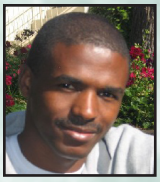
ISR Institute for Software Research
University of California, Irvine

A Generic Framework for Modeling Decentralized Reputation-based Trust Models

Girish Suryanarayana
University of California, Irvine
sgirish@ics.uci.edu



Mamadou Diallo
University of California, Irvine
mdiallo@ics.uci.edu

Richard N. Taylor
University of California, Irvine
taylor@ics.uci.edu

August 2007

ISR Technical Report # UCI-ISR-07-4

Institute for Software Research
ICS2 217
University of California, Irvine
Irvine, CA 92697-3455
www.isr.uci.edu

A Generic Framework for Modeling Decentralized Reputation-based Trust Models

Girish Suryanarayana, Mamadou Diallo, Richard N. Taylor
Institute for Software Research
University of California, Irvine
{sgirish,mdiallo,taylor}@ics.uci.edu

ISR Technical Report # UCI-ISR-07-04

August 2007

Abstract: Decentralized applications do not have a single centralized authority that can safeguard peers in the system from malicious attacks. Each peer is autonomous and must adopt measures to protect itself. Reputation-based trust management systems enable peers to develop trust relationships with each other based on their reputations. These trust relationships help a peer determine the trustworthiness of other peers in the system and thus help safeguard itself from malicious peers. A number of decentralized reputation-based trust models have been discussed in the literature. However, a common understanding of what a trust model is and what its constituents are has been lacking. Further, there has been little work directed towards the creation of a generic framework that will comprehensively help to express existing reputation models as well as create new models. In this paper, we present the 4C framework for modeling decentralized reputation-based trust models. The 4C framework builds upon the common functional aspects of reputation models and consists of four generic sub-models that help to express reputation models. The 4C framework is described using an XML-based schema that makes the 4C framework extensible for enabling the expression of new types of reputation models in the future. We have evaluated the 4C framework by using it to describe three decentralized reputation models and have built a 4C editor to facilitate the generation of XML-based descriptions of reputation models. We have also demonstrated how these trust model descriptions can be leveraged to aid the construction of decentralized trust-enabled applications.

A Generic Framework for Modeling Decentralized Reputation-based Trust Models

Girish Suryanarayana, Mamadou Diallo, Richard N. Taylor

Institute for Software Research
University of California, Irvine
Irvine, CA 92697-3425
{sgirish,mdiallo,taylor}@ics.uci.edu

ISR Technical Report # UCI-ISR-07-04

August 2007

ABSTRACT

Decentralized applications do not have a single centralized authority that can safeguard peers in the system from malicious attacks. Each peer is autonomous and must adopt measures to protect itself. Reputation-based trust management systems enable peers to develop trust relationships with each other based on their reputations. These trust relationships help a peer determine the trustworthiness of other peers in the system and thus help safeguard itself from malicious peers. A number of decentralized reputation-based trust models have been discussed in the literature. However, a common understanding of what a trust model is and what its constituents are has been lacking. Further, there has been little work directed towards the creation of a generic framework that will comprehensively help to express existing reputation models as well as create new models. In this paper, we present the 4C framework for modeling decentralized reputation-based trust models. The 4C framework builds upon the common functional aspects of reputation models and consists of four generic sub-models that help to express reputation models. The 4C framework is described using an XML-based schema that makes the 4C framework extensible for enabling the expression of new types of reputation models in the future. We have evaluated the 4C framework by using it to describe three decentralized reputation models and have built a 4C editor to facilitate the generation of XML-based descriptions of reputation models. We have also demonstrated how these trust model descriptions can be leveraged to aid the construction of decentralized trust-enabled applications.

Keywords

Decentralized Trust Management, Reputation Management

1. INTRODUCTION

Decentralized applications are those that lack a centralized single authority. Instead, each entity, also called a peer, is autonomous and makes local decisions towards its individual goals. Peers directly interact with each other and exchange information and resources. In an open decentralized system, peers can enter or leave the system at any time, thus exposing the system to the presence of malicious peers. While a centralized authority could help protect a centralized system against such malicious peers, in a decentralized system, each peer has to adopt its own measures to protect itself.

Trust management has been found to serve as a potential countermeasure against the threats of malicious peers in open decentralized applications. Trust relationships between peers help them determine each other's trustworthiness, following which peers can make well-informed decisions about interacting with other peers. Trust management systems that utilize the reputation of peers to determine trustworthiness are called reputation-based trust management systems. The benefits of such decentralized reputation-based systems have been recognized and, consequently, much attention has been focused on building new types of reputation models and systems.

However, current literature lacks a common understanding of what a trust model really is. A trust model means different things to different people. For some, the various information elements that constitute trust is the trust model [6]. For others, a trust model includes the algorithm that computes a trust value [18] or the protocol that is employed to gather trust information [7]. Given the role of decentralized trust and reputation management in decentralized applications, we believe it is imperative to have a clear understanding of what a trust model is and what are the constituent elements that characterize a trust model.

Additionally, there has been little work in the research literature directed at creating a generic framework that can comprehensively express existing decentralized reputation-based trust models as well as help create new models. Models discussed in the literature adopt different approaches and express these reputation models in different ways. There is no standard platform that can be used to describe existing models as well as help explore newer models.

Towards addressing these shortcomings, in this paper we present our definition of a trust model and introduce the 4C framework. 4C is a generic and extensible framework for modeling decentralized reputation-based trust models. 4C is based on the common functional aspects of trust models and consists of four sub-models: Content, Communication, Computation and Counteraction. These

sub-models are generic and help describe existing and new models. The 4C framework is expressed using an XML-based trust model schema. The use of XML makes the framework extensible so that it can provide richer descriptions of trust models in the future.

In order to evaluate the benefits of the 4C framework, we have used the 4C framework to express three sample decentralized reputation-based trust models. We have also implemented a GUI-based editor based on the 4C framework that first guides a user through the selection of elements for a trust model and finally generates an XML-based trust model description based on the user's selection. To demonstrate the benefit of generating trust model descriptions, we have created an XML-based description of a decentralized reputation model using the 4C editor tool and then used this description with the PACE architectural style [25] to automatically generate software components and utilities for building a trust-enabled crisis response decentralized application.

The rest of the paper is structured as follows. Section 2 introduces decentralized trust management and the concepts of trust and reputation. Section 3 discusses relevant related work. Section 4 describes the essential aspects of a trust model while section 5 presents the 4C framework. The evaluation of the 4C framework is presented in section 6. The paper ends with a discussion of future work and conclusions in sections 7 and 8 respectively.

2. DECENTRALIZED TRUST MANAGEMENT

In this section, we present a a discussion of trust and reputation concepts. As discussed in the previous section, trust relationships have been found to be immensely useful in determining the trustworthiness of peers and protecting against potential malicious attacks in decentralized applications. Therefore, decentralized trust management has received increasing attention from researchers.

2.1 Trust

Trust has been a familiar concept since the inception of society. We are dependent upon trust in almost all facets of our lives. The concept of trust is not only basic to society but also undoubtedly significant. Therefore, interest in it is not limited to electronic communities but reaches across several research disciplines such as psychology, sociology, computer science etc. Below we briefly take a look at some definitions and concepts related to trust.

Deutsch [10] coined one of the most popular definitions of trust which states that: *(a) an individual is confronted with an ambiguous path, a path that can lead to an event perceived to be beneficial or to an event perceived to be harmful; (b) he perceives that the occurrence of these events is contingent on the behavior of another person; and (c) he perceives the strength of a harmful event to be greater than the strength of a beneficial event. If he chooses to take an ambiguous path with such properties, he makes a trusting choice; else he makes a distrustful choice*.

