

Object Trust Management for Information Quality Assurance in Virtual Organisations

Yanjun ZUO
Brajendra PANDA

Ensuring information quality in a virtual organisation is crucial in order for the participants to confidently use imported data. Information quality is measured by the extent to which a user trusts the information as accurate. The component-based scheme presented in this paper assumes that several versions exist for a given object. Each version represents a subject's opinion about the value(s) of the object, called object value(s). The presented model allows a user to evaluate the trustworthiness of a version of the given object based on how the object version was formed and how its components should be trusted. A label is associated with each object to provide available versions of the object and component information for each version. The trustworthiness of each component in our model can be used in calculating the primary trust value, which is calculated by direct experience of the evaluator, of a compound object version. The secondary trust value, calculated for each component using trusts assigned by other subjects to determine the final trust value of the compound object version, as explained in the paper, is more computationally efficient. But for that method to work, subject trust calculation is required. In addition to those methods for object trust calculation, heuristic approaches based on component information of a given object version are also presented to facilitate object trust evaluation.

- virtual organisation
- subject trust
- object and object version
- primary and secondary trust of object version
- component-based approach

1

Introduction

A virtual organisation is a general term and represents a decentralised system with a set of independent participants sharing resources according to a set of pre-defined rules. Examples of a virtual organisation include Grid systems, Peer-to-peer systems, virtual communities, and so on. Some common features of those virtual organisations are (1) self-organised by participants based on mutual interests and trusts, (2) without a single administrative authority, (3) resource sharing in a controlled and accountable manner, and (4) decentralised – decisions to offer and consume resources are made by each participant.

A virtual organisation offers various advantages for its participants. For a commercial virtual organisation, business partners can take advantage of each other's strengths and offer high-quality products and services at lower costs. A computing virtual organisation allows individual participants to use computing resources that they could not access otherwise. Government agents form a virtual organisation to share resources in order to cooperate on important issues and respond to emergent events such as a natural disaster or a terrorist attack. Academic institutions form a virtual organisation to conduct a large-scale scientific project.

Much of the research work on virtual organisation has focused on studying the behaviours of members of a virtual organisation and defining rules that govern resource usages [1, 2, 3]. Security and privacy issues have become primary concerns for the well-being of a virtual organisation recently. Existing virtual organisations provide mechanisms for user authentication and authorisation, confidential communications, and privacy preservation. The notations of trust and reputation have been applied in existing virtual organisations to address the scalability problem of traditional security mechanisms as those presented in [4, 5, 6, 7]. Several models have been developed for trust management ([8, 9, 10, 11, 12, 13], to cite a few). But none of the existing virtual organisations offers a complete solution to the problem of information quality, i.e. whether a piece of external information is correct and accurate. Low barriers to publishing information in a virtual organisation require novel mechanisms to verify the quality features of external information. The existing methods only provide a preliminary solution for information quality assurance. A user assesses a given piece of information based on the reputation and trustworthiness of the owner of the information. But that is not a reliable way to evaluate the quality of the information. A formal approach for ensuring the information

This work is based on an earlier work 'Component based trust management in the context of a virtual organisation', Proceedings of the 2005 ACM Symposium on Applied Computing (SAC'05), © 2005 ACM. doi.acm.org/10.1145/1066677

quality based on the intrinsic features of the given information is demanded to accommodate the growing interests for global information sharing.

Existing trust models focus at subject level and few of them have provided effective methods for a subject to evaluate the trustworthiness of a piece of information in term of its quality. In a virtual organisation, analysing the information's intrinsic features is important since eventually each participant decides whether the information is

trustworthy and hence it can be used. Focusing the study of information quality at object level gives a user higher confidence to use a piece of information since that information has been directly assessed. Making a decision only based on a subject's trustworthiness is not always reliable. We know that even the most honest people make mistakes. Compared to subject trust management, however, object trust evaluation has not attracted much attention. The primary focus of this paper is trust evaluation at object level. A labelling scheme has been developed to represent component information and facilitate object trust evaluation based on component-based approach.

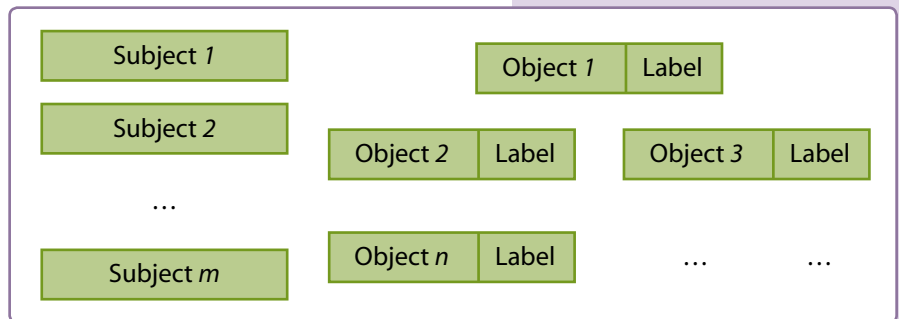
Figure 1 illustrates the data infrastructure of a virtual organisation. As a participant of a virtual organisation, a subject is an independent entity, producing and consuming information. The general term *information* has been used so far to denote an intangible entity such as a data item, a statement or other types of knowledge. Every piece of information expresses a subject's opinion (or idea) towards a topic or issue. In the following discussion, the term *object* is used to denote such a topic or issue. This notation (object) is chosen because it is frequently used together with the term *subject*. An object has a value or a set of values, called object value(s), representing the inherent features(s) of the object. For instance, current economic growth is considered as an object and its object value, i.e. a real number, represents how fast the economy is growing. A subject has its opinion about the object value(s) for a given object. The object values expressed by different subjects could be different. It is very likely that different subjects have different views on a given issue or topic. For example, different groups of economists may have used different analytical tools and collected different sets of data to calculate the economic growth rate. Hence, they have different opinions on this value. The term *object version* is used to represent such an opinion that a subject has on the object value(s) for the given object. In addition, the owner of an object version also supplies its confidence in the proposed object value, which is expressed as the trust that it puts on the proposed object value. There are reasons that the owner does not fully trust the object value that it has provided. The raw data collected by the version owner in producing the object version may be incomplete or it may contain errors as supplied by its sources. Information processing may introduce sampling errors. The owner of the object version may use approximation or assumption in producing the object version. All those reasons explain that the owner of the object version has a certain level of confidence in the version that it has provided. The trust level is used to represent this confidence.

