General Methodology for Analysis and Modeling of Trust Relationships in Distributed Computing

Weiliang Zhao
School of Computing and Mathematics, University of Western Sydney, Australia
Email: wzhao@cit.uws.edu.au

Vijay Varadharajan
Department of Computing, Macquarie University, Australia
Email: vijay@ics.mq.edu.au

George Bryan
School of Computing and Mathematics, University of Western Sydney, Australia
Email: wzhao@cit.uws.edu.au

***Abstract*—In this paper, we discuss a general methodology for analysis and modeling of trust relationships in distributed computing. We discuss the classification of trust relationships, categorize trust relationships into two layers and provide a hierarchy of trust relationships based on a formal definition of trust relationship. We provide guidelines for the analysis and modeling of trust relationships. We review operations on trust relationships and relative types of trust relationships in our previous work. We provide a set of definitions for the properties of direction and symmetry of trust relationships. In order to analyze and model the scope and diversity of trust relationship, we define trust scope label. We provide some example scenarios to illustrate the proposed definitions about properties of trust relationship. All the definitions about the properties of trust relationships are elements of the taxonomy framework of trust relationships. We discuss the lifecycle of trust relationships that includes the analysis and modeling of trust relationships, trust relationships at runtime, and change management of trust relationships. We propose a trust management architecture at high level to place the analysis and modeling of trust relationships under the background of trust management.**

***Index Terms*—trust relationship, properties of trust relationship, life cycle of trust relationship, trust management, distributed information system**

I. Introduction

Many researchers have realized that trust has immense significance in distributed computing. There are a lot of researches to study or model trust across different areas. The notion of trust has been around for many decades in different disciplines in different disguises. In security, the concept of "trusted systems" has been around explicitly at least since late 1970s [1, 2]. Trust is used in the concept of convincing observers that a system is correct and secure. Following this concept, Trusted Computing Base and Trusted Platform become hot topics for both academic and industry people. Trust is a complex subject relating to belief or perception of the trusted entity [3]. Some researchers have tried to formalize trust as a computational concept such as S. Marsh [4] and A. Jøsang et al [5, 6]. There are many services and applications that must accommodate appropriate notions of trust and related elements of trust such as community reputation and security credentials. Reputation-based systems such as XREP [7], NICE [8] and P-Grid [9] provide facility to compute the reputation of an involved entity by aggregating the perception of other entities in the system. Some reputation systems like TrustNet[10] and NodeRanking[11] utilize existing social relationships to compute reputations based on various parameters. There are many trust management systems based on credentials. Public key certificates X.509 and PGP use credentials to deal with trust management problem. As a further step, M. Blaze et al firstly identified trust management as a distinct and important component of security in distributed environments and proposed PolicyMaker [12]. After that, several automated trust management systems have been proposed and implemented including PolicyMaker[12], KeyNote[13] and REFEREE [14]. All the above systems use credentials as evidence of required trust. Normally, there are credential verification and secure application policies to restrict access to resources and services [15]. Several automated trust negotiation systems have been proposed [16-18]. Automated trust negotiation is the approach to establishing trust between strangers through iterative disclosure of digital credentials.

Trust plays an important role in distributed information systems. The properties of trust and how to define/model trust relationships are important concerns in the analysis

Based on "Analysis and Modeling of Trust in Distributed Information Systems", by Weiliang Zhao, Vijay Varadharajan, and George Bryan which appeared in the Proceedings of the First International Conference on Information Systems Security 2005, Kolkata, India, December 2005. © 2005 Springer.

and design of distributed information systems. Actually, there is no consensus in the literature on what trust is. Though trust has been a foundational stone for security, it has been a difficult concept to define clearly. In order to reflect many of the commonly used notions of trust, we outlined a formal definition of trust relationship in our previous work [19]. In order to have a clear understanding of trust relationships, it is desirable to have taxonomy framework for describing and categorizing trust relationships. The taxonomy framework will reflect the different forms of trust relationships based on their specific characteristics. Based on the formal definition, we build up a taxonomy framework where a range of useful trust relationships can be expressed and compared. The formal definition of trust relationship provides corner stone and starting point to analyze both commonly used and some unique trust notions that arise in distributed computing. Our main objective of this research is to develop a sound understanding of trust and create a powerful set of tools to analyze and model trust relationships in distributed information systems. Our target is to achieve a general methodology for analysis and modeling of trust relationships in distributed information systems.

Based on the results of our previous research [19], this paper discusses the general methodology for analysis and modeling of trust relationships in distributed computing. After we review the definition of trust relationship and discuss the classification of trust relationships, we provide guidelines for the analysis and modeling of trust relationships in distributed information systems. For the properties of trust relationships, we review some results about operations on trust relationships and relative types of trust relationships in our previous work [19] and provide a set of definitions to define the properties of direction and symmetry of trust relationships and the properties of scope and diversity of trust relationship. We discuss the lifecycle of trust relationships that includes the analysis and modeling of trust relationships, trust relationships at runtime, and the change management of trust relationships. We propose trust management architecture based on our current understanding of trust relationships in distributed information systems. In this paper, we only provide some high level results of the trust management architecture.

The remainder of the paper is organized as follows. In Section II, we provide the formal definition of trust relationship. In Section III, we discuss the classification of trust relationships. We categorize trust relationships into two layers and provide a trust relationship hierarchy based on our formal definition of trust relationship. In Section IV, we outline some guidelines for the analysis and modeling of trust relationships in distributed information systems. In Section V, we discuss the properties of trust relationships. Operations on trust relationships and relative types of trust relationships are defined. We define the properties about the direction and symmetry of trust relationship and the properties about the scope and diversity of trust relationship. Some scenario examples are provided in this section. In Section VI, we discuss the lifecycle of trust relationships. In Section VII, we provide a trust management architecture. Finally Section VIII provides some concluding remarks.