An interesting fact about the above definition pointed out by Marsh [21] is that trust is considered to be subjective and dependent on the views of the individual. Deutsch further refined his definition of trust as *confidence that an individual will find what is desired from another, rather than what is feared* [11]. This definition is also echoed by the Webster dictionary which defines trust as *a confident dependence on the character, ability, strength, or truth of someone or something*.

Another popular definition of trust that has also been adopted by computer scientists is the one coined by Diego Gambetta [15]. He defines trust as *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 (or independently of his capacity ever to be able to monitor it) and in a context in which it affects his own action*. Gambetta introduced the concept of using values for trust and also defended the existence of competition among cooperating agents. A recent definition of trust has been put forth by Grandison and Sloman [16] who define trust as *the firm belief in the competence of an entity to act dependably, securely, and reliably within a specified context*.

Trust is conditionally transitive. This means that if Alice trusts Bob and Bob trusts Carol, Alice may trust Carol only if certain possibly application-specific conditions are met. Trust can also be multi-dimensional and depend upon the context of the trust. For example, Alice may trust Bob completely when it comes to repairing electronic devices but may not trust Bob when it comes to repairing cars. Further, trust can be expressed in various ways such as binary or continuous values or a set of discrete values.

2.2 Reputation

The concept of reputation is closely related to trust and can be used to determine the trustworthiness of an entity [32]. Abdul-Rehman [2] defines reputation as *an expectation about an individual's behavior based on information about or observations of its past behavior*. In online communities, where an individual may have very less information to determine the trustworthiness of others, their reputation information is typically used to determine the extent to which they can be trusted. An individual who is more reputed is generally considered to be more trustworthy. Reputation can be determined in several ways. For example, a person may either rely on his direct experiences, or rely on the experiences of other people, or existing social relationships or a combination of the above to determine the reputation of another person.

3. RELATED WORK

As discussed above, trust management has received increasing attention with special interest in decentralized applications. Consequently, a number of decentralized trust management solutions have been researched and developed [16, 26, 32]. The term Decentralized Trust Management was first coined by Blaze who introduced credential-based access systems such as Policymaker [4]. These systems restrict access to resources by verifying presented credentials against application-defined policies. Trust management systems also include reputation-based systems such as XREP [8] and hTrust [6] that use reputation to determine the trustworthiness of peers in the system. Since the focus of our work is decentralized reputation management systems, our discussion below is mainly focused on decentralized reputation models and systems.

In reputation-based systems, peers exchange reputation information about other peers in the system in order to determine the trustworthiness of a peer. Different models use different kinds of reputation mechanisms. For example, the Distributed Trust model [1] uses recommendations and recommender trust to arrive at a reputation value for a peer. In this model, the provider peer queries other peers in the system for recommendations and aggregates the data received to determine whether the requestor can be trusted. The NICE [20] trust model which uses cookies to encapsulate reputation data, on the other hand, shifts the onus of trust data collection on the peer that needs a particular service. Thus a peer that wants to access a particular resource or service needs to present the provider with a set of cookies that help prove its trustworthiness.

There are other reputation-based trust models that additionally use social relationships to help evaluate a peer's trustworthiness. REGRET [24] is one such model that constructs a sociogram based upon existing relationships within a community, and uses it to allot reputation values to peers. Other models that use social relationships include community-based reputation [31] and NodeRanking [23]. Several trust and reputation algorithms have also been developed. These include the Beta Reputation algorithm [18] that uses beta probability density functions to combine feedback and derive reputation ratings in a decentralized application and the Eigentrust algorithm [19] that computes global trust values for each peer based on the peer's history of uploads. There also exist several trust frameworks and platforms, most of which are centralized that support trust management. Examples include the trust-based admission control architecture [17], XenoTrust [13], and the SECURE computing platform [5]. A generic trust model that can be used for the design of trust-related services in electronic commerce applications has been presented in [30]. This trust model is based on the concept of party and control trust. Party trust refers to trust in another party while control trust refers to the trust that the parties have in a third entity that controls the risk of the interaction.

4. FOUNDATIONS OF 4C FRAMEWORK

In this section, we present our definition of a trust model, discuss the different roles of peers in a decentralized system and identify and describe the three typical functional aspects of trust models. These functional aspects form the basis of the 4C framework that is described in the next section.

4.1 Trust Model

A trust model essentially helps model the trust relationships between peers in the system. Different types of trust and reputation models have been developed towards different objectives or targeted at specific applications. However, the problem is there is no uniform definition across the research community of what a trust model is. For some, the various elements that constitute the trust data define the trust model [6]. For some others, it may mean just a trust algorithm and a way of combining different trust information to compute a single trust value [18, 19], while for others, a trust model may also encompass a trust-specific protocol to gather trust information [1, 7]. Yet others may want a trust model to also specify how and to whom any trust data should be communicated [3].

We present the following definition of a trust model towards unifying these above aspects. *A trust model describes what trust information is used to establish trust relationships, how that trust information is exchanged, how that trust information is combined to determine trustworthiness, and how that trust information is modified in response to personal and reported experiences.*

The above definition of a trust model identifies three important functional aspects that constitute a trust model: trust information gathering, trust information analysis, and incorporating feedback obtained from interactions among peers. These three functional aspects together form a superset of the aspects found in existing trust models. These functional aspects form the basis of the four sub-models that constitute the 4C framework.

4.2 Roles of peers

We refer to each entity in a decentralized system as a "peer". Peers are autonomous and make local decisions about their behavior in the system. Peers are distinguished through the use of unique digital identities. A user with multiple digital identities would not be treated as a single peer, but multiple peers. Similarly, a group of entities sharing a digital identity would be treated as a single peer. Thus, the term "peer" maps to a unique digital identity rather than a unique user or machine. In the context of trust and reputation, there are different roles that a peer can take. We identify three roles that a peer can simultaneously take - *subject*, *target*, and *recommender*. The *subject* refers to the peer who wants to evaluate the reputation of the *target* peer. The *recommender* refers to a peer who provides the subject peer information about the reputation of the target peer. It should be noted that a recommender may also provide trust information related to other recommenders.

4.3 Information Gathering

Each peer needs to determine the extent to which it can trust another peer before actually interacting with it. Since it is generally assumed that past behavior provides a good basis for prediction of future behavior, each peer typically stores its impressions from past interactions with other peers. It also gathers information about the past behavior of concerned peers by asking other peers in the system. This information gathering aspect has two parts: the actual trust information that needs to be collected, and the protocol that is used by peers to exchange trust information.

The choice of a trust model that is adopted for an application is also affected by whether the application permits peers to be grouped together according to some criteria. For example, it is possible for an application to group peers according to their interests, or expertise, or their membership of certain organizations. A trust model adopted for an application that supports the formation of groups must be equipped to include group relationships in trust determination. Therefore, there is a need to distinguish between applications that support groups and applications that don't.

4.3.1 Applications without Groups

Trust Information. The trust information in such an application

typically contains the identity of the recommender who is providing the trust information, the identity of the target peer whose trust value is being provided, and the trust value. This trust value may be expressed as discrete integers or continuous real numbers. The trust model may also optionally include the context of the trust being provided and time of expiry of the trust value which dictates how long the trust value is valid.

Protocol. The protocol used to gather trust data is typically of a query-response nature. A peer queries other peers in the system for trust data about a target peer. This originator can also specify the context in which it is seeking trust data. Depending upon the nature of the application and the trust model, the originator may either send this query to selected peers or may choose to broadcast the query and specify a hop count to limit the control the flooding of queries. The protocol may specify if and under what condition peers who receive this query forward it to other peers. Upon receiving a query addressed to it, a peer may respond accordingly with the required trust data. Or if a broadcast mechanism is being used, any peer with relevant trust data may choose to respond. In the next phase, the trust model may require the originating peer to confirm received responses with the responding peers, or may require the originating peer to query trust data about the responders. If so, the originating peer will issue queries similar to the original queries and receive responses accordingly.

