrals and reputation.

### 4.2 TrustValue

A TrustValue represents a single instance of a trust value a relying party has in a trusted party in a particular context. In effect each TrustValue instance is the output of a trust algorithm. The trust value itself is represented as float. Each TrustValue has a type associated with it. The type indicates the source of the data used to derive the value. Firstly, this can be direct experience, where the trust value was calculated using a direct experience report. Secondly, this can be referral, where the relying party has received a trust value from a third party and has used an algorithm to calculate a trust value based on referrals. Thirdly, the trust value can be derived from a reputation based algorithm, where a set of trust values is used as an input. Finally, the trust value can be calculated using a hybrid of direct experience, referral and reputation algorithms.

### 4.3 ExperienceReport

An ExperienceReport represents a unit of experience from an end user pertaining to an experience with a trustee in a context. The ExperienceReport forms the basis of all trust calculations as it is required to calculate any direct experience TrustValues. Referrals are only valid if their source is one of direct experience.

The creation of ExperienceReports is vital for the successful deployment of our trust network and two implementations are currently available. The first of these is called SimpleExperienceReport, which considers experience as a simple negative or positive value, i.e. the experience is either good or bad. The second implementation is called AccountedReport (see Figure 3.6) which makes use of data which will be created by our Accountability framework.

### 4.4 TrustAlgorithm

Trust algorithms have the role of taking some known information and deriving a new trust value according to some predefined logic. A discussion of suitable trust algorithms is provided earlier in this paper in section 3. An interface for TrustAlgorithm is defined with two methods. The first method, setTrustValues(String, float), allows the user to input hard values for some trusted parties.

For example, a service consumer might decide that it would like to overload the algorithm with some hardcoded values for some of the service providers it interacts with. The second method is setParameters(String, String) which is used to load parameters of the logic in the algorithm. An example of these parameters would be the rate of increase or decrease in trustworthiness for a specific context. Three further interfaces extend this base interface

### 4.5 Configuring the TrustManager

Each TrustManager manages a set of trust algorithms for various contexts. As each relying party has a TrustManager to manage its preferred algorithms, it is useful to devise a means of configuring the TrustManager to achieve this on a run time basis. To aid this a TrustAlgorithmConfiguration XML schema is designed. Trust Contexts are configured separately and each Context can define TrustAlgorthms for direct experience, referrals and reputation.

## 5 Conclusion

The work described here provides an implementation of a trust overlay network based on a model developed in the OPAALS EU FP6 Network of Excellence project. The implementation can be easily reused by third parties to introduce trust to existing software. Sample algorithms are provided in the implementation and further algorithms can be easily developed by developed and used by the framework through a plugin model.

## Acknowledgment

This work is funded by the EU FP6 Network of Excellence OPAALS, `http://www.opaals.org`.

## References

1. Aitken, D., Bligh, J., Callanan, O., Corcoran, D., Tobin, J.: Peer-to-peer technologies and protocols (2001), `http://ntrg.cs.tcd.ie/undergrad/4ba2.02/p2p/`
2. Dingledine, R., Freedman, M., Molnar, D.: Accountability measures for peer-to-peer systems. In: Peer-to-Peer: Harnessing the Power of Disruptive Technologies. O'Reilly, Sebastopol (2000)
3. Douceur, J.R.: The sybil attack. In: Druschel, P., Kaashoek, M.F., Rowstron, A. (eds.) IPTPS 2002. LNCS, vol. 2429, pp. 251–260. Springer, Heidelberg (2002)
4. Gambetta, D.: Can we trust trust (July 09, 1988), `http://citeseer.ist.psu.edu/489470.html`, `http://www.sociology.ox.ac.uk/papers/gambetta213-237.pdf`
5. Grandison, T.: Conceptions of trust: Definition, constructs and models. In: Song, R. (ed.) Trust in E-Services: Technologies, Practices and Challenges. IGI Global (2007)

6. Jøsang, A., Ismail, R., Boyd, C.A.: A survey of trust and reputation systems for online service provision. Decision Support Systems pp. 618–644 (March 01, 2007), http://eprints.qut.edu.au/archive/00007280/, http://eprints.qut.edu.au/secure/00007280/01/JIB2007-DSS.pdf
7. Li, H., Singhal, M.: Trust management in distributed systems. IEEE Computer 40(2), 45–53 (2007), http://doi.ieeecomputersociety.org/10.1109/MC.2007.76
8. McGibney, J., Botvich, D.: Distributed dynamic protection of services on ad hoc and p2p networks. In: Medhi, D., Nogueira, J.M.S., Pfeifer, T., Wu, S.F. (eds.) IPOM 2007. LNCS, vol. 4786, pp. 95–106. Springer, Heidelberg (2007)
9. McKnight, D.H., Chervany, N.L.: The meanings of trust. MISRC Working Paper Series 96-04, Carlson School of Management, University of Minnesota (2006), http://misrc.umn.edu/wpaper/WorkingPapers/9604.pdf
10. Mui, L., Mohtashemi, M., Halberstadt, A.: A computational model of trust and reputation for E-businesses. In: HICSS. p. 188 (2002), http://dlib2.computer.org/conferen/hicss/1435/pdf/14350188.pdf?
11. Ruohomaa, S., Kutvonen, L.: Trust management survey. In: Herrmann, P., Issarny, V., Shiu, S.C.K. (eds.) iTrust 2005. LNCS, vol. 3477, pp. 77–92. Springer, Heidelberg (2005), http://dx.doi.org/10.1007/11429760_6
12. Solhaug, B., Elgesem, D., Stlen, K.: Why trust is not proportional to risk. In: ARES '07: Proceedings of the Second International Conference on Availability, Reliability and Security, pp. 11–18. IEEE Computer Society, Washington (2007)