II. Definition of Trust Relationship

Trust is a very general concept which can be used in different context. In information technology, it is desirable to have a formal definition for trust that can be used for the purpose of computing. We have given a formal definition of trust relationship with a strict mathematical structure in our previous work [19]. This definition of trust relationship has a broad expressive power and it is the cornerstone of our trust taxonomy framework. All trust notions discussed in this paper are based on this definition. The definition of trust relationship is expressed as:

Definition 1: *A trust relationship is a four-tuple T =< R, E, C, P > where:*

- *R is the set of trusters. It contains all the involved trusters. It is a non empty set.*
- *E is the set of trustees. It contains all the involved trustees. It is a non-empty set.*
- *C is the set of conditions. It contains all conditions (requirements) for the current trust relationship. Normally, a trust relationship has some specified conditions. If there is no condition, the condition set is empty.*
- *P is the set of properties. The property set describes the actions or attributes of the trustees. It is a non-empty set. The property set can be divided into two sub sets:*
  - *Action set: the set of actions that trusters trust that trustees will and can perform.*
  - *Attribute set: the set of attributes that trusters trust that trustees have.*

The formal definition of trust relationship can reflect the commonly used notions of trust and provides a taxonomy framework. When trust relationships are used, the full syntax (four-tuple < R, E, C, P > must be followed. Trust relationship T means that under the condition set C, truster set R trust that trustee set E have the properties in set P. The definition of trust relationship provides a starting point for capturing different forms of commonly understood notions of trust. The above strict definition of the trust relationship is the basis for all properties of trust relationships and it is the starting point for the general methodology for analysis and modeling of trust in distributed information systems.

III. Classification of Trust Relationships

Some researchers have tried to identify different forms of trust relationships. T. Grandison et al [20] have given a bottom-up classification and used the terms as resources access trust, service provision trust, certification trust, delegation trust and infrastructure trust. From the view point of establishment or evaluation of trust relationships, all the above trust types must build on a more basic trust

relationship that is the authentication trust or identity trust. We will categorize trust relationships into two layers. Authentication trust is on layer one and other types are on layer two.

Authentication has continuously been an important topic in information security community. There are many popular authentication schemes such as X.509 and PGP. Authentication trust belongs to a separate layer and all other trust types belong to another layer above the authentication trust. This is illustrated in Figure 1.

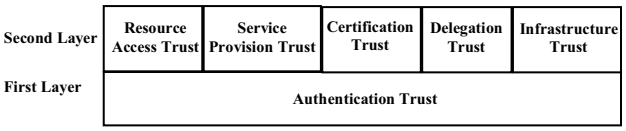

Fig. 1. Trust Layers

Note that trust types of layer two may not be necessarily specified in terms of an identity. Anonymous authorization belongs to access trust and it is an example that there is no specified identity. Anonymous authorization can be implemented using certificates with capabilities. The real identity of the involved trustee will not be revealed. For example, a customer has a certificate for accessing some resources on the Internet. The customer's behaviors of accessing the resources can be recorded. If it is desirable that the customer cannot be identified, the related access trust is a kind of anonymous access trust. Particularly for the resource access trust and service provision trust, the anonymous authentication is desirable in some cases. In such a situation, the layer of authentication still needs to provide a mechanism to deal with the same entity as the trustee in the whole scope of the trust process. Normally, there is a temporary and dynamic identification which will be uniquely connected with the involved trustee in the scope of the trust process.

At layer two, trust relationships can be classified in different ways. In the following, we will give another kind of classification which is different from the bottom-up classification of trust. Based on the strict definition of trust relationship, trust relationships at layer two can be classified according to the nature of the trustees in trust relationship < R, E, C, P >. If E is an infrastructure, the trust relationship belongs to infrastructure trust. If E is not an infrastructure, the trust relationship belongs to non-infrastructure trust. Non-infrastructure trust relationships can be classified based on the ownership of the property set. If the trusters have the ownership of the property set, the trust relationship belongs to access trust. If the trustees have the ownership of the property set, the trust relationship belongs to provision trust. If some properties are owned by trustees and some other properties are owned by trusters, then the trust relationship belongs to mixture (A&P) trust. The hierarchy of trust relationships at layer two is illustrated in Figure 2. In such a classification, delegation trust and certification trust are not independent types. The delegation trust is a special form of provision trust, trustees are the providers of delegated decisions on behalves of trusters. A certification trust can be any subtype of non-infrastructure trust based on the nature of its property set.

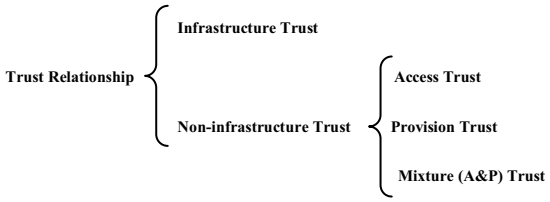

Fig. 2. Trust Relationship Hierarchy

IV. Guidelines for Analysis and Modeling of Trust Relationships in Distributed Information Systems

Trust relationships between possible entities play crucial roles in distributed computing. Analysis and modeling of trust relationships requires identification of sometimes subtle trust assumptions and normally it is a challenging task. The analysis and modeling of trust relationship must be integrated with other requirements of the whole distributed information systems. With the life cycle of the development of a distributed information system, the modeling, implementing and maintaining of trust relationships is an incremental, iterative process. The life cycle of trust relationships share the following basic steps although the details and the ordering of the steps may be changed in some cases.

1. Extract trust requirements in system.
2. Identify possible trust relationships from trust requirements.
3. Choose the whole set of trust relationships from possible trust relationships.
4. Refine the whole set of trust relationships.
5. Implement the whole set of trust relationships.
6. Maintain all the trust relationships in system.

Trust requirements and trust assumptions belong to the business requirements of an information system. The analysis of business requirements is not the main concern of this paper. Any way, our strict definition of trust relationship and the properties of trust relationship discussed and defined in the following section are helpful to understand the possible trust issues and requirements.

How to define and maintain a good set of trust relationships is one of the most important tasks of the trust management in system analysis, system design, implementation and maintenance. The follows are some high level guidelines for the modeling and design of quality trust relationships in distributed information systems.

1. **Completeness:** The completeness means that all trust relationships defined in the system can capture all of trust requirements in the information system.
2. **Sufficiency:** The trust relationships defined in an information system provide suitable level of abstraction to permit meaningful and efficient usage for all issues related with trust in the whole

information system. If possible, a trust relationship had better have more coverage and the total number of trust relationships in the system can be reduced.