4.3.2 Applications with Groups

Trust Information. In such an application, there are two types of trust data that are used - one that describes trust data for individual peers, and the other that describes trust data for groups. The trust data for individual peers used in a structured application can contain all the elements of trust data used in the case of a unstructured application. But additionally the trust data may include the name of the groups or communities that the peers belong to. This is to enable the additional use of trust data related to groups and group relationships for determining trustworthiness. Group-related trust data contains similar elements to the trust data for individual peers. In other words, trust data about a group contains the identity of the peer reporting the trust data, identity of the group about whom the trust data is being reported, the trust value, an optional trust context and an optional time of expiry of the trust value.

Protocol. In an application that permits groups, there are two parts to the protocol that is used to exchange trust information. The first part is used to query and gather trust data for individual peers. And the second part is used to query and gather group-related trust data. It is possible that the same protocol mechanism may be used to query and gather both types of trust data. The protocol used to gather peer trust data is very similar to the one used for unstructured applications. The only difference is that in this case, additional information about any groups that a peer belongs to is returned. Similarly, the protocol used to gather group trust data is identical to the protocol used to gather peer trust data.

The trust information and the protocols used for both structured and unstructured applications are modeled by the Content and the Communication sub-models of the 4C framework respectively. These sub-models are described in section 5.

4.4 Trust Analysis

For the determination of trust, a trust model may consider peer and group trust information obtained from two main sources: a peer's own past experience (history which we represent here by P), and trust information received from other peers whom we call recommenders (recommendations which we represent here by R). The information reported by these recommenders is given importance according to their trustworthiness as recommenders. Thus a peer also needs to determine and maintain information about the trustworthiness of these recommenders. There are several factors that play an important role in the computation of trust. These factors include the context of the trust information, the period for which the trust information is valid, and the existence of any group relationships. These factors have varying influence on the computation of trust depending upon the nature of the trust model.

4.5 Incorporating Feedback

After a peer computes a trust value for a target peer, it may decide to proceed with interaction with the target peer. The result of the interaction will affect the belief of the peer in both the target peer and the peers who recommended the target peer. For example, if the interaction is successful, the peer will decide to trust the target peer and the recommenders more and if the interaction turns out to be unsuccessful, the peer will reduce its trust in the target peer and the recommenders. The peer can also choose to actively propagate any trust information it considers very important to other peers in the system. This is especially useful for warning other peers about malicious peers. For this the trust model may require a peer to either broadcast or send to selected peers a message with the necessary trust information.

5. THE 4C FRAMEWORK

We propose 4C, a generic extensible framework for creating and expressing decentralized reputation-based trust models. The 4C framework is based upon the functional aspects of reputation models described in the previous section and consists of four sub-models: Content, Communication, Computation and Counteraction. These sub-models are generic and can be extended in the future to enable richer expressions of trust models. These sub-models together provide the ability to express existing reputation-based trust models as well as to create new models. Below, we describe each of the four sub-models of the 4C framework in detail.

5.1 Content Sub-Model

The Content model describes the various information elements of a decentralized reputation model and corresponds to the trust information used in the Information Gathering functional aspect.

5.1.1 Social Groups

As mentioned in the previous section, peers in decentralized communities may form certain social groups based on common concerns, interests or even expertise. It is also possible that a peer may not just belong to one group but instead be a member of more than one group at any time. Further, a group may have a hierarchy

of nested sub-groups. Such structures serve as potential sources of reputation information. Group relationships provide additional information that help model better the trust relationships between peers. Sometimes, a peer may also maintain an informal structure in the form of a list of peers that it trusts absolutely. Having such a list is useful so that a peer can trust their recommendations about recommenders and other peers. If the application is structured, the reputation information of a peer may include information about the groups it belongs to. Further, in addition to the reputation of every peer, the Content sub-model will also model the reputation of groups. Thus the following elements of the Content sub-models refer to both peer and group reputation.

5.1.2 Expression

The reputation value can be either expressed as a set of binary, discrete or continuous values. A binary representation only enables the expression of trust or distrust. A discrete set of values help better express trust levels, but they do not provide the richness of expression and comparison that continuous values do.

5.1.3 Context

The context is an extremely important factor in determining trust. As mentioned earlier, trust depends upon the context. For example, a peer may consider another peer completely trustworthy when it comes to information about a particular topic or a certain peer, but may not consider it trustworthy in other cases.

5.1.4 Period of validity

Reputation may also has a time-to-live attribute. A recommender can specify the time until which its data can be considered valid. Thus, a piece of reputation information encapsulates information about the subject, target, and recommender peers, and specifies the context of the reputation and the time until the reputation expires.

5.2 Communication Sub-Model

The Communication sub-model defines how peers interact with each other and my specify the protocol used by peers to exchange trust data.

Type of protocol used to collect trust information. The choice of the type of protocol depends upon the type and needs of the application, and can be classified into two main types: subject-initiated and target-initiated. In particular, when a peer needs trust information about the data or resources received from another peer, the onus of deciding the provider's (target peer's) trustworthiness lies with the receiver (subject peer). In this case, the target peer may not necessarily care whether the subject peer believes the target peer is trustworthy. This type of communication mechanism is termed subject-initiated since the subject peer initiates the collection of trust information. Interactions between peers in such a scenario is as follows:

- The subject peer queries other peers for information about the target peer.
- Recommenders directly respond to queries received from the subject peer if they have the requested information.
- If recommenders do not have the requested information and a hop count is specified, they forward the query to other known recommenders or peers. This process continues until the specified hop count limit is reached.
- These recommenders respond back to the subject peer with requested information about the target peer.

In the case where the application contains groups and group relationships are considered in the determination of trust, a subject peer may additionally query other peers for group information. Peers may also disseminate information about their group affiliation to other peers as their group affiliation changes over time.

However, in some applications, a target peer may need to prove its trustworthiness to the subject peer because the target peer requires certain resources from the subject peer or wants the subject peer to believe in the information being reported by the target peer. In such cases, the onus of aggregating and presenting trust information about the target peer to the subject peer lies with the target peer itself. This type of communication mechanism is termed target-initiated since the target peer initiates the collection of trust information about itself. Interactions between peers in such a scenario is as follows:

- The target peer sends out a query requesting reputation information about itself. This query can either be broadcast to all peers in the system with a specified hop count or can only be sent to peers that the peer has interacted with before.
- When a peer gets this request, it responds back with a message containing reputation information about the target.
- The target peer collects all these messages and presents them to the subject peer.
- The subject peer may choose to confirm presented information by directly querying some of those recommenders whose messages were presented by the target peer.

The same procedure can also be used to eliciting group information if the application consists of groups.

Hop count. Typically, it would be best to broadcast queries for trust information to all peers in the decentralized system in order to acquire as much trust data as possible. However, such a message flooding can consume a lot of bandwidth and possibly slow down communication between peers as the size of the system increases. Therefore, trust models may specify a hop count within the query message to limit the number of peers to whom the query is forwarded. Every time the query is forwarded to another peer, the hop count value of the query message is decremented by 1 until the hop count limit is reached. The value of the hop count thus decides the number of peers whom the query message will be forwarded to, and thus plays an important role in trust determination. An optimal value of the hop count value can be determined based on the trust model, the nature of the application, and the extent to

which peers in the system are connected.