An object can have multiple versions as provided by different subjects. To distinguish among available versions of a given object, the symbol $V_i^{(O)}$ is used to denote the i -th version of object O . Two types of object versions may exist in a virtual organisation as discussed below.

Definition 1. A *simple object version* is an atomic data entity, which is provided by a subject to represent its opinion about the object value(s) for a given object. No component information is required for a simple object version and it is considered as 'undividable'.

Definition 2. A *compound object version* is a composite data entity presenting the object value(s) for a given object and it is derived through a composing process from a set of other

Figure 1. Logical view of a virtual organisation depicting subjects, objects and labels



object versions, called its components. The composing process, also called the computational procedure, is expressed by a set of composing functions.

The following example will help clarify the idea of information composition. To compute the final grade in a course, a student's scores in the mid-term test, final test, and class project are considered. In this case, the final grade of the student in the course is a compound object version where as her scores in the mid-term, final test, and project constitute the three components. The composing functions, for instance, can be a weighted average on the three components.

Among all multiple versions of a given object, one version is called the official version of the object that is recognised to be of the highest quality. But a user still makes its own decision in selecting any version to use. However, when no better alternative is available, by default, a user selects the official version of a given object to use.

This work is based on an earlier work: 'Component Based Trust Management in the Context of a Virtual Organisation', in Proceedings of the 20th ACM Symposium on Applied Computing, Santa Fe, USA, © ACM, 2005. The remaining of the paper is organised in the following manner. [Section 2](#) introduces the component-based approach, which facilitates object trust evaluation. [Section 3](#) describes the labelling method and the standard format of a label. [Section 4](#) gives an example using multiple objects and their corresponding labels in a virtual organisation. The same example is also used in [Section 5](#) to illustrate object trust calculation. Section 5 proposes methods to calculate subject trust and object trust including primary and secondary trusts of an object version. In [section 6](#), two component-based heuristic approaches are provided as complementary methods for object trust calculation. [Section 7](#) concludes this paper.

2

Component-based approach for object trust management

Information processing is accumulative and recursive, e.g., some information is formed using others as its components. For instance, a public key encryption algorithm (e.g., RSA) uses such methods as large primary number generation and testing, key distribution, and one-way function (e.g., modular operation) as its building blocks. A software package uses various modules, library functions, and methods, which are incorporated to construct a complete working program. The Dow Jones Industrial Average summaries 30 stock prices in the average and divide it by a constant called 'divisor'. Information integration or derivation is one of the major forms of information processing in a data intensive system for science and commerce. In domains as diverse as global climate change, high-energy physics, and computational genomics, science is becoming increasingly dependent on the generation and reuse of massive amounts of data, a trend sometimes known as data-intensive science. An example is provided in Section 5 to further illustrate this idea.

Information composition can facilitate trust analysis for a given object version. If a user has a way to study the components of a compound object version as well as the computational procedures to compose the object version, the user would be able to more accurately evaluate the trustworthiness of the object version. Tracing components of an object version gives an evaluator more insights into the intergradient and intrinsic features of the object version. The trust value of an object version can be induced or determined by the trust values of its components and the composing logic to form it.

In this paper, trust evaluation is studied from two different perspectives, namely subject trust and object version trust. The former investigates the level of trust a subject places on another and the latter determines the amount of trust a subject puts on an object version in term of its quality. Next, we provide some definitions that formally describe subject trust and object version trust.

Definition 3. *Subject trust* is a binary relation between two subjects indicating how the first subject trusts the second.

Subject trust describes a general confidence that the trustor has on the trustee in term of its honesty, for example. Therefore, it is also called 'basic trust', which means it is independent of any context. Subject trust is reflexive but not symmetric.

Definition 4. *Object version trust* refers to the degree a subject trusts an object version in term of its quality, i.e. how correctly and accurately the object version describes the object value of its target object.

In the component-based approach, if a compound object version uses other object versions as its components, the compound version is said to be dependent on its components versions. Version dependency relationship is transitive. An object version, say $V_i^{(Ok)}$, is indirectly dependent on a version of a different object, say, $V_j^{(Ol)}$, if some intermediate object version exists such that $V_i^{(Ok)}$ is dependent on the intermediate version and the intermediate version is dependent on $V_j^{(Ol)}$. In this case, $V_j^{(Ol)}$ is called a sub-component of $V_i^{(Ok)}$.

The chain of version dependency relationships keeps track of components and sub-components for a given object version. Searching those component information generates a tree-like structure, called version dependency tree for the given object version.

Definition 5. The *version dependency tree* for $V_i^{(O)}$ represents a logical view of components and sub-components of $V_i^{(O)}$ and how they are used to form $V_i^{(O)}$.

We depict a version dependency tree as a special tree with two types of nodes, namely, *formula nodes* and *version nodes*. A formula node, denoted as a rectangle, represents the logical compositions used to form the respective compound object version from its components. A version node, denoted as a circle, represents an object version. Formula nodes and component version nodes appear alternatively along any path from the root of the tree to a leaf. The depth of a version dependency tree is defined as the number of formula nodes, or the number of component nodes (excluding the root), along the longest path from the root of the tree to a leaf. A version dependency tree is dynamic and changes as the component information of the corresponding object version is updated. Hence, for efficiency, a version dependency tree is not stored but constructed as needed by a subject based on relevant labels as defined in [Section 3](#).