3. **High Cohesion:** A trust relationship has a narrow mission. It had better be meaningful and easy to be understood. The trust relationship itself has some inherent maintainability. When the high cohesion and sufficiency of more coverage conflict with each other, there is trade-off between them.
4. **Primitiveness:** In an ideal system, all the trust relationships defined and used in the system are primitive trust relationships (primitive trust relationship will be defined in Section V). There is no information redundancy.

The analysis and modeling of the whole set of trust relationships in a distributed information system is one part of the analysis and design of the whole system and they are dependent on or strongly coupled with the design of other parts of the system. Normally, the analysis and modeling of whole set of trust relationships is quite complicated. The process has an incremental and iterative life cycle. The initial set of trust relationships will be smoothed and refined in multiple life cycles in the terms of completeness, sufficiency, high cohesion and primitiveness. However, these guidelines are at a very high level and they are suitable for any task of analysis and design in information systems. Definitely, they have not touched the specific characteristics of trust in distributed information systems. In order to better understand the specific trust issues and situations and provide suitable terms and tools to analyze and model trust relationships, we need to study and define the properties of trust relationships from different angles.

The goal of the general methodology for analysis and modeling of trust relationships in distributed computing is to help in the design, implementation and maintenance of a distributed information system by highlighting the trust issues inherent to the system. The general methodology will provide not only guidelines but also terms and tools for the analysis and modeling trust relationships in distributed information systems. To achieve the above target, we need to study the properties of trust relationships from different angles. We will provide a taxonomy framework of trust which includes important properties of trust relationships in distributed information systems. These properties include classification of trust, the relations of trust relationships, direction and symmetry of trust relationship, scope and diversity of trust relationship and life cycle of trust relationship. We will provide a set of definitions and operations for the properties of trust relationships. These definitions and operations can enable users to better understand the specific trust issues and situations. From the view point of system analysis and design, they can provide suitable terms and helpful tools to enable the analysis, modeling, implementation and maintenance of trust relationships in distributed information systems.

V. Properties of Trust Relationships

In this section, we will discuss the properties of trust relationships in distributed information systems. We will provide a set of operations and definitions for the properties of trust relationships. These operations and definitions can be used as enabling tools in the analysis and modeling of trust relationships in distributed information systems. We will provide scenario examples to show readers how to understand and use the proposed definitions about the properties of trust relationships in the real world.

A. Operations on Trust Relationships

In this sub section, we provide a set of operations on trust relationships. From the nature of trust relationship and its mathematical structure, some new trust relationships can be derived based on the existing trust relationships. The operations of using two existing trust relationships to generate a new trust relationship under specific constraints and operations of decomposing one existing trust relationship into two new trust relationships under specific constraints are defined as follows:

Operation 1: *Let $T_1 = (R_1, E_1, C_1, P_1)$ and $T_2 = (R_2, E_2, C_2, P_2)$. There is a set $T = (R_1 \cap R_2, E_1 \cap E_2, C_1 \cup C_2, P_1 \cup P_2)$. If $R_1 \cap R_2 = \Phi$ or $E_1 \cap E_2 = \Phi$, $T = \Phi$.*

If $R_1 = R_2$ and $E_1 = E_2$, the operation becomes:

Operation 1A: *Let $T_1 = (R, E, C_1, P_1)$ and $T_2 = (R, E, C_2, P_2)$. There is a set $T = (R, E, C_1 \cup C_2, P_1 \cup P_2)$.*

If $R_1 = R_2$, $E_1 = E_2$ and $C_1 = C_2$, the operation becomes:

Operation 1B: *Let $T_1 = (R, E, C, P_1)$ and $T_2 = (R, E, C, P_2)$. There is a set $T = (R, E, C, P_1 \cup P_2)$.*

Operation 2: *Let $T_1 = (R_1, E_1, C, P)$ and $T_2 = (R_2, E_2, C, P)$. There is a set $T = (R_1 \cup R_2, E_1 \cap E_2, C, P)$.*

If $E_1 = E_2$, the operation becomes:

Operation 2A: *Let $T_1 = (R_1, E, C, P)$ and $T_2 = (R_2, E, C, P)$. There is a set $T = (R_1 \cup R_2, E, C, P)$.*

Operation 3: *Let $T_1 = (R_1, E_1, C, P)$ and $T_2 = (R_2, E_2, C, P)$. There is a set $T = (R_1 \cap R_2, E_1 \cup E_2, C, P)$.*

If $R_1 = R_2$, the operation becomes:

Operation 3A: *Let $T_1 = (R, E_1, C, P)$ and $T_2 = (R, E_2, C, P)$. There is a set $T = (R, E_1 \cup E_2, C, P)$.*

Operation 4: *Let $T = < R, E, C, P >$. If there are R_1, R_2 and $R = R_1 \cup R_2$, there are trust relationships $T_1 = <R_1, E, C, P>$ and $T_2 = <R_2, E, C, P>$.*

Operation 5: *Let $T = <R, E, C, P>$. If there are E_1, E_2 and $E = E_1 \cup E_2$, there are trust relationships $T_1 = <R, E_1, C, P>$ and $T_2 = <R, E_2, C, P>$.*

Operation 6: *Let $T = <R, E, C, P>$. If there are P_1, P_2 and $P = P_1 \cup P_2$, there are trust relationships $T_1 = <R, E, C, P_1>$ and $T_2 = <R, E, C, P_2>$.*

This operation has the following special case:

Operation 6A: *Let $T = <R, E, C, P>$. If there are P_1, P_2, C_1, C_2 and $P = P_1 \cup P_2$, $C = C_1 \cup C_2$, If C_1 is the condition set for P_1 and C_2 is the condition set for P_2, there are trust relationships $T_1 = <R, E, C_1, P_1>$ and $T_2 = <R, E, C_2, P_2>$.*

The above operations can be used to generate new trust relationships from the existing trust relationships under some specific constrains. The **Operation 1** deals with any two trust relationships and a new trust relationship is possibly generated (if the result is not Φ). The **Operation 1A**, **1B**, **2A** and **3A** deal with how to use two trust relationships to generate one trust relationship under some specific constraints. **Operation 4**, **5**, **6** and **6A** deal with how to decompose one trust relationship into two trust relationships under some specific constraints. **Operation 1A** and **Operation 6A** are inverse operations. **Operation 1B** and **Operation 6** are inverse operations. **Operation 2A** and **Operation 4** are inverse operations. **Operation 3A** and **Operation 5** are inverse operations.