Messages. The Communication sub-model also facilitates the specification of messages that are used by peers to exchange trust information. These messages could be queries for trust information as well as responses to those queries. Additionally, they could be trust revocation messages as in the case of DTM [1] or confirmation messages as in the case of XREP [8]. Each message type must have a name that identifies its type. A message type could also optionally include any of the following elements: the identity of the message sender, the identity of the peer for whom the message is intended, the identity of the target peer about whom trust information is being queried or reported, the trust value of the target peer as reported by the sender, a time-to-live value that specifies how long the trust value is valid, a hop count specified by the 'hop count' attribute of the Communication sub-model as described above, and contexts in which the trust information is being sought or reported. A trust model could also specify whether a message type uses any authentication mechanism.

5.3 Computation Sub-Model

For the determination of trustworthiness of a target peer, two sources of information are used by the subject peer: the subject peer's own past experience (P) and information received from recommenders (R). This information not only includes data related to interaction with the target peer, but also any structure or group information that may have a bearing on the target peer's reputation. We first briefly discuss the two sources of trust information and then describe some of the ways this information can be combined to compute trust.

5.3.1 Personal Past Experience

The subject peer can use its own personal past experience with the target peer in the computation of the trustworthiness of the target peer. The personal experience may be used to varying degrees in this computation. Sometimes, depending upon the application, a subject peer may decide to solely rely on its own personal experience. At other times, the personal experience may be variously combined with reputation information received from other peers to determine the trustworthiness of the target peer.

5.3.2 Recommendations

Recommendations refer to the trust information received from other peers in the system. The information reported by these recommenders may be given varying importance according to criteria such as their reputation as recommenders, expertise in a particular area etc. A subject peer must also determine information about the reputation of these recommenders because their opinions have a significant bearing on the subject peer's decision-making.

5.3.3 Trust Computation Mechanisms

In this section, we list some common ways in which trust information can be combined to provide a single trust value. This trust information may either be based on personal experience or recommendations or both.

Average. This scheme applies to both personal experience and recommendations. Trust impressions from previous personal experiences can be directly averaged to arrive at a single trust value for P. Similarly, received recommendations can be averaged to compute value for R.

Weighted average. In this scheme, recommendations are combined with the reputation of the corresponding recommenders as weights in order to compute a weighted average trust value for R. A more complicated weighted average scheme could entail involving chains of recommenders and combining their recommendations suitably weighted by their corresponding reputations as recommenders.

Trusting unknowns. If there is no trust data about a recommending peer, a trust model can decide the extent to which this peer's recommendations can be trusted. A very liberal trust model may decide to trust new unknown peers completely. On the other hand, a more conservative trust model may either decide to bestow little or no trust on such unknown peers. A trust model may also decide to not pass any judgement on an unknown peer's trustworthiness and just mark it as "unknown". In order to capture this ability of trust models, 4C includes a parameter called 'degree of trust in unknowns' that specifies to what extent the trust model trusts unknown peers.

Using the period of validity. A trust model can also choose to use the time-to-live attribute of the reputation information in the computation of both personal perception-based trust and recommendation-based trust. Specifically, reputation values whose time-to-live attribute has expired or exceeded a certain threshold can be either discarded or assigned a low weight during trust computation. This ensures that newer information is given more importance than older information. This also permits a recent change in behavior of a peer to be reflected sooner in its reputation that may be desired by some trust models.

Interaction limit. A trust model may also limit the trust data used for trust evaluation to a specific number of recent interactions. This is also referred to as varying the experience window which refers to the amount of history a peer maintains for calculating reputation [14]. In order to capture this, 4C includes a parameter called 'interaction limit' that specifies the percent of all interactions that are included during trust computation. Further, weights for recent and older interactions can be assigned while specifying the trust model. These weights can then be used during the trust computation to specify how much value old interactions should be given. For this, 4C also includes a parameter called 'recent interaction weight' to specify the weight assigned to recent interactions during trust computation.

Context-based trust. The context of trust plays a significant role in determining both P and R. Contexts can be composed of sub-contexts [24], and if a peer were to be considered trustworthy in a particular context, he could be considered trustworthy in all the sub-contexts that form a subset of the context. To better illustrate this concept, consider the case, when a peer is considered trustworthy in the context of cars. It could imply that the peer can be considered equally trustworthy when it comes to car engines, car tires, transmission, and other sub-contexts of a car.

Utilizing group information. If the decentralized application allows the formation of groups, group relationships can be used to infer trust among peers. For example, a peer belonging to a friendly group can be trusted to a greater extent than a peer belonging to an inimical group.

Trusting certain recommenders more. Trust models also typically specify a threshold value which helps determine who is considered trustworthy. Peers with trust values above this threshold value are considered trustworthy and those with values below or equal to the threshold are considered untrustworthy. This threshold is represented by the 'trust threshold' parameter. This parameter can be used by a trust model to only listen to those recommenders whose trust value is greater than the trust threshold. In such a case, a trust model may combine the recommendations of these trusted recommenders using either a average-based or weighted average-based scheme.

5.3.4 Evaluating Trust

Finally, a trust model combines its past personal experiences and received recommendations in a suitable manner to compute a trust value. Let 'w' and '1-w' where (0<= w <= 1) represent the weights in which the personal impressions and recommendations are to be combined, then the actual computation for trust becomes

Trust Value = w*P + (1-w)*R

P represents a single trust value based on personal experiences that accounts for the possible time degradation of trust information, the trust context, and group reputation. Similarly R represents a single trust value based on received recommendations that accounts for the trustworthiness of the recommenders in addition to time degradation, trust context and group reputation. For the case, when a peer decides to only rely on its own personal experience and not trust any recommenders, value of 'w' becomes 1 and the trust value reduces to the value of P. If the value of 'w' is 0.5, the trust value becomes a simple average of P and R.

This final computed trust value is compared against the specified 'trust threshold' to determine if the target peer can be trusted or not. If the final trust value is greater than the trust threshold, the trust model considers the target peer trustworthy otherwise it considers the target peer untrustworthy.

5.4 Counteraction Sub-Model

Once a subject peer determines that the target peer can be trusted to a suitable level, it proceeds with interaction with the target peer. The result of the interaction may have a significant bearing on the reputation of the target peer, recommenders who recommended the target peer, and existing groups as perceived by the affected subject peer. These impressions of the subject peer are added to its personal experience repository, so that they can be utilized suitably the next time while computing trust. Once a subject peer has updated its perception of the reputation of concerned peers and groups in the system, it can choose to inform other peers in the system about its recent perceptions. A trust model can enable this in two ways namely active dissemination and passive dissemination.

Active Dissemination. The peer can choose to actively propagate any information it considers very important to other peers in the system by sending explicit updates or warning messages.

Passive Dissemination. Alternately, a peer may decide to hold on to this perception, and provide updated information in the form of recommendations only when explicitly queried by other peers in the system. Upon receiving new reputation information, peers can choose to update their existing perceptions.

6. EVALUATION

Our evaluation for the 4C framework can be categorized into three main efforts. The first effort is to study and evaluate whether existing decentralized reputation models can be successfully expressed using the 4C framework. The second effort consists of building a 4C-based editor and using it to generate XML-based trust model descriptions. The third effort is to examine how these automatically generated trust model descriptions can be used to design and build trust-centric decentralized applications. These efforts are discussed below in further detail.

6.1 Expressing Reputation Models

In this section, we describe how the 4C framework can be used to express existing reputation models. We consider three existing reputation-based trust models and for each model, we first provide a brief description and then discuss how it can be expressed in terms of the four sub-models of the 4C framework.

6.1.1 Distributed Trust Model

Description. In the Distributed Trust Model [1] (DTM), a trust relationship is always between exactly two entities, is non-symmetrical, and is conditionally transitive. Mutual trust is represented as two distinct trust relationships. A peer uses two types of trust data for determining trust - its own personal experience and recommendations provided by other peers in the system. However, recommendations are utilized for computing trust only when there are no direct trust experiences with a particular peer. DTM identifies two different types of trust relationships. A *direct* trust relationship is when one peer trusts another. But if a peer trusts another peer to give recommendations about a third peer's trustworthiness, then there is a *recommender* trust relationship between the two.