Figure 2. An example of version dependency tree for $V(O)1$ with depth of 2

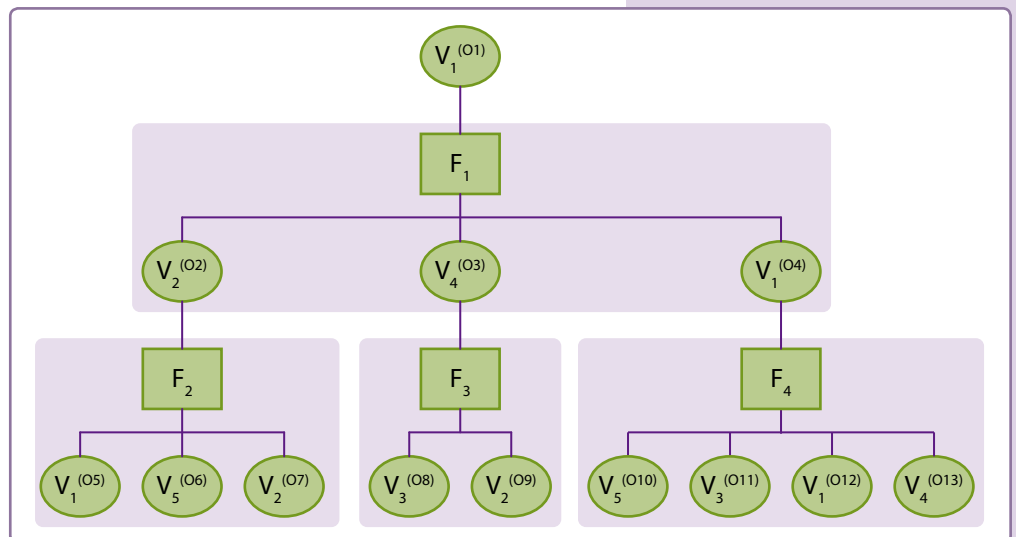


Figure 2 presents a version dependency tree for the object version $V_1^{(O1)}$, which is composed of three components, i.e. $V_2^{(O2)}$, $V_4^{(O3)}$, and $V_1^{(O4)}$. The composing functions, denoted by F_1 , have been used to form $V_1^{(O1)}$ by integrating these three components. Each component can be further tracked down to their components as well as the corresponding composing functions.

A version dependency tree is useful in recursive evaluation of the trust value of a given object version. In this recursive evaluation process, the trust value of a compound object version is determined based on the trust values of its components. The trust value of each of its components is further determined based on the trust values of its components. This 'divide and conquer' style evaluation continues until some object versions, which were considered trustworthy, are encountered. Since this type of recursive evaluation requires intensive computations, alternative ways for object trust evaluation will be discussed in [Section 5](#).

3

The labelling method

Component information is essential in evaluating the trustworthiness of an object version. There must be a standard method to formally express the versions of an object and component information for each version. This will help all subjects in the corresponding virtual organisation interpreting this information precisely. A label is developed and used for this purpose.

The concept of a label has been used in other literatures. In [14] the authors defined a label as a user-supplied program annotation, which describes the allowed methods of flow of information in a program. In their notations, a label represents meta-data that controls flow of information. In our model, we define a label in the following way.

Definition 6. A label associated with an object is a set of metadata representing information about available versions of an object and the component information for each version.

A label for an object version contains information such as the owner, i.e. the producer, of the version, the object value of the object viewed by the version's owner, a trust value of the proposed object value estimated by the version's owner, the version's components, and composing functions used to form the version.

Syntactically, a label for an object O is represented in the format $L = \{OV, RV\}$, where OV stands for official version and RV represents recommended versions. OV of a label L is denoted as $L(OV)$ and RV as $L(RV)$ respectively. The format of a label is visualised in Figure 3 (a). The format of $L(OV)$ and $L(RV)$ are discussed next.

- (1) $L(OV)$ is the set of 'official' entries, which together describes the official version of the object associated with the label. The first entry, which is about the official version of O , represents the owner of this official version, the object value for the object as provided by the owner, and the trust value for this object value for its owner. Each of the following entries (except the last one) represents a component version that is used to form the official version. The last entry, F , is a set of composing functions used to integrate all the components to get this official version. Each entry except the last one is in the format of E as shown below.

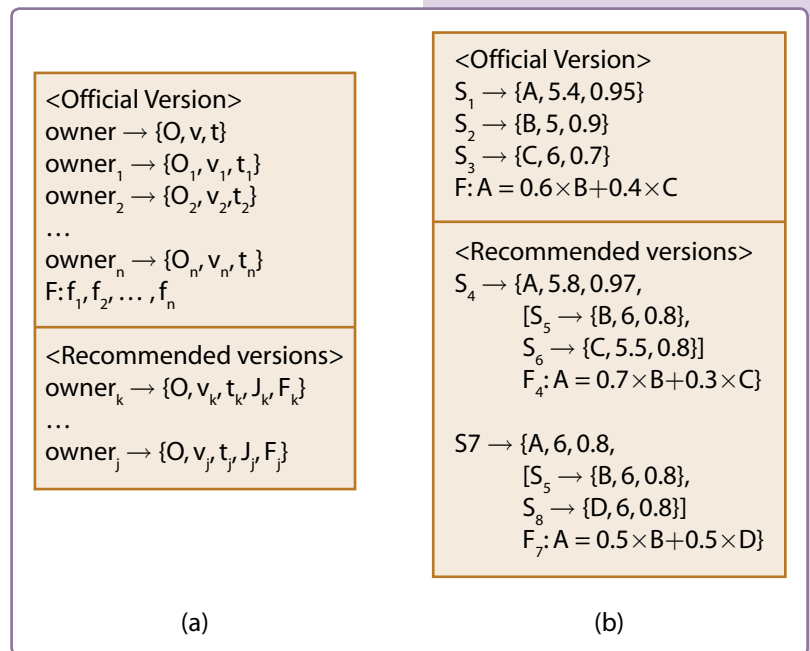
A tuple E has the format $[owner \rightarrow \{object, v, t\}]$, where $owner$ represents the owner of this version, i.e. the subject which produces this version; $object$ represents the target object that this version is about; $v \in V$ and V represents the domain of object value of $object$; t represents the trust value of v viewed by $owner$.

The semantic meaning of an entry represented in format E is that $owner$ proposes that the object value of $object$ is v , and $owner$ has assigned the trust level t to v .

- (2) RV , denoting the recommended versions, represents a set of version of O that are recommended by other subjects in the virtual organisation. Each entry of RV represents one recommended version and it is in format E' as described below.

The format of a tuple E' is $[owner_1 \rightarrow \{object, v', t', J, F'\}]$, where $owner_1$ represents the owner of this recommended version; $v' \in V$ (V has the same meaning as above);

Figure 3. Format of a label (a) and an example (b)



J , shorthand for *justification*, is a set $J = \{J_1, J_2, \dots, J_m\}$, where J_i represents an entry for a component used to form this recommended version and $1 \leq j \leq m$. J_i is in the same format as E ;
 F' contains a set of composing functions that are used to integrate those components to get this recommended version.

An entry in format E' semantically means that a subject owner1 recommends new value v' of *object*. The trust value of v' for $owner_1$ is t' . J represents the set of components, which has been used by owner1 to calculate v' and t' . Each $J_i \in J$ specifies a component selected by $owner_1$ in forming t' in the recommended version as the object value for *object*.