B. Relative Types of Trust Relationships

In this sub section, we will discuss the relative types of trust relationships. We will define the equivalent, primitive, derived, direct redundant and alternate trust relationships. We will classify the direct redundant trust relationships into different types.

Definition 2: *Let $T_1 =< R_1, E_1, C_1, P_1 >$ and $T_2 =< R_2, E_2, C_2, P_2 >$. If and only if $R_1 = R_2$ and $E_1 = E_2$ and $C_1 = C_2$ and $P_1 = P_2$, then T_1 and T_2 are equivalent, in symbols:*

$$T_1 = T_2 \Leftrightarrow R_1=R_2 \text{ and } E_1=E_2 \text{ and } C_1=C_2 \text{ and } P_1=P_2$$

Definition 3: *If a trust relationship cannot be derived from other existing trust relationships, the trust relationship is a primitive trust relationship.*

Definition 4: *If a trust relationship can be derived from other existing trust relationships, the trust relationship is a derived trust relationship*.

Note: when a set of trust relationships are defined in an information system, a derived trust relationship is always related to one or more other trust relationships in the information system. For an independent trust relationship, it is meaningless to judge it as a derived trust relationship or not.

Definition 5: *Let $T=<R, E, C, P>$. If there is trust relationship $T' = <R', E', C', P'>$ and $T \neq T'$, $R \subseteq R'$, $E \subseteq E'$, $C \supseteq C'$, $P \subseteq P'$. T is a direct redundant trust relationship.*

We now discuss several special cases of direct redundant trust relationships based on the single tuple of trust relationship. We believe that these special cases play important roles in the analysis and design of trust relationships.

TYPE 1: DRLR (Direct Redundant of Less Trusters) *Let $T = <R, E, C, P>$. If and only if there is a trust relationship $T' = < R', E, C, P>$ and $R' \supset R$, T is a DLR-redundant trust relationship.*

Trust relationship T is DRLR-redundant trust relationship means that there is another trust relationship with super set of trusters and all other tuples are same as peers in T.

TYPE 2: DRLE (Direct Redundant of Less Trustees) *Let $T=<R, E, C, P>$. If and only if there is a trust relationship $T'=<R, E', C, P>\$$ and $E' \supset E$, T is a DLE-redundant trust relationship.*

Trust relationship T is DRLE-redundant trust relationship means that there is another trust relationship with super set of trustees and all other tuples are same as peers in T.

TYPE3: DRMC (Direct Redundant of More Conditions) *Let $T= <R, E, C, P>$. If and only if there is an alternate trust relationship $T' = <R, E, C', P>$ and $C' \subset C$, T is a DRMC-redundant trust relationship.*

Trust relationship T is DRMC-redundant trust relationship means that there is another trust relationship with sub set of conditions and all other tuples are same as peers in T.

TYPE 4: DRLP (Direct Redundant of Less Properties) *Let $T = <R, E, C, P>$. If and only if there is a trust relationship $T' = <R, E, C, P'>$ and $P' \supset P$, T is a DRLP-redundant trust relationship.*

T is DRLP trust relationship means that there is another trust relationship with super set of properties and all other tuples are same as peers in T.

Definition 6: *Let $T = <R, E, C, P>$, $T' = <R, E, C', P>$ and $C \neq C'$. T and T' are alternate trust relationships of each other.*

An alternate trust relationship means that there is an alternate condition set for the same truster set, trustee set and property set. Perhaps, there are multiple alternate trust relationships. In distributed information systems, multiple mechanisms and multiple choices are necessary in many situations and it is the main reason why we define and discuss alternate trust relationships here.

Scenario Example I: Consider an online e-commerce service called FlightServ, which can provide flight booking and travel deals. FlightServ is designed using web services. FlightServ connects with customers, airlines, hotels and credit card services (some of these may also be web services). The whole system could be very complicated, but in this example, we only consider some basic trust relationships in the system. In the system, customers are classified into normal flyers and frequent flyers. Originally, some trust relationships are modeled as follows:

TS1-1: *Airlines trust normal flyers can make their airline bookings, if they have address details & confirmed credit card information.*
TS1-2: *Airlines trust frequent flyers with no condition that frequent flyers can make their airline bookings.*
TS1-3: *Hotels trust normal flyers can make their hotels booking, if they have address details & confirmed credit card information.*
TS1-4: *Hotels trust frequent flyers can make their hotels booking, if they have address details & confirmed credit card information.*
TS1-5: *Credit card services are trusted by all possible entities without any condition that the credit card services will give the correct evaluation of credit card information.*
TS1-6: *Credit card services are trusted by all possible entities without any condition that the credit card services will keep the privacy of credit card information.*

For the above trust relationships in the system, based on definitions and operations in section 3, we have the following analysis:

- All above trust relationships are primitive.
- Using the **Operation 3A**, trust relationships **TS1-3** and **TS1-4** can be merged to a new trust relationship **TS1- (3)(4)**: "Hotels trust customers if they have address details & confirmed credit card information that customers can make their hotels booking". If **TS1- (3)(4)** has been defined in the system, **TS1-3** and **TS1-4** becomes DRLE trust relationships and will be removed out of the system.
- Using the **Operation 1B**, trust relationships **TS1-5** and **TS1-6** can be merged to a new trust relationship **TS1-(5)(6)**: "Credit card services are trusted by all possible entities without any condition that the credit card services will give the correct evaluation of credit card information & the credit card services will keep the privacy of credit card information". If **TS1-(5)(6)** has been defined in the system, **TS1-5** and **TS1-6** becomes DRLP trust relationships and will be removed out of the system.

We hope that the above scenario example can provide a general picture for using the operations on trust relationships and the relative types of trust relationships to analyze and modeling trust relationships in distributed information systems.

In the following sub sections, we will provide more definitions about the properties of trust relationship that include the direction and symmetry of trust relationship and the scope and diversity of trust relationship.

C. Direction and Symmetry of Trust Relationship

In this sub section, we will provide a set of definitions for the properties of trust direction and trust symmetry. The properties of trust direction and trust symmetry play an important role in the analysis and modeling of trust relation ships in distributed information systems. These definitions provide general descriptions about the properties of trust direction and trust symmetry. A scenario example is provided to illustrate these definitions and their usage. We hope that these definitions can cover most situations in the real world and can be used as standard scenarios for analyzing and modeling trust relationships about properties of direction and symmetry. In real systems, one or multiple kinds of trust direction and trust symmetry can be chosen based on the specified requirements of the information systems.