DTM uses discrete integral trust values to represent the trustworthiness of peers with -1 representing distrust, 0 representing lack of knowledge, 1-3 representing increasing trust and 4 representing complete trust. Trust categories are used by peers to classify trust towards other peers depending upon which aspect of that entity is under consideration. Trust categories essentially specify the context for the trust. A reputation is defined as a tuple consisting of a peer's name, the trust category and the specific trust value. DTM uses digital identities and signature-based authentication to authenticate messages. DTM uses three types of messages to communicate trust information between peers. These are *Request for Recommendation*, *Recommendation*, and *Refresh* messages. When a peer needs a service offered by another peer for the first time (no prior transactions between the

two), the peer sends out a *Request for Recommendation* message to the peers it trusts as recommenders. These recommender peers can respond by sending *Recommendations* if they know the target peer, else they forward the request to other peers whom they trust as recommenders. Since the opinion of peers may change over time, recommendations are valid only for a limited time. When recommendations expire or the trust values associated with them change, they are updated using *Refresh* messages. *Refresh* messages are also used to revoke *Recommendations* by sending *Refresh* messages with trust values 0.

4C Realization. The Content sub-model for the DTM includes the context, discrete values of trust, and the period of validity. Trust determination is made on the basis of personal past experiences and recommendations received from recommenders. The Computation sub-model combines recommendations using a weighted average scheme. The weights represent the extent to which the subject peer trusts the recommenders. The Counteraction sub-model uses revocation of recommendations as an active mechanism to spread updated information about the target peer. The Communication sub-model uses the non-access-based communication mechanism.

6.1.2 REGRET Model

Description. REGRET [24] includes social relationships between peers in its reputation model. REGRET adopts the stance that the overall reputation of a peer is an aggregation of different pieces of information. REGRET is based upon three dimensions of reputation - individual, social, and ontological. It combines these three dimensions to yield a single value of reputation. When a member peer depends only on its direct interaction with other members in the society to evaluate reputation, the peer uses the *individual dimension*.

If the peer also uses information about another peer provided by other members of the society it uses the *social dimension*. The social dimension relies on group relations. In particular, since a peer inherits the reputation of the group it belongs to, the group and relational information can be used to attain an initial understanding about the behavior of the peer when direct information is unavailable. Thus, there are three sources of information that help peer "A" decide the reputation of a peer "B" - the individual dimension between A and B, the information that A's group has about B called the *Witness* reputation, and the information that A's group has about B's group called the *Neighborhood* reputation. Figure 1 illustrates these various reputation relationships.

REGRET believes reputation to be multi-faceted and presents the following example as illustration - the reputation of being a good flying company summarizes the reputation of having good planes, the reputation of never losing luggage, and the reputation of serving good food. In turn, each of these reputations may summarize the reputations of other dependent factors. The different types of reputation and how they are combined to obtain new types of reputation is defined by the *ontological dimension*. Clearly, since reputation is subjective, each peer typically has a different ontological structure to combine reputations and has a different way to weigh the reputations when they are combined.

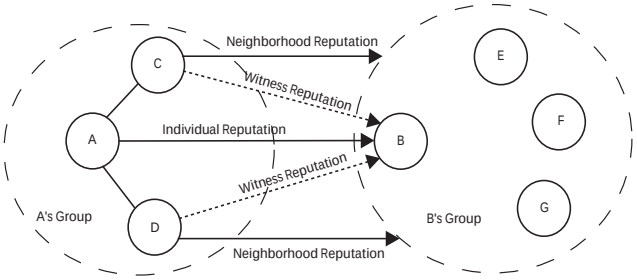


Figure 1. Individual and Social Reputation in REGRET

REGRET stores reputations in the form of impressions. An impression is the subjective evaluation made by an agent on a certain aspect of a outcome. An impression consists of the subject and target peers, the outcome, the aspect of the outcome being rated, the time of generation of the impression, and the subjective rating of the outcome aspect from the subject peer's perspective expressed as a continuous value.

4C Realization. The Content sub-model for REGRET includes the use of both recommendations and group reputations. Outcome aspects map to trust contexts and reputation is expressed as continuous values. The Communication sub-model uses the non-access based communication mechanism. The Computation sub-model uses group relationships to determine the trustworthiness of the target peer. It also employs context-based trust analysis through the use of ontological reputation dimension. The Counteraction sub-model uses a passive dissemination mechanism that provides reputation information only upon request by a subject peer.

6.1.3 Complaint-based Model

Description. In a complaint-based model, negative reputation information is encapsulated and stored as a "complaint". In such a model, peers do not store information about successful interactions or trustworthy peers, but rather record their negative experience in the form of complaints against interacting peers. The complaint-based trust model is based on binary trust. Peers perform transactions and if a peer cheats in a transaction, it becomes untrustworthy from a global perspective. Upon receiving a request, this information in the form of a complaint about dishonest behavior can be sent to other peers.

When a peer wants to evaluate the trustworthiness of a target peer, it first searches its own history to locate any previous complaints registered by itself. It can also query other peers for other existing complaints about the target peer. Upon receiving a query, peers that have the required complaints respond accordingly. Since these peers themselves can be malicious their trustworthiness needs to be determined. Consequently, queries for complaints about these peers are sent out by the original peer and so on. In order to prevent the entire network from being explored, which would become expensive in a large system, if similar data about a specific peer is received from a sufficient number of peers, no further checks are carried out. Complaints received from other peers are included in the determination of the target peer's trustworthiness.

This kind of complaint-based scheme has been adopted by trust management systems such as the one based on the P-Grid data structure [3].

4C Realization. The Content sub-model in the complaint-based model uses complaints which are essentially binary expressions of trust. Group relationships are not considered in this model. The Communication sub-model is pretty simple and peer interactions are limited to filing and querying complaints. It essentially uses a non-access-based communication mechanism. The Computation sub-model uses a simple weighted average algorithm to compute the negative reputation of the target peer. The peer queries for complaints about recommenders and uses received responses to determine whether complaints created by those recommenders can be trusted. This can be used to determine whether the target peer should be trusted. The Counteraction sub-model in the complaint-based model uses a passive mechanism to disseminate complaints to other peers. Knowledge about complaints is obtained through explicit requests for complaints.

6.2 4C Editor

6.2.1 4C Trust Model Schema

XML provides an ideal platform for enabling extensible trust model descriptions. Further, there are several existing off-the-shelf XML-based tools that help create, edit, validate, and parse XML-based documents. Using these tools prevents the need for building specific custom-made editors and parsers. Therefore, we used XML-based technology to enable the description and expression of trust models. In particular, we have created a simple XML-based trust model schema that defines a set of rules to which XML-based trust model descriptions must conform. These set of rules in the trust model schema help describe trust models in terms of the four sub-models of the 4C framework.

We used XMLBeans [22] to generate Java bindings from the our trust model schema. XMLBeans allows access to XML instances through get and set accessor methods similar to those in JavaBeans. XMLBeans compiles the trust model schema and generates a set of Java interfaces that can be used to process XML trust model instances that conform to the schema. XMLBeans allows changes made through these Java interfaces to modify the underlying XML representation.