Figure 3 (b) shows an example label format for object A. The official version of A proposes object value of 5.4 for A and trust value of 0.95 of this object value for subject S_1 . S_1 is the owner of this official version of A. S_1 calculates these values by using the composing function $A = 0.6 \times B + 0.4 \times C$. The components of this official version of A are one version of B and one version of C, whose details are explained by the entries $S_2 \rightarrow \{B, 5, 0.9\}$ and $S_3 \rightarrow \{C, 6, 0.7\}$.

The first component indicates that one of the versions of B has an object value of 5 as provided by subject S_2 , which is the owner of that version of B, and the trust value of this provided object value is 0.9. The other component, a version of C, can be interpreted in a similar way.

Two recommended versions of object A are provided by subject S_4 and S_7 respectively. For the first recommended version, the recommended value of A by S_4 is 5.8 and the trust value of this provided object value (5.8) is 0.97 for S_4 . Furthermore, S_4 explains how the object value has been obtained, i.e. the composing function $A = 0.7 \times B + 0.3 \times C$ was used and the components used to form this recommended object value, i.e. a version of B and a version of C, are provided by the entries $S_5 \rightarrow \{B, 6, 0.8\}$ and $S_6 \rightarrow \{B, 5.5, 0.8\}$.

4 An example

This section gives an example to illustrate objects and their corresponding labels in a commercial virtual organisation. There are six subjects relevant to this example and they are: (1) a consulting corporation (C); (2) a government agent (G), e.g., the bureau of statistics; (3) a university survey centre (U); (4) a ranking agent (R); (5) an investment bank (I); and (6) a business firm (F). The firm F is in the process of evaluating the overall investment risk of a large project under consideration. F believes that this overall investment risk is based on two major factors, the project's inherent technical risk and the general investment environment as represented by the ranking level of the current social and economic development. The latter is further determined based on the public confidence towards the well being of the society and economy. The economic well-being is evaluated based on the GDP growth and unemployment rate.

Several subjects in this virtual organisation (such as those mentioned above) provide their opinions about the object value(s) of the relevant objects as required to measure this investment risk for F. Each opinion is regarded as a version for the corresponding object. These logical relationships (as shown in **Figure 4**) among object versions are considered as the basis for the composing functions. For simplicity, in this example, it is assumed that all the subjects agree on the same composing functions to form an object version. But this is not required in general.

Relevant objects in this virtual organisation include (1) public confidence towards the society (PC); (2) GDP Growth Rate (GG); (3) ranking level of the society's well-being (RL); (4) investment risk from economic fluctuation, called economic risk (ER); (5) unemployment rate (UR); (6) the overall investment risk of the project under consideration (IR); and (7) the technical risk of the project under consideration (TR). For computation purpose, the domain of the object value of each object is standardised in range [0, 10], in-

Figure 4. Logical relationships among objects in the example

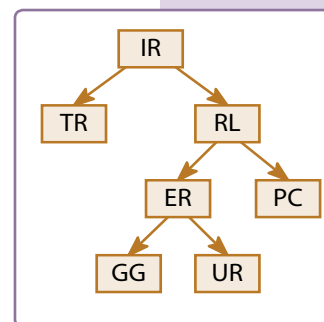
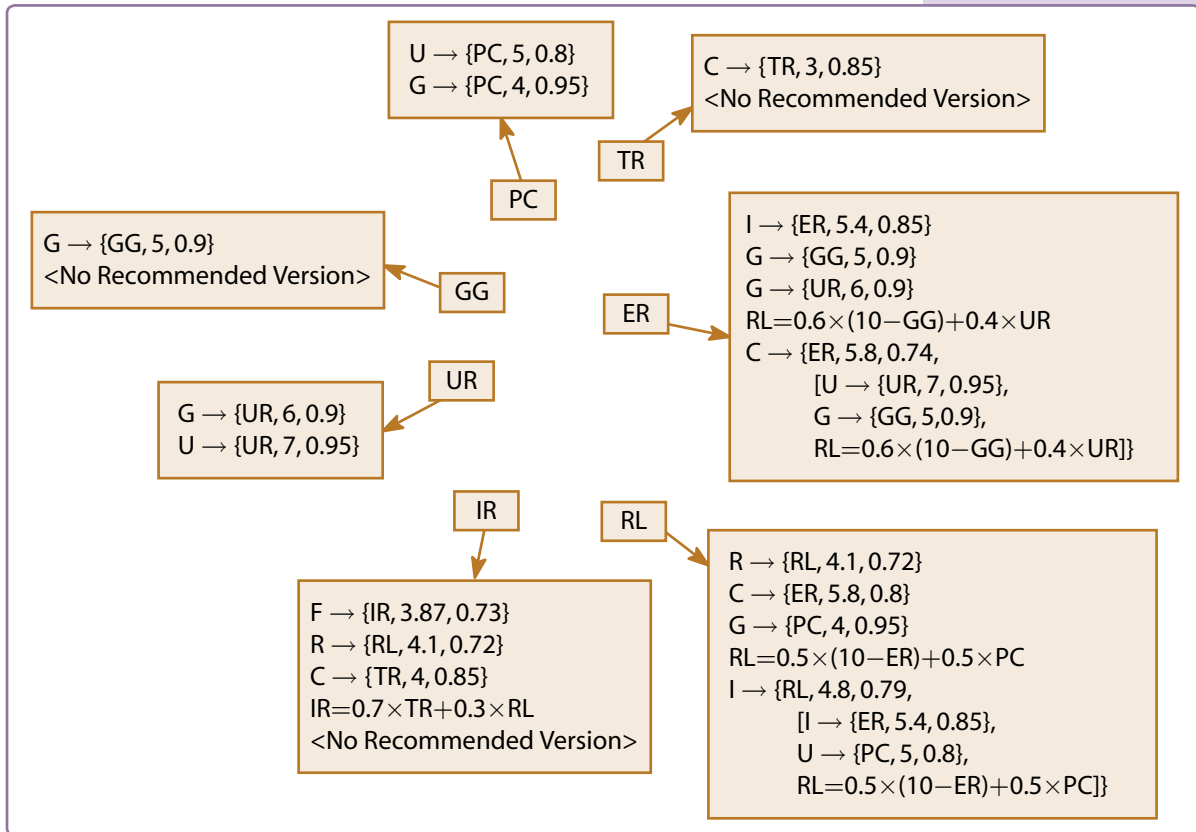


Figure 5. A snapshot of objects and their corresponding labels in the example



clusive of 0 and 10. For instance, the value of 0 for the estimated technical risk of the project represents the minimum risk (or no risk at all) and the value of 10 indicates the maximum risk (or very risky). Any real number between 0 and 10 indicates certain level of risk as viewed by the information provider. The objects and their corresponding labels containing some example object values assumed to be provided by different subjects are shown in Figure 5. Computation of the labelled trust value of each compound object version will be shown in Section 5.2.