The properties of trust direction and symmetry are related to each other and they should be cooperatively used to analyze and model the properties of direction and symmetry of trust relationships in distributed information systems. For the properties of trust direction, one-way trust relationship, two-way trust relationship and reflexive trust relationship are defined. For the properties of trust symmetry, symmetric trust relationships, symmetric two-way trust relationship, and the whole set of trust relationships are defined. The details of the definitions are described as follows.

Definition 7: *One-way trust relationship is the trust relationship with a unique trust direction from the trusters to trustees.*

One-way is the default feature of a trust relationship if there is no further description. Two-way trust relationship can be defined and used in information systems. Actually, two-way trust relationship is the result of binding two one-way trust relationships together. We define two-way trust relationship as follows:

Definition 8: *Two-way trust relationship TT' is the binding of two one-way trust relationships T =< R, E, C, P > and T' =< R', E', C', P' > with R' = E and E' = R. T and T' are the reflective trust relationships with each other in the two-way trust relationship.*

In the above definition, "binding" is the key word. If there are two one-way trust relationships between R and E but they are not bound with each other, then they are only two one-way trust relationships and there is no two-way trust relationship. When two one-way trust relationships are bound together, there is a two-way trust relationship and these two one-way trust relationships can be called reflective trust relationships with each other.

If the trusters and the trustees are the same, the trust relationship is reflexive. The reflexive trust relationship is defined as follows:

Definition 9: *Trust* relationships T =< R, E, C, P > is a reflexive trust relationship when R = E.

The symmetry of two trust relationships could be an important concern in the analysis or modeling of trust relationships in distributed information systems. The symmetry of two trust relationships is defined as the follows:

Definition 10: *If there is trust relationship T' =< R', E', C', P' > which is the result of swapping trusters and trustees in another trust relationship T =<R, E, C, P > (the swapping includes all possible ownerships in condition set and property set), there is symmetry between T and T', T and T' are symmetric trust relationships with each other.*

In the above definition, the swapping of trusters and trustees includes all possible ownerships in condition set and property set. The two trust relationships have the same condition set and property set except the possible ownerships in them.

The symmetric/asymmetric two-way trust relationship is defined as follows:

Definition 11: *A two-way trust relationship TT' is symmetric two-way trust relationship if there is symmetry between T and T'; otherwise TT' is an asymmetric two-way trust relationship.*

Sometimes it is necessary to discuss the symmetry of all trust relationships between a truster set and a trustee set, we have the following definition:

Definition 12: *WTR(R,E) is the whole set of trust relationships with same truster set R and trustee set E.*

Definition 13: *If every trust relationship in WTR(R,E) has a symmetric trust relationship in WTR(E,R) and every trust relationship in WTR(E,R) has a symmetric trust relationship in WTR(R,E), the trust between R and E are symmetric.*

Scenario Example II: Here we use Microsoft's domain trust as a regressive scenario example to discuss the properties of trust direction and trust symmetry defined in this section. Domain trust allows users to authenticate to resources in another domain. Also, an administrator is able to administer user rights for users in the other domain. Our general definitions for the properties of direction and symmetry of trust relationships have general expressive power and can cover broad range of commonly used notations. The related concepts in domain trust can be viewed as specific cases of these general definitions. In the following, we will use our terms defined in this paper to review some concepts in domain trust.

- Based on **definition 1** in section 2, the domain trust can be expressed as "entities in domain A trust entities in domain B without any condition that entities in domain B have the right to get access of the set of resources in domain A".
- Microsoft's domain trust includes both one-way trust and two-way trust. In Microsoft's domain trust, one-way trust is defined as a unidirectional authentication path created between two domains. This means that in a one-way trust between domain A and domain B, users in domain A can access resources in domain B. However, users in domain B cannot access resources in domain A. Microsoft's one-way trust is an example of one-way trust relationship in **definition 7**. In a two-way domain trust, authentication requests can be passed between the two domains in both directions. Two-way trust is an example of two-way trust relationship in **definition 8**.
- The entities in same domain trust each other without any condition that entities have the right to get access of the set of resources in the same domain. This is an example of reflexive trust relationship in **definition 9**.
- There is symmetry in the two-way domain trust. The two one-way trust relationships bound in the two-way trust relationship are "entities in domain A trust entities in domain B without any condition that entities in domain B have the right to get access of the set of resources in domain A" and "entities in domain B trust entities in domain A without any condition that entities in domain A have the right to get access of the set of resources in domain B". These two one-way trust relationships are symmetric trust relationships with each other in **definition 10**. Microsoft's two-way trust is symmetric two-way trust relationship in **definition 11**.
- In domain trust, the WTR(A,B) based on **definition 12** has only one trust relationship from truster domain A to trustee domain B. For two-way domain trust, the trust between domain A and domain B is symmetric based on **definition 13**.

The above definitions about the properties of trust direction and trust symmetry are new elements of the taxonomy framework about trust. We believe that they can cover most situations related with direction and symmetry of trust relationship in the real world. These definitions can provide suitable terms and can be used as scenario examples in the analysis and modeling of trust relationships in distributed information systems.

D. Scope and Diversity of Trust Relationship

In this sub section, we will discuss the scope and diversity of trust relationship in distributed information systems. The diversity of trust has been discussed by Jøsang [14] who expresses trust in three diversity dimensions. The first dimension represents trusters or trust originators; the second represents the trust purpose; and the third represents trustees. Jøsang uses the term trust purpose based on the observation that trust is relative to a domain of actions. In our formal definition of trust relationship, trusters and trustees are two tuples and they are similar to the terms of Jøsang. The origin

diversity about trusters and target diversity about trustees are straightforward and have been described clearly by Jøsang [14]. Jøsang's term of trust purpose is related to a domain of actions. Under our taxonomy framework, we will define trust scope label to take the place of the trust purpose. There are multiple benefits of trust scope label other than the trust purpose and they will be discussed later in this sub section. The trust scope label is the binding of the condition set and property set based on the formal definition of trust relationship. The trust scope label is a new element of our taxonomy framework. The definition of trust scope label is expressed as follows:

Definition 14: *A trust scope label is a two-tuple TSL =< C, P > where C is a set of conditions and P is a set of properties.*

The details of condition set C and property set P can be found in the formal definition of trust relationship in section 2. Actually, trust scope label provides a new layer of abstraction under the trust relationship that only includes property set and condition set. To compare two trust scope labels $TSL_1 =< C_1, P_1 >$ and $TSL_2 =< C_2, P_2 >$, we have the following rules:

1. $C_1 \subseteq C_2$ and $P_1 \supseteq P_2 <=> TSL_1 \geq TSL_2$;
2. $C_1 = C_2$ and $P_1 = P_2 <=> TSL_1 = TSL_2$;
3. $C_1 \supseteq C_2$ and $P_1 \subseteq P_2 <=> TSL_1 \leq TSL_2$;
4. In other cases, TSL_1 and TSL_2 can not be compared with each other.