6.2.2 4C Editor

We have built a Java-based editor for describing reputation-based trust models using the 4C framework. This editor enables a user to create a decentralized reputation-based trust model using a Graphical User Interface (GUI). The GUI is divided into four main interfaces, one for each sub-model, and mirrors the trust model schema used. In each of the four interfaces of the 4C editor, the user can browse through various options corresponding to the specific sub-model and make appropriate selections depending upon the requirements of the trust model being generated. The 4C editor thus guides the user through the different sub-models and finally generates an XML-based description of the trust model that conforms to the trust model schema. Figure 2 shows a snapshot of the Content sub-model GUI for the Distributed Trust Model. In the center, the expression of reputation values, time-to-live, and the context tree is visible. The content of the four sub-models is displayed in the text window to the far right. Table 1 shows the actual XML-based description of the Distributed Trust Model that is generated by the 4C editor.

6.3 Building A Trust-centric Application

We have used trust model descriptions generated by our 4C editor to assist in the design and development of trust-enabled applications. Specifically, we have designed a tool called the PACE Support Generator (PSG) that uses the 4C trust model descriptions created by the 4C editor to produce trust components and utility classes for integration within the PACE architectural style. PACE stands for Practical Architectural approach for Composing Egocentric trust [27]. PACE is an architectural style for trust management in decentralized applications and provides a set of principles that guide the integration of trust components within the architecture of a decentralized peer. PACE describes the trust-enabled architecture of each peer using the xADL 2.0 architecture description language [9].

Components in PACE communicate using request and notification messages. Integrating a trust model within the PACE architecture requires the creation of certain trust model-specific messages that will traverse the architecture. Further, in order to facilitate the manipulation of these messages, appropriate trust-related data structures need to be created. Finally, expressions for the evaluation of trust need to be specified. These requirements are not specific to any trust model, in fact, they are common to all trust models. Therefore, PSG provides a GUI-based interface to assist a user in the creation of trust model-specific data structures, messages, and formulae.

Figure 3 illustrates the architecture of PSG. The Data Structure Creator component allows the specification of trust-related data structures. The Message Creator component allows the specification of messages underlying the trust model and creates messages based upon the data structures created by the Data Structure component. The Trust Formula Creator component facilitates the specification of the trust formula using content enumerated in the data structures and messages.

The PACE Support Creator (PSC) component is responsible for using the elements generated by the other three components to generate trust-related components and utility files. PSC first stores the data structure, messages, and reputation formulas created by the other components into a PACE support description file. Doing so provides the ability to modify the file later on (e.g. for optimization) and generate corresponding components. Next, PSC uses this file to automatically produce components, utility classes and interfaces for the trust model. Specifically two main components are generated by PSC and added to the application layer of the PACE architecture. The first component is the Message Manager, which encapsulates all the operations associated with the management of messages for the trust model. These operations include the exchange of trust information between peers and saving or retrieving messages. The second component is the Trust Evaluator component. This component handles trust computation at the application layer; therefore, it needs to communicate directly with the Trust Manager. Upon

Table 1. 4C-based description of Distributed Trust Model

```
<?xml version="1.0" encoding="UTF-8"?>
<trus:trustModel xmlns:trus="http://www.ics.uci.edu/~sgirish/
TrustSchema.xsd">
<trus:contentModel>
<trus:individual>
<trus:individualExpression>
<trus:discrete>
<trus:startValue>-1.0</trus:startValue>
<trus:endValue>4.0</trus:endValue>
<trus:incrementValue>1.0</trus:incrementValue>
</trus:discrete>
</trus:individualExpression>
<trus:individualTTL>true</trus:individualTTL>
<trus:individualContext>
<trus:contextTree><![CDATA[<?xml version="1.0" encoding="UTF-
8"?>
<java version="1.5.0_06" class="java.beans.XMLDecoder">
 <object class="javax.swing.tree.DefaultTreeModel">
  <object class="javax.swing.tree.DefaultMutableTreeNode">
   <void property="userObject">
    <string>CRASH data</string>
   </void>
   <void method="add">
    <object class="javax.swing.tree.DefaultMutableTreeNode">
     <void property="userObject">
      <string>Medical data</string>
     </void>
     <void method="add">
      <object class="javax.swing.tree.DefaultMutableTreeNode">
       <void property="userObject">
        <string>Required resources</string>
       </void>
      </object>
     </void>
     <void method="add">
      <object class="javax.swing.tree.DefaultMutableTreeNode">
       <void property="userObject">
        <string>Hospital location data</string>
       </void>
      </object>
     </void>
    </object>
   </void>
   <void method="add">
    <object class="javax.swing.tree.DefaultMutableTreeNode">
     <void property="userObject">
      <string>Crisis location data</string>
     </void>
    </object>
   </void>
   <void method="add">
    <object class="javax.swing.tree.DefaultMutableTreeNode">
     <void property="userObject">
      <string>Crisis victim data</string>
     </void>
    </object>
</void>
  </object>
 </object>
</java>
]]></trus:contextTree></trus:individualContext>
</trus:individual>
</trus:contentModel><trus:communicationModel>
<trus:communicationType>Subject-initiated</trus:communicationType>
<trus:hopCount>None</trus:hopCount>
<trus:message>
<trus:messageName>Recommendation</trus:messageName>
<trus:sender>true</trus:sender>
<trus:receiver>true</trus:receiver>
<trus:target>true</trus:target>
<trus:value>true</trus:value>
<trus:timeToLive>true</trus:timeToLive>
<trus:hopCount>false</trus:hopCount>
<trus:context>true</trus:context>
<trus:authentication>true</trus:authentication>
</trus:message>
<trus:message>
<trus:messageName>RRQ</trus:messageName>
<trus:sender>true</trus:sender>
<trus:receiver>false</trus:receiver>
<trus:target>true</trus:target>
<trus:value>false</trus:value>
<trus:timeToLive>true</trus:timeToLive>
<trus:hopCount>false</trus:hopCount>
<trus:context>true</trus:context>
<trus:authentication>true</trus:authentication>
</trus:message>
<trus:message>
<trus:messageName>Refresh</trus:messageName>
<trus:sender>true</trus:sender>
<trus:receiver>true</trus:receiver>
<trus:target>true</trus:target>
<trus:value>true</trus:value>
<trus:timeToLive>true</trus:timeToLive>
<trus:hopCount>false</trus:hopCount>
<trus:context>true</trus:context>
<trus:authentication>true</trus:authentication>
</trus:message>
</trus:communicationModel>
<trus:computationModel>
<trus:weight>1.0</trus:weight>
<trus:threshold>0.0</trus:threshold>
<trus:trustInUnknowns>0</trus:trustInUnknowns>
<trus:interactionLimit>None</trus:interactionLimit>
<trus:interactionWeight>None</trus:interactionWeight>
<trus:personalAlgorithm>Average</trus:personalAlgorithm>
<trus:recommendationAlgorithm>Weighted Average</trus:recommenda-
tionAlgorithm>
<trus:trustFormula>PPE and RECOM</trus:trustFormula>
</trus:computationModel>
<trus:counteractionModel>
<trus:counteractionType>Proactive</trus:counteractionType>
<trus:counteractionMessages>Refresh</trus:counteractionMessages>
</trus:counteractionModel>
</trus:trustModel>
```

generating these components, PSC then modifies the architecture description accordingly to include these two new components. PSC also modifies the Trust Manager in the PACE architecture to include the algorithm that computes trust. Finally, PSC creates relevant utility classes and interfaces along with their data members and generic methods according to GUI-based user specifications.