5 Trust calculation

This section discusses methods to calculate the trustworthiness of an object version in term of its quality. Subject trust, i.e. the trust value of a subject for another, has been incorporated into the calculation of the trust value of an object version. It is helpful for a user to evaluate the trust value of a given object version if information about the trustworthiness of the version's owner is available. Next, a method for subject trust calculation is introduced.

Subject trust

An effective way to calculate the trust value of a subject for a given subject, where the former is not a neighbour of the latter, is by using a trust network, which describes direct trust relationship between two neighbour subjects. One subject is a neighbour to another if the latter has built and maintained personal trust relationship with the former via series of contacts between them. A subject evaluates the performances of its neighbour after each transaction between them and uses that evaluation to update the trust value for the neighbour, which is maintained locally by the evaluating subject. Those direct trust form a trust network, which is also called a web of trust.

Definition 7. A *trust network* is a weighted directed graph with each vertex representing a subject and each directed edge with weight g from vertices V_1 to V_2 representing that V_1 directly trusts V_2 with trust value of g .

Example of a trust network, G , is given in **Figure 6 (a)**, which corresponds to the example discussed in **Section 4**. It is assumed that all the subjects in the virtual organisation know this trust network.

Although a trust network represents only direct trust relationships between subjects and their neighbours, indirect trust between two subjects that are not neighbours can be calculated based on the principle of trust transitivity. Given a trust network, several methods [16, 17, 18], for example, have been developed to recursively calculate the transitive trust between two non-neighbour subjects. These methods use interactive algorithms to seek converged values as global trusts for all subjects in the system. They hope that after finite number of interactions the global trust values will be available. M. Richardson [17] used path algebra and transitive closure in their calculation of global trust. Each subject needs only to know its personal trust to a given subject, and the aggregated trust of its neighbours towards the same subject. Then a global trust to that target subject can be calculated.

Similar methods were used by other models (such as [8]) to integrate discounted trusts calculated through different paths in a trust network. They differ primarily in the concatenation and aggregation functions. Researchers have used the maximum principle to combine trusts from different paths of a trust network [19]. Under this principle, a user believes anything that is believed by at least one of the users she trusts.

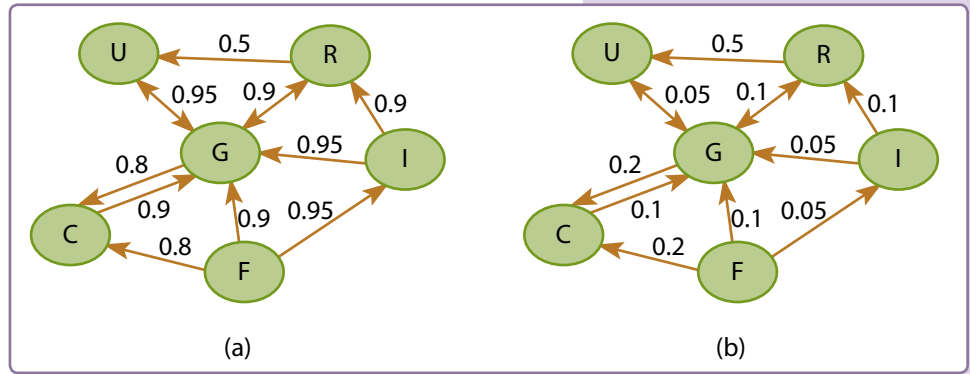
One concern about these proposed methods is their intensive computations before the converged global trust values are obtained. Furthermore, the methods cannot be applied to situations when the iterations don't coverage. A method has been developed in this paper to eliminate this problem. It, as outlined in the following algorithm, is computationally simple and easy to understand.

Algorithm. All-pair shortest path discovery for indirect subject trust computation

1. For every $e \in E(G)$, change the weight of e from t to t' where $t' = 1 - t$
/* This generates a new trust network, G' , such that $V(G) = V(G')$ and $E(G) = E(G')$ but each $e \in E(G')$ has new weights as t' ; */
2. Perform an 'all pair shortest path' algorithm to find the shortest path between every given pair of vertices in G' and generate a path $P = \{S_i, S_{i+1}, \dots, S_j\}$ for a given pair of nodes S_i and S_j .
3. For every pair of subjects (S_i, S_j) , the amount of trust S_i has on S_j is calculated as $T(S_i, S_j) = T(S_i, S_{i+1}) \times T(S_{i+1}, S_{i+2}) \times \dots \times T(S_{j-1}, S_j)$
where edge $(S_k, S_{k+1}) \in E(G)$ and $(S_k, S_{k+1}) \in P$ for $i \leq k \leq j-1$; $T(S_k, S_{k+1})$ is the amount of trust S_k has on S_{k+1} for as indicated in G , i.e. edge value of (S_k, S_{k+1}) and $i \leq k \leq j-1$.

The algorithm, 'all pair shortest path discovery for indirect subject trust computation', which is different from the maximum aggregation principle, identifies a path with minimum sum of distrust for every pair of remote subjects. Then appropriate aggregation of trust along that path can be applied to calculate the indirect trust values between any pair of subjects based on a trust network. Given a trust network G , for every edge $e \in E(G)$ with edge weight t , where t is a real number in the range $[0, 1]$, re-label e with $t' = 1 - t$. So a new graph G' is generated with $V(G) = V(G')$ and $E(G) = E(G')$ but each edge of G' has different value from the corresponding edge of G . Intuitively, t' specifies the degree of direct distrust between two subjects. G' is called a distrust network. **Figure 6 (b)** shows the distrust network corresponding to the given trust network **Figure 6 (a)**. The goal of this algorithm is to find a path between any pair of nodes of G' with minimum sum of t' values along that path. The algorithm uses G' instead

Figure 6. A Trust Network G (a) and the Corresponding Distrust Network G' (b)



of G because finding a path with maximum length between two given nodes would be computationally non-tractable if each edge is labeled with the trust value. It is easy to prove that the time complexity algorithm shown below is polynomial in the worst case, i.e. $O(n^3)$, where n is the number of vertex in the given graph.