The trust scope label is beyond the trust purpose in several aspects. Trust scope label composes of a subspace of trust relationships (two tuples out of four tuples) and describes the characteristics of the combination of condition set C and property set P. Trust scope labels could be treated as an independent subspace of trust relationships in the analysis and design of overall information systems. The property set in trust scope label covers not only actions but also attributes of trustees. Two trust scope labels could be compared with each other based on the rules provided above.

Scenario Example III: Consider an online software shop. We assume that anybody who wants to enter the online shop must register as a member of the online shop first. For describing the condition set and property set in possible trust relationships between the shop and possible customers, we use the following notations:

- p_1 stands for that customers can read the documentation of the software.
- p_2 stands for that customers can download the software.
- c_1 stands for certificate of membership.
- c_2 stands for the commitment of the payment for the software.
- c_3 stands for the payment for the software.

We have the following trust scope labels:

1. $TSL_1 =< \{c_1\}, \{p_1\} >$

2. $TSL_2 =< \{c_1, c_2\}, \{p_1, p_2\} >$

3. $TSL_3 =< \{c_1, c_2, c_3\}, \{p_1, p_2\} >$

Based on the rules to compare two trust scope labels, we have

- TSL_1 cannot be compared with TSL_2 (or TSL_3). There is no obvious relationship between TSL_1 and TSL_2 (or TSL_3).
- $TSL_2 > TSL_3$. It means that the trust scope of TSL_2 is less strict than that of TSL_3.

The scope and diversity of trust is another aspect to be considered in the analysis and modeling of trust in distributed information systems. The trust scope label may be quite complicated and the above comparison rules provide helpful tools in making judgments. The scope and diversity of trust relationship may be coupled with other trust properties such as trust direction and trust symmetry.

VI. Lifecycle of Trust Relationships

A trust relationship has a lifecycle in distributed information systems. The whole life cycle of trust relationships includes several stages such as extracting trust requirements in system, identifying possible trust relationships from trust requirements, choosing and refining the whole set of trust relationships from possible trust relationships, implementing trust relationships in systems and maintaining trust relationships in systems. The life cycle of trust relationships includes three aspects: analysis and modeling of trust relationships; trust relationships at runtime; and the change management of trust relationships.

A. Analysis and Modelling Trust Relationships

The general methodology for analysis and modeling of trust relationships in distributed information systems is the main concern of this paper. As an aspect of lifecycle of trust relationships, the analysis and modeling of whole set of trust relationships in the system is one part of the analysis and design of the whole system and they are dependent on or strongly coupled with the design of other parts of the system. The analysis and design of trust relationships is a quite complicated process. We need to consider different aspects of the trust requirements in the target information system. The formal definition of trust relationship and the classification of trust relationship provide a starting point for the understanding of all trust issues. The operations and definitions provided in Section V will be used as terms and tools to analyze and model different specific properties of trust relationships in distributed information systems. The properties of target trust relationships are the main concerns to address trust requirements in distributed information systems. When we analyze a trust relationship, we must concern the properties of trust relationships from different angles. The operations on trust relationships and relative types of trust relationships provide terms and tools to discuss the relations between trust relationships. The definitions about properties of direction and symmetry of trust relationship and properties of scope and diversity of trust

relationship provide terms and tools to discuss the related properties. All the definitions can be viewed as scenario examples in the analysis and modeling of trust relationships.

The properties of trust relationships and relations of trust relationships will reconsidered in the multiple lifecycles of trust relationships. The modeling and maintaining of trust relationships is an incremental, iterative process. The set of the trust relationships initially modeled in an information system will be smoothed and refined in the life cycle of the information system to address the change of trust requirements or to modify and improve the information system.

The operations and definitions about the properties of trust relationships provide a set of tools in the analysis and modeling of trust relationships. These operations and definitions are based on our current understanding about trust relationships. It is possible to define more operations and properties.

B. Trust Relationships at Runtime

Trust is a vast topic which not only includes the analysis and modeling of trust relationships but also includes the evaluation and establishment of trust relationships at runtime. From the view point of system analysis and design, it is convenient to define trust relationship with truster set, trustee set, condition set and property set as four tuples of the definition of trust relationship. There are more arguments about why trust relationship is defined based on set of trusters(trustees) not on individual truster (trustee) [19]. From the view point of system running, trust is always evaluated based on one truster, one trustee, a set of condition and a set of properties. The set of properties in the evaluation should be a subset of property set in a trust relationship. An instance of trust relationship is defined as follows:

Definition 15: *When trust is evaluated based on trust relationship $T =< R, E, C, P >$ at runtime, only one truster r, one trustee e and requested properties p will be involved. There are $r \ni R$, $e \ni E$, $c \equiv C$, $p \subseteq P$. The $t =< r, e, c, p >$ is called an instance of trust relationship T.*

An instance of trust relationship must be assessable at runtime. The evaluating decision can be made by the involved truster based on the conditions for the trusting of the trustee with the set of properties. In Section III, we have described the two layers of trust relationships. At runtime, the authentication trust on layer one is always evaluated at first, then the trust relationship on layer two can be evaluated. The trust evaluation is actually to judge that the conditions of a trust relationship can be satisfied or not. Actually the conditions of trust relationship are against different risks. The risks maybe come from actions of trustee or third parties and unstable environments. All conditions can be classified into two subtypes as pre-conditions and post-conditions. Pre-conditions are existing evidences and they could be evidence credentials from trustees, local stored data or knowledge of trusters, environmental data, and remote data from third parties (maybe community based reputation). Post-conditions are post evidences (such as non-repudiation or guaranty) that are commitments from trusters. At runtime, trust is always between exactly two entities. Trust is normally non symmetric, but symmetric trust could be defined in specific situations based on related definitions about the direction and symmetry of trust relationship in Section V

C. Change Management of Trust Relationships

Normally, trust relationships are not static and it is necessary to modify and refine them to reflect the changing business requirements. Change management of trust relationships focuses on how to introduce or update trust relationships in a consistent manner and how to deal with the dynamic evolution of trust relationships in distributed information systems. Change management of trust relationships is related with the analysis and modeling, implementation, evolution, and management of trust relationships in distributed information systems. In dynamic environments, change management of trust relationships is a challenging issue.