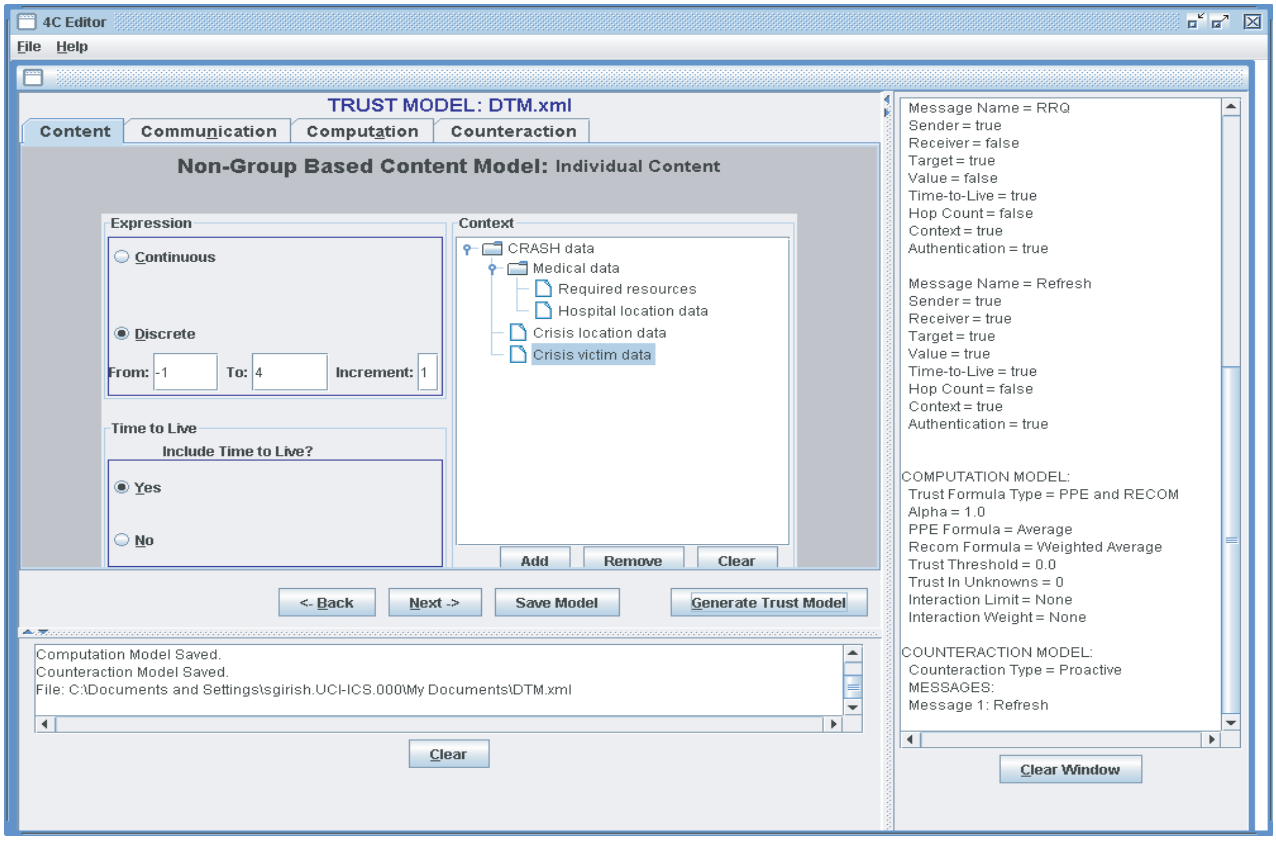


Figure 2. Snapshot of Content sub-model for Distributed Trust Model in 4C editor

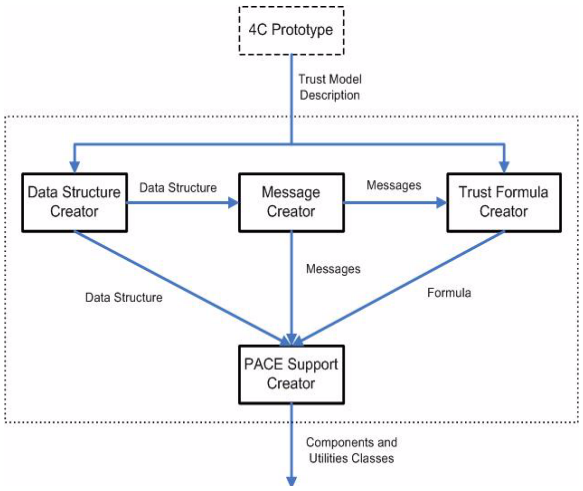


Figure 3. PACE Support Generator Architecture

We have used PSG to build a prototype of a trust-enabled decentralized emergency response application called CRASH. CRASH stands for Crisis Response and Situation Handling, and consists of independent agencies working together in a crisis situation. Specifically, we have used the XML-based description of the Distributed Trust Model (DTM) with PSG to help incorporate DTM into the PACE-based architecture of each CRASH entity. A more detailed explanation of the PSG tool and how we used it to incorporate DTM into the CRASH application can be found in [12].

We have evaluated our DTM-enabled CRASH implementation by subjecting it to various threat scenarios typical to a decentralized system[28]. These scenarios include cases where an agency tries to impersonate another, or where one or more agencies purposely lie about the trustworthiness of other agencies, and so on. We observed that the incorporation of DTM within the CRASH application helped detect and guard against some of these attacks. In particular since DTM uses unique digital identities and authentication mechanisms, CRASH entities could easily detect impersonation attacks. Similarly, the use of explicit revocation messages helped to warn other entities in the system about malicious entities.

PSG reduces the cost associated with the design and implementation of a trust model by not only automatically generating the necessary trust-centric components but also placing those components appropriately in the PACE architecture of the system. The basic PACE framework provides a skeleton Trust Manager component that is responsible for managing and computing trust information. Based on the XML description of the trust model, PSG modifies this Trust Manager component to automatically add the necessary implementation required for the computation of trust. Furthermore, PSG supplies all the necessary classes, interfaces, and generic methods that help implement new and existing trust-related components. These generic methods

automate the creation of request and notification messages based on class instances. The task of designers is thus reduced to creating instances of the generated classes and making use of the generic methods.

7. FUTURE WORK

The 4C framework provides an excellent starting point for further exploration of different types of decentralized reputation-based trust models and their various characteristics. We believe our continuing study and investigation of different types of trust protocols and algorithms will enable us to continually refine the 4C framework. The 4C trust model schema and the 4C editor currently include only the general characteristics of trust models. For example, the Communication sub-model generalizes the communication mechanisms into access and non-access based mechanisms. It does not specify the actual messages being exchanged between peers because these messages will differ from one trust model to another. However, every trust model will still use either an access-based or a non-access based type of communication mechanism. Similarly, the Computation sub-model does not specify how exactly time degradation is applied to the computation of trustworthiness because it is trust-model specific and may vary with trust models. In the future, we plan to examine how these varying characteristics of different reputation models can be effectively captured using the trust model schema to enable a richer representation for reputation models.

Peers in decentralized applications are vulnerable to potential attacks perpetrated by malicious peers. While trust models are essentially supposed to protect peers from such attacks, we have found that existing models either provide little or no safeguards against most of these threats [29]. There is a need for new kinds of trust models that are focused primarily on protecting peers from these malicious attacks. Towards this end, we believe the 4C framework provides fertile ground for the exploration of new trust models in the future. For example, consider a new trust model that is similar to DTM but uses continuous trust values instead of discrete trust values. This gives the new model the ability to better express and compare trustworthiness of peers enabling it to better detect malicious peers.

Further, the various concepts currently included in the 4C framework are common to several trust models and can provide a starting step towards exploring mechanisms for trust model translations. The use of such translation mechanisms will enable peers with different trust models to interoperate in a seamless fashion with each other and provide greater application flexibility.

8. CONCLUSIONS

Current trust literature has focused mainly on developing trust and reputation models but lacks a common understanding of what a trust model is and what are the elements that constitute a trust model. There has also been little work devoted towards the exploration of a framework to express decentralized trust and reputation models. Towards addressing these shortcomings, this paper presents our definition of a trust model and describes a generic extensible framework, called 4C, to express decentralized reputation models. We believe the 4C framework serves as an initial but important step towards the creation of a standard trust model framework in the future.