By applying the above algorithm, the indirect subject trust values, as shown below, can be obtained based on the distrust network given in **Figure 6 (b)**. Those values are used in **Section 5.2** in calculating trust values of object versions. Direct trust of a subject for its neighbour is given directly in a trust network.

The trust value of U for C is calculated as:

$$T(C, U) = T(C, G) \times T(G, U) = 0.9 \times 0.95 = 0.86 \text{ (the shortest path from } C \text{ to } U \text{ on the distrust network } G' \text{ is } C \rightarrow G \rightarrow U).$$

Similarly, the following indirect subject trust values can be obtained:

$$T(R, C) = T(R, G) \times T(G, C) = 0.9 \times 0.8 = 0.72;$$

$$T(I, U) = T(I, G) \times T(G, U) = 0.95 \times 0.95 = 0.9;$$

$$T(F, R) = T(F, I) \times T(I, R) = 0.95 \times 0.9 = 0.86.$$

Object version trust

Object version trust refers to the degree to which a subject evaluates the trustworthiness of an object version in term of its quality features. According to the component-based approach, calculating the trust value of a compound object version is based on the trustworthiness of its components and the composing functions. The trust value of the object version obtained in this way is called the object version's *primary trust value* for the evaluator.

Definition 8. The *primary trust value* of an object version $V_i^{(O)}$ for subject S , denoted as $T(S, V_i^{(O)})_{primary}$, is the trustworthiness of $V_i^{(O)}$ for S in term of its quality and is calculated based on S 's direct experiences of studying the closely related information about $V_i^{(O)}$, e.g., the trustworthiness of the components of $V_i^{(O)}$ and the appropriation of composing functions used to form $V_i^{(O)}$.

Let elements in the set $\{C_1, C_2, \dots, C_n\}$ denote the components of a compound object version $V_i^{(O)}$ and elements in the set $\{S_1, S_2, \dots, S_n\}$ represent the owners of those components with S_i being the owner of C_i for $0 < i \leq n$. For subject S , $T(S, V_i^{(O)})_{primary}$ can be calculated based on formula (5.1):

$$T(S, V_i^{(O)})_{primary} = \Gamma_{(S, V_i^{(O)})}(F, T(S, C_1), T(S, C_2), \dots, T(S, C_n)) \quad (5.1)$$

where $\Gamma_{(S, V_i^{(O)})}$ represents the trust function used to evaluate $Trust(S, V_i^{(O)})_{primary}$ based on the trust values of the components of $V_i^{(O)}$. $T(S, C_i)$ represents the trust value of component C_i for S , where $1 \leq i \leq n$. Component trust will be discussed in more detail in next section. A trust function can provide answer to the question 'how much should a compound object version be trusted given the trust values of its components and the composing functions used to form that compound object version?' Function $\Gamma_{(S, V_i^{(O)})}$ takes composing functions represented by F and a set of component trust values as input. One reason that different subjects may view the same object version with different levels of trustworthiness is that they may have different trust functions for a given object version and/or they trust the components of the object version to different degrees.

In order to apply formula (5.1) to calculate the trust value of a given object version, the following two questions should be answered: (1) how to calculate the trust value of each component and (2) how to determine the trust function given the composing functions used in forming the object version. The following two sub-sections discuss these issues and their solutions. The third subsection offers some examples of object version trust calculation based on the example provided in the **section 4**.

I. Component Trust Values

As described below, there are two options available to evaluate the trustworthiness of the component versions, each of which is an object version by itself, for a given compound object version.

- (1) Continue to apply formula (5.1) to each component, i.e. evaluate the trustworthiness of each component by studying the trust values of its components. This 'divide and conquer' style process continues until some trivial cases for object trust evaluation are encountered. For instance, some object versions have been tested in practice and have been proved to have higher trust values. Other object versions have been publicly recognised as trustworthy entities, i.e. they have high reputations of being 'good' object versions. As discussed earlier, this approach requires intensive computation. Given the depth, d , of the version dependency tree for the given object version under evaluation, the time complexity of the algorithm in this way is exponential in the worst case, i.e. $O(m^d)$, where m is the branching factor, i.e. the average number of components that each object version has. Another challenge in applying this method is that a trivial case of evaluating object version may not be available in every situation.
- (2) The trust value of each component is obtained through the evaluator's secondary experiences such as the opinion about the component version from the component version's owner and the trustworthiness of the component owner itself for the evaluator. The trustworthiness of the owner of a given component can be obtained by using the 'all pair shortest path discovery for indirect subject trust computation' algorithm, whose time complexity is polynomial in the worst case, i.e. $O(n^3)$. So the time complexity of calculating the trust value of an object version using component-based approach is $m \times O(n^3) = O(n^3)$ where m is the branching factor as explained above and is considered as a constant. This time complexity is irrelevant to d , the depth of the version dependency tree for the object version under evaluation.

The trust value of a component version calculated through the evaluator's secondary experiences is called the *secondary trust value* of the component version. In the following discussion, this secondary trust value is used as the trust value of a component for computational efficiency. The general definition of the secondary trust value of an object version is given below.

Definition 9: The *secondary trust value* of $V_i^{(O)}$ for S , denoted as $T(S, V_i^{(O)})_{secondary}$, is the trustworthiness of $V_i^{(O)}$ obtained through secondary experiences of S , e.g., the information S has on the trustworthiness of $V_i^{(O)}$ from other parties such as the owner of $V_i^{(O)}$.

For an evaluating subject, S , $T(S, V_i^{(O)})_{secondary}$ can be calculated as the mathematical product of the trust level of the object version for its owner, S' , the trust level of S' for S , as well as a context adjusting parameter. The context adjusting parameter reflects the degree of belief of S about the ability of S' evaluation of $V_i^{(O)}$ in term of its quality. The formula to calculate $T(S, V_i^{(O)})_{secondary}$ is given below.

$$T(S, V_i^{(O)})_{secondary} = T(S, S') \times CJ(V_i^{(O)}) \times T(S', V_i^{(O)}) \quad (5.2)$$

where $CJ(V_i^{(O)})$ is a context adjusting parameter.