In order to accommodate new business requirements, it may be necessary to introduce new trust relationships. When new trust relationships are introduced, the change management of trust relationships must be considered. It may be necessary to remove some existing trust relationships out of the system as well. These new trust relationships or removed trust relationships are related with some applications. It is possible that these applications are executing during the change for introducing or removing some trust relationships. There are three possible strategies for the related change management. The first strategy is to let all the related executing applications be completed according to the old trust arrangement. The second strategy is to abort all related executing applications and restart the applications according the new trust arrangement. The third strategy is to allow a migration from the old trust arrangement to the new one. There are some conditions for the third strategy and it may be very complicated [21].

VII. TRUST MANAGEMENT ARCHITECTURE

Our main objective about trust research is to create a powerful set of tools to enable trust management in distributed information systems. The analysis and modeling trust relationships is important in the trust management but it is not the end of the whole story. How to merge the trust relationships into the overall distributed information systems is another important topic and it provides lots of challenges. To provide an overall solution for trust issues in distributed information systems, we are currently working on a trust management architecture. The main aim of the trust management architecture is to establish infrastructure and tools of trust management for developing distributed applications.

The trust management architecture must support a wide range of different context-based trust policies. The trust policies are normally task-specific and they may be supported by multiple mechanisms. A trust policy can

require one or multiple trust relationships. When a trust policy is enforced, the required trust relationships in the trust policy must be satisfied. The trust decision is context-based and it is based on the evaluation of one or multiple trust relationships.

To illustrate trust management architecture, we propose TrustEngine to hold all trust related components that could be separated from applications. The formal definition of trust relationship is the starting point of the trust management architecture. TrustEngine could address applications' trust requests like a database query engine. TrustEngine accepts request with input as a trustor, a trustee, a set of conditions, and a set of properties which describe the actions or attributes of the trustees. Depending on the form of the query, TrustEngine can return yes/no answer or additional restrictions that would make the trust evaluation possible. Trust relationships are defined and loaded into the TrustEngine when applications are being developed. At runtime, single instance of trust relationship is evaluated.

TrustEngine is a container for all trust components and it has the flexibility to be expanded easily to hold new trust components. Each component in TrustEngine performs some trust function or has some data storage to be used by other trust functions. TrustEngine has TrustDatabase for storage of trust related data. TrustEngine includes components packages: TrustControl, LocatingTrust, EvaluatingTrust and ConsumingTrust. The architecture is expressed in the following Figure.

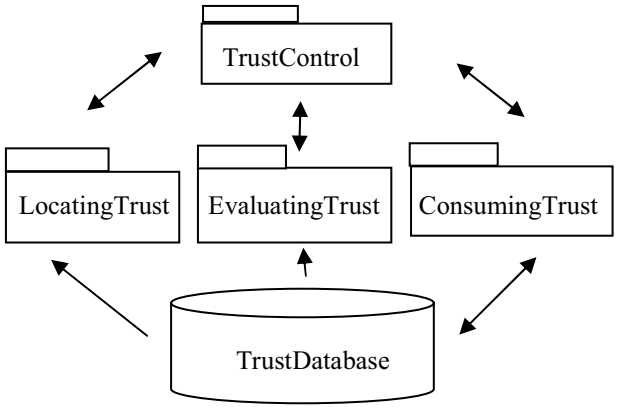

Fig.3. TrustEngine Package Hierarchy

TrustDatabase is the data storage of TrustEngine to look after trust relationships and other trust parameters. It is necessary for TrustEngine to have a persistent storage mechanism for storing and retrieving information about trust. The storage mechanism can be relational database or data profile. After trust relationships have been analyzed and modeled based on trust requirements in an information system, these trust relationships must be loaded into TrustDatabase before the information system is running. There may be other data as trust parameters that should be loaded into the TrustDatabase as well. At runtime, it may be necessary to store instances of trust relationships.

TrustControl is the package for the overall management and control of TrustEngine at run time. TrustControl links applications and functional packages of TrustEngine (LocatingTrust, EvaluatingTrust and ConsumingTrust).

LocatingTrust is the package for finding the requested trust relationship. It receives the request from applications and finds the requested trust relationship from TrustDatabase.

EvaluatingTrust contains all computing components for the evaluation of a trust relationship. The evaluation of a trust relationship is to check that the conditions of a trust relationship can be satisfied or not. The conditions of a trust relationship are against risks from evil actions of trustees, evil actions from other parties, and unstable environments. There are different conditions in a trust relationship and there may be multiple mechanisms to support the trust evaluation. The existing reputation-based systems and credential based systems can be employed as components of trust evaluation. Any successful systems or mechanisms for checking or evaluating of evidence and reputation can be included in EvaluatingTrust. Existing standards and successful systems related to trust can be put into the trust management architecture easily. EvaluatingTrust has functional components for specific evaluating tasks such as credential evaluation, reputation evaluation, stored data evaluation, and environment evaluation. In real implementation, the package of EvaluatingTrust will be customized based on the requirements and normally some components will not be involved. On the other hand, new components may be added to support other mechanisms or functions if necessary.

ConsumingTrust contains the computing components for consuming trust. Consuming trust deals with how to use the output of the evaluation of a trust relationship. The evaluation of a trust relationship is not always be consumed immediately. The result of evaluation of trust relationship can be stored and distributed in different ways. There are three normal ways to use the output of trust evaluation. The first way is that the result of trust evaluation is immediately used by consuming applications. The second way is to generate credentials with the result of trust evaluation as input. These credentials will be used in the future by the same or other applications. The third way is that the result of trust evaluation is stored in database and the data will be retrieved and used by applications in the future.