There are several benefits of the 4C framework. The first benefit is that it facilitates a better understanding of reputation models and permits a comparison of their characteristics. For example, consider DTM and the REGRET model. The Counteraction sub-model helps expose the fact that DTM employs an active dissemination mechanism that uses explicit revocation messages whereas the complaint-based model uses a passive dissemination mechanism. Similarly, the Content sub-model reveals the presence of application structure expressed through the use of groups in REGRET and its absence in DTM.

The second benefit of the 4C framework derives from the use of the 4C editor that enables a user to create a trust model description which can be used in turn to support the design and development of trust-enabled decentralized systems. Additionally, since the 4C framework and editor along with the PACE Support Generator facilitate the quick generation and integration of new trust models into decentralized applications, it allows users to potentially play around with different trust models and ultimately choose a model that best fits their needs.

The third benefit stems from the fact that the 4C framework is described using XML-based schemas. The use of XML to describe trust models not only enables a user to leverage off-the-shelf XML tools but also makes the 4C framework flexible and extensible in order to provide a richer expression of trust models in the future.

9. ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 0524033.

10. REFERENCES

[1] Abdul-Rahman, A. and Hailes, S. A Distributed Trust Model. In *Proceedings of the New Security Paradigms Workshop.* Langdale, Cumbria UK, 1997.

[2] Abdul-Rahman, A. and Hailes, S. Supporting trust in virtual communities. In *Proceedings of the Hawaii International Conference on System Sciences.* Maui, Hawaii, Jan 4-7, 2000.

[3] Aberer, K. and Despotovic, Z. Managing Trust in a Peer-2-Peer Information System. In *Proceedings of the Conference on Information and Knowledge Management.* Atlanta, Georgia, November 5-10, 2001.

[4] Blaze, M., Feigenbaum, J., et al. Decentralized Trust Management. In *Proceedings of the IEEE Symposium on Security and Privacy.* p. 164-173, May, 1996.

[5] Cahill, V., Gray, E., et al. Using Trust for Secure Collaboration in Uncertain Environments. *IEEE Pervasive Computing Mobile and Ubiquitous Computing.* 2(3), p. 52-61, August, 2003.

[6] Capra, L. Engineering Human Trust in Mobile System Collaborations. In *Proceedings of the 12th International Symposium on the Foundations of Software Engineering (SIGSOFT 2004/FSE-12).* Newport Beach, California, USA, November, 2004.

[7] Chhabra, S., Damiani, E., et al. A protocol for reputation

management in super-peer networks. In *Proceedings of the 15th International Workshop on Database and Expert Systems Applications.* p. 979-983, Zaragoza, Spain, 30 Aug-3 Sept, 2004.

[8] Damiani, E., di Vimercati, S.D.C., et al. A Reputation-Based Approach for Choosing Reliable Resources in Peer-to-Peer Networks. In *Proceedings of the 9th ACM Conference on Computer and Communications Security.* Washington DC, November, 2002.

[9] Dashofy, E.M., Hoek, A.v.d., et al. A Highly-Extensible, XML-Based Architecture Description Language. In *Proceedings of the Working IEEE/IFIP Conference on Software Architecture (WICSA 2001).* Amsterdam, The Netherlands, August 28-31, 2001.

[10] Deutsch, M. Cooperation and Trust: Some Theoretical Notes. In *Nebraska Symposium on Motivation*, Jones, M.R. ed. Nebraska University Press, 1962.

[11] Deutsch, M. *The Resolution of Conflict: Constructive and Destructive Processes*. Yale University Press: New Haven, 1973.

[12] Diallo, M., Suryanarayana, G., et al. *Tool Support for Incorporating Trust Models into Decentralized Applications.* Institute for Software Research, UC Irvine, Report UCI-ISR-06-04, April, 2006.

[13] Dragovic, B., Kotsovinos, E., et al. XenoTrust: Event-based distributed trust management. In *Proceedings of the Second International Workshop on Trust and Privacy in Digital Business.* Prague, Czech Republic, Sep, 2003.

[14] Folkerts, J. *A Comparison of Reputation-based Trust Systems.* Masters Thesis. Dept. of Computer Science, Rochester Institute of Technology, 2006.

[15] Gambetta, D. *Trust*. Gambetta, D. ed. Blackwell: Oxford, 1990.

[16] Grandison, T. and Sloman, M. A Survey Of Trust in Internet Applications. *IEEE Communications Surveys.* 3(4), December, 2000.

[17] Gray, E., O'Connell, P., et al. *Towards a Framework for Assessing Trust-Based Admission Control in Collaborative Ad Hoc Applications.* Distributed Systems Group, Department of Computer Science, Trinity College, Report TCD-CS-2002-66, 2002.

[18] Josang, A. and Ismail, R. The Beta Reputation System. In *Proceedings of the 15th Bled Electronic Commerce Conference.* Bled, Slovenia, June 17-19, 2002.

[19] Kamvar, S., Schlosser, M., et al. The EigenTrust Algorithm for Reputation Management in P2P Networks. In *Proceedings of the WWW.* Budapest, Hungary, May 20-24, 2003.

[20] Lee, S., Sherwood, R., et al. Cooperative peer groups in NICE. In *Proceedings of the IEEE Infocom.* San Francisco, USA, April 1-3, 2003.

[21] Marsh, S. *Formalising Trust as a Computational Concept.* Thesis. Department of Mathematics and Computer Science, University of Stirling, 1994.

[22] Project, A.X. *Apache XMLBeans http://xmlbeans.apache.org*. <http://xmlbeans.apache.org>, Website, 2003.

[23] Pujol, J., Sanguesa, R., et al. Extracting reputation in multi agent systems by means of social network topology. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems.* Bologna, Italy, July 15-19, 2002.

[24] Sabater, J. and Sierra, C. REGRET: A Reputation Model for Gregarious Societies. In *Proceedings of the 4th Workshop on Deception, Fraud and Trust in Agent Societies.* Montreal, Canada, 2001.

[25] Suryanarayana, G., Erenkrantz, J.R., et al. PACE: An Architectural Style for Trust Management in Decentralized Applications. In *Proceedings of the 4th Working IEEE/IFIP Conference on Software Architecture.* p. 221-230, Oslo, Norway, June, 2004.

[26] Suryanarayana, G. and Taylor, R.N. *A Survey of Trust Management and Resource Discovery Technologies in Peer-to-Peer Applications.* UCI Institute for Software Research, Technical Report UCI-ISR-04-6, July, 2004.

[27] Suryanarayana, G., Erenkrantz, J.R., et al. An Architectural Approach for Decentralized Trust Management. *IEEE Internet Computing Special Issue on Ad Hoc and P2P Security.* 9(6), p. 16-23, Nov-Dec, 2005.

[28] Suryanarayana, G., Diallo, M., et al. Architectural Support for Trust Models in Decentralized Applications. In *Proceedings of the 28th International Conference on Software Engineering.* Shanghai, China, May 20-28, 2006.

[29] Suryanarayana, G. and Taylor, R.N. *TREF: A Threat-centric Comparison Framework for Decentralized Reputation Models.* Institute for Software Research, UC Irvine, Report UCI-ISR-06-2, Jan., 2006.

[30] Tan, Y.-H. and Thoen, W. Towards a Generic Model of Trust for Electronic Commerce. *International Journal of Electronic Commerce.* 5(2), p. 61-74, 2001.

[31] Yu, B. and Singh, M.P. A social mechanism of reputation management in electronic communities. In *Proceedings of the Fourth International Workshop on Cooperative Information Agents.* p. 154-165, 2000.

[32] Zacharia, G. and Maes, P. Trust Management Through Reputation Mechanisms. *Applied Artificial Intelligence.* 14, p. 881-907, 2000.