If the secondary trust values of the components of a compound object version under evaluation are used as the trust values of those components, then the primary trust of the compound object version can be calculated as **Formula (5.3)**.

$$T(S, V_i^{(O)})_{primary} = \Gamma_{(S, V_i^{(O)})} (F, T(S, S_1) \times CJ(C_1) \times T(S_1, C_1), T(S, S_2) \times CJ(C_2) \times T(S_2, C_2), \dots, T(S, S_n) \times CJ(C_n) \times T(S_n, C_n)) \quad (5.3)$$

II. Trust Function

A trust function helps measure the 'composed' trust value of a compound object version after its components have been evaluated and the composing functions used to form the compound object version have been specified. Hence, a trust function depends on the format of the corresponding composing functions. Developing a general format for a trust function is domain

dependent. It is also subjective and varies from user to user. Two common cases are discussed below.

Case 1. For a weighted average composing function $F = (w_1 \times C_1) \theta (w_2 \times C_2) \theta \dots \theta (w_n \times C_n)$, the corresponding trust function is

$$\Gamma_{(S, V(O))} (F, T(S, S_1) \times T(S_1, C_1), T(S, S_2) \times T(S_2, C_2), \dots, T(S, S_n) \times T(S_n, C_n)) = w_1 \times T(S, S_1) \times T(S_1, C_1) + w_2 \times T(S, S_2) \times T(S_2, C_2) + \dots + w_n \times T(S, S_n) \times T(S_n, C_n) \quad (5.4)$$

where w_1, w_2, \dots, w_n are real numbers in the range $[0, 1]$ and they add up to 1; the symbol θ represents binary operators such as addition, subtraction, multiplication, division, etc. Intuitively, if a composing function has the format as a weighted average, then the same parameters in the composing function are used to integrate the trust value of each component in order to calculate the trust value of the compound object version as shown in **Formula (5.4)**.

Case 2. For the composing function $F = c \theta A$, where c is a constant parameter, the corresponding trust function is

$$\Gamma_{(S, V(O))} (F, c, T(S, V_i^{(O)}) = T(S, V_i^{(O)}) \quad (5.5)$$

III. Object Version Trust Calculation

Given the method to compute the trust value of each component of a given compound object version and the heuristic formula for trust function, we now illustrate the calculation of trust values of compound object versions shown in the example provided in **Section 4**. The trust value of **ER** for subject **I**, i.e. the official version of **ER**, is obtained in the following way by applying **formula (5.3)** and **(5.4)** (for calculation simplicity, the context adjusting parameters are assumed to be 1):

$$T(I, ER) = 0.6 * T(I, G) * T(G, GG) + 0.4 * T(I, G) * T(G, UR) = 0.6 * 0.95 * 0.9 + 0.4 * 0.95 * 0.9 = 0.85$$

For subject **C**, the trust value of **ER**, as provided in the recommended version of **ER**, is

$$T(C, ER) = 0.6 \times T(C, G) \times T(G, GG) + 0.4 \times T(C, U) \times T(U, UR) = 0.6 \times 0.9 \times 0.9 + 0.4 \times 0.86 \times 0.9 = 0.8$$

Other object version trust values are calculated as demonstrated below.

$$T(R, RL) = 0.5 \times T(R, C) \times T(C, ER) + 0.5 \times T(R, G) \times T(G, PC) = 0.5 \times 0.72 \times 0.8 + 0.5 \times 0.9 \times 0.95 = 0.72$$

$$T(I, RL) = 0.5 \times T(I, I) \times T(I, ER) + 0.5 \times T(I, U) \times T(U, PC) = 0.5 \times 1 \times 0.85 + 0.5 \times 0.9 \times 0.8 = 0.79$$

$$T(F, IR) = 0.3 \times T(F, R) \times T(R, RL) + 0.7 \times T(F, C) \times T(C, TR) = 0.3 \times 0.86 \times 0.72 + 0.7 \times 0.9 \times 0.85 = 0.73$$

Referring back to the example introduced in **Section 4**, the firm **F** determines that the risk of investing in the project under consideration is 3.87 in the scale of $[0, 10]$ given the estimation of investing environment and technical risk of the project.

6 Heuristic Methods for Object Trust Calculation

In this section, two heuristic approaches are described as complementary methods to the object trust evaluation as discussed in **Section 5**. They both use component-based approach.

Object trust combination

In subject trust management, if no single subject can have the required trust level for a user, then cooperation of multiple subjects as a whole may provide a combined trust value, which satisfies the requirement of the user. For example, the owner of a private bank requires that only highly trusted personnel authorise critical transactions and endorse relevant documents. If no single employee can have the desired trustworthiness in the eye of the bank owner, in order to authorise such a crucial transaction, the bank owner may consider allowing several employees to co-sign the transaction. This form of trust aggregation is called subject trust combination and the trust values obtained through this process is regarded as higher than the trust value of any single subject. This method is based on the assumption that subjects working as a team are more trustworthy than any single one of them working alone and the chance of cooperatively cheating is relatively smaller.

Similar line of thought can be applied to object trust management. For a given object, different subjects have different opinions about its object value and it is likely that none of the single opinion satisfies a user’s requirement for information quality. However, if several opinions about the same object present close results and these results provided by the subjects are obtained by using different methods, then the users have higher confidence in the results. Hence, for a user, the trust value of those similar results is higher than any one of the individual opinions considered separately. Intuitively, those object versions with similar results for the given object provide ‘multiple proof’ of their correctness. They are not the same by coincidence.

The calculation of a combined object trust requires three steps. The first step is to identify a set of object versions for the given object so that those identified object versions provide close object values for the given object. Object versions can be in various forms including documents, files, etc. Techniques developed for pattern matching or documents comparison can be employed in identifying similarity between two given documents or files. Oliveria and Zaiane [20] offered a background tutorial for comparing the similarity between two objects, each of which has values for a set of quantitative attributes. Measuring the similarity between two text documents can be achieved by employing certain clustering techniques, which essentially group similar documents into the same set. The methods used for text clustering include decision trees, statistical analysis, neural nets, inductive logic programming, and rule-based systems.

After producing a set of object versions with similar results for a given object, the second step in calculating combined object trust is to identify those object versions, which have been integrated using significantly different components, composing functions, or both. This essentially indicates that similar results have been obtained by different subjects using significantly different approaches. We have developed an algorithm to calculate the dissimilarity between two component structures and analysed the time complexity of the algorithm [15]. Due to page limitation, the algorithm is not presented in this paper.

The third and the final step is to calculate the combined trust value based on the trust value of each individual object version as identified in step 2. This object trust combination can be viewed as stochastic processes. If the trust values of the two versions of a given object are t_1 and t_2 respectively, where t_1 and t_2 are real numbers in the range $[0, 1]$, then the combined trust value t is

$$t = 1 - (1-t_1) \times (1-t_2) \quad (6.1)$$

Component pattern discovery and version reconstruction

In general, an object has multiple versions as supplied by different subjects in a virtual organisation. For some object versions, while certain components are highly trusted, others are questionable. If there is a way to discover the component patterns based on different versions of an object and identify some components that are frequently used together to form an object version, then a new version with a higher trust value can be constructed using these identified components (along with trustworthy composing functions). This reconstructed version is regarded to have higher trustworthiness than other versions of the same object since it is based on highly trusted components of all available versions. This new version provides more confidence to users regarding its quality.

REFERENCES

1. Wasson G. and Humphrey M. (2003), ‘Towards explicit policy management for virtual organisation’, *Proceedings of IEEE 4th International Workshop on Policies for Distributed Systems and Networks*, Lake Como, Italy.
2. Patil R. and Shyamasundar R.K. (2003), ‘Role-based authorization and delegation system’, *Proceedings of International Conference on Computational & Experimental Engineering & Sciences*, Corfu, Greece.
3. Mowshowitz A. (2003), ‘Virtual organisation: towards a theory of societal transformation stimulated by information technology’, *ACM-Ubiqity*, vol. 4, issue 11.
4. Brewer D.F. and Nash J. (1989), ‘The Chinese wall security policy’, *Proceedings of the IEEE Symposium on Security and Privacy*, Oakland, California, USA.
5. Denning D.E. (1976), ‘A lattice model of secure information flow’, *Communications of the ACM*, 19(5):236–243.
6. Bell D. and LaPadula L. (1997), ‘The Bell-LaPadula model’, *Journal of Computer Security*, 5:303–339.
7. Berson T.A. and Lunt T.F. (1987), ‘Multilevel security for knowledge-based systems’, *Proceedings of the 1987 IEEE Symposium on Privacy and Security*, pp. 235–242, Oakland, California, USA.
8. Blaze M., Feigenbaum J. and Lacy J. (1996), ‘Decentralized trust management’, *Proceedings of 17th Symposium on Security and Privacy*, Oakland, USA.
9. Li N. and Mitchell J. and Winsborough W. (2002), ‘Design of a role-based trust-management framework’, *IEEE Symposium on Security and Privacy*, pp. 114–131.
10. Abudual-Rahman A. and Hales S. (2000), ‘Supporting trust in virtual communities’, *Proceedings of 33rd Hawaii International Conference on System Science*, Maui, USA.
11. Azzedin F. and Maheswaran M. (2004), ‘Integrating trust into grid resource management systems’, *Proceedings of 1st IEEE International Workshop on Security and Grid Computing*, Beijing, China.
12. Dimmock N., Belokoztolszki A. and Eysers D. (2004), ‘Using trust and risk in role-based access control policies’, *Proceedings of 9th ACM Symposium on Access Control Models and Technologies*, New York, USA.
13. Josang A. (2002), ‘The consensus operator for combining belief’, *Artificial Intelligence Journal*, vol. 141/1-2, pp. 157–170.

The authors in [15] have developed a model to allow a user to discover association rules for a set of components, which are most frequently used together with a composing function. Then the user will be able to 'rebuild' a new version for the object. Hence, the object values as provided by this new version should have higher accuracy. In addition, the user's personal experiences can be incorporated in the reconstruction process to make the new version more trustworthy.

7

Conclusions

Object trust management has yet to receive much attention from the research community. In an open or semi-open system such as a virtual organisation, it is vital to have object trust management in order to ensure the quality of external information received. This is true especially when an object has multiple versions and each version presents a different view of a subject towards the object. Component-based approach enables a user to study how a given object version has been formed and how those components are trusted. In order to present component information in a standard manner and that information can be understood by all the subjects, our model uses object labelling mechanism to express available versions for a given object and component information for each version. The presented model also allows users to compare the official and recommended versions for a given object. The users can then select the versions that they believe are most trustworthy.

The paper presents formula to allow an evaluator to use its first-hand information in evaluating the trustworthiness of a given object version. The trust value calculated in this way is called the primary trust value of the object version. This computation can be conducted recursively by applying the evaluation process to each component. But this method may not be computationally efficient. Alternatively, a user may use the secondary trust value as the trustworthiness of each component in calculating the primary trust value of the corresponding compound object version. The second method is much more efficient since no recursion is required. In any case, trust function is important to measure the trust value of a compound object version given the trust values of its components as well as the composing functions. Furthermore, based on the component-based approach, two heuristic methods have been presented that can be used to estimate the trustworthiness of an object version. They are not to replace the general object trust method but serve as complementary approaches to the computations of object version trust. Then a user has an option in considering different ways in trust evaluation for an object version.

ACKNOWLEDGMENT

This work was supported in part by US AFOSR under grant FA9550-04-1-0429.

We would like to thank Dr. Robert Herklotz for his support of this research and the anonymous reviewers for their suggestions and comments for improving the quality of this paper.

14. Myers A.C. and Liskov B. (1997), 'A decentralized model for information flow control', *Proceedings of the 16th ACM Symposium on Operating Systems Principles*, Saint-Malo, France.
15. Zuo Y. (2005), *Towards a Framework of Object Trust Management for Information Assurance within a Virtual Organisation*, Ph.D. dissertation.
16. Ray I. and Chakraborty S. (2004), 'A vector model of trust for developing trustworthiness systems', *Proceedings of the 9th European Symposium on Research in Computer Security*, Sophia Antipolis, France.
17. Richardson M., Agrawal R. and Domingos P. (2003), 'Trust management for the Semantic Web', *Proceedings of 2nd International Semantic Web Conference*, Sanibel Island, USA.
18. Josang A. (1999), 'An algebra for assessing trust in certification chains', *Proceedings of the Internet Society 1999 Network and Distributed System Security Symposium*, San Diego, USA.
19. Bellman R. and Giertz M. (1973), 'On the analytic formalism of the theory of fuzzy sets', *Information Sciences*, vol. 5, pp. 149–159.
20. Oliveira S. and Zaiane O. R. (2004), 'Privacy-preserving clustering by object similarity-based representation and dimensionality reduction transformation', *Proceedings of Workshop on Privacy and Security Aspects of Data Mining in conjunction with the Fourth IEEE International Conference on Data Mining*, Brighton, UK.