The main concern of this paper is about the analysis and modeling of trust relationships in distributed information systems. Trust management architecture is the next step target of our research about trust. The details of trust management architecture are beyond the scope of this paper and they will be described in a separate paper.

VIII. Concluding Remarks

In this paper, we have discussed the general methodology of analysis and modeling of trust relationship in distributed information systems. We have discussed the classification of trust relationships and put

authentication at layer one of trust relationship and other trust relationships on layer two. We have proposed a hierarchy of layer two trust relationships based on the nature of four tuples of a trust relationship. We provide guidelines for the analysis and modeling of trust relationships in distributed information systems. We review the operations on trust relationships and the relative types of trust relationships. We provide a set of definitions for the properties of direction and symmetry of trust relationships. We define trust scope label to model the properties of scope and diversity of trust relationships. All the definitions proposed in this paper are elements of our taxonomy framework of trust relationships and they can be used as enabling tools in the analysis and modeling of trust relationships in distributed information systems. We provide some discussions about the lifecycle of trust relationships. The lifecycle of trust relationship includes the analysis and modeling of trust relationships, trust relationships at runtime, and the change management of trust relationships. We propose trust management architecture for the overall solution about trust issues in distributed information systems. The trust management architecture is currently at a high level.

In real implementations, the properties of trust relationships discussed in this paper will be customized and configured based on the specific requirements. The definition of trust relationship provides a starting point and it is the cornerstone for our research about trust. The classification of trust relationships is helpful for better understanding of trust relationships. The guidelines for analysis and modeling of trust relationships provide high level guide to define a good set of trust relationships in distributed information systems. The operations on trust relationships and relative types of trust relationships provide terms and tools to concern the relations of trust relationships. The definitions about the properties of direction, symmetry, scope and diversity of trust relationships can provide suitable terms for the related properties and they can be used as tools for enabling the analysis and modeling of trust relationships in distributed information systems. The discussed lifecycle of trust relationships and proposed trust management architecture place the analysis and modeling of trust relationships under the background of trust management.

Web services can be viewed as specific distributed information systems. In the web services paradigm, the proposed properties of trust relationships and the general methodology for analysis and modeling of trust relationships can provide solid foundation to understand and deal with trust related issues in WS-Trust, WS-Security, WS-Policy and WS-Federation.

REFERENCES

[1] TCSEC. Trusted computer system evaluation criteria. Technical report, U.S.A. National Computer Security Council, 1985. DOD standard 5200.28-STD.

[2] J. Landauer, T. Redmond, and T. Benzel. Formal policies for trusted processes. In Proceedings of the Computer Security Foundations Workshop II, 1989, pages 31–40. 1989.

[3] Deutsch, M.: Cooperation and trust:some theoretical notes. In: Nebraska Symposium on Motivation. Nebraska University Press (1962)

[4] S. Marsh. Formalising trust as a computational concept. Phd thesis, University of Sterling, 1994.

[5] A. Jøsang. The right type of trust for distributed systems. In Proceedings of the 1996 New Security Paradigms Workshop, pages 119–131. ACM, 1996.

[6] A. Jøsang. An Algebra for Assessing Trust in Certification Chains. In *Network and Distributed Systems Security (NDSS'99) Symposium*. 1999.

[7] Ernesto Damiani, De Capitani di Vimercati, Stefano Paraboschi, Pierangela Samarati, and Fabio Violante. A reputation-based approach for choosing reliable resources in peer-topeer networks. In *CCS '02: Proceedings of the 9th ACM conference on Computer and communications security*, pages 207–216. ACM Press, 2002. Washington, DC, USA.

[8] S. Lee, R. Sherwood, and Bobby Bhattacharjee. Cooperative peer groups in nice. In INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications *Societies. IEEE*, volume 2, pages 1272–1282 vol.2, 2003. TY - CONF.

[9] Karl Aberer and Zoran Despotovic. Managing trust in a peer-2-peer information system. In CIKM '01: Proceedings of the tenth international conference on Information and knowledge *management*, pages 310–317. ACM Press, 2001. Atlanta, Georgia, USA.

[10] M. Schillo, M. Rovatsos, and P. Funk. Using trust for detecting deceitful agents in artificial societies. Applied Artificial Intelligence Journal, Special Issue edited by Castelfranchi, C., Tan, Y., Falcone, R. and Firozabadi, B. on Deception, Fraud and Trust in Agent Societies., 14(8):825–848, 2000.

[11] Josep M. Pujol, Ramon Sang, esa, and Jordi Delgado. Extracting reputation in multi agent systems by means of social network topology. In *AAMAS '02: Proceedings of the first international joint conference on Autonomous agents and multiagent systems*, pages 467–474. ACM Press, 2002. Bologna, Italy.

[12] M. Blaze, J. Feigenbaum, and J. Lacy. Decentralized trust management. In *Proceedings of IEEE Symposium on Security and Privacy*, pages 164–173. 1996.

[13] M. Blaze, J. Feigenbaum, and A.D. Keromytis. KeyNote: Trust management for public-key infrastructures (position paper). *Lecture Notes in Computer Science*, 1550:59–63, 1999.

[14] Y. H. Chu, J. Feigenbaum, B. LaMacchia, P. Resnick, and M. Strauss. REFEREE: Trust management for Web applications. *Computer Networks and ISDN Systems*, 29(8–13):953–964, 1997.

[15] G. Suryanarayana, J.R. Erenkrantz, S.A. Hendrickson, and R.N. Taylor. Pace: an architectural style for trust management in decentralized applications. In *Software Architecture, 2004. WICSA 2004. Proceedings. Fourth Working IEEE/IFIP Conference on*, pages 221–230, 2004. TY - CONF.

[16] M. N. Huhns and D. A. Buell. Trusted autonomy. Internet Computing, IEEE, 6(3):92–95, 2002.

[17] W. H. Winsborough, K. E. Seamons, and et al. Automated trust negotiation. In Proceedings of DARPA Information Survivability Conference and Exposition, 2000.

[18] M. Winslett, T. Yu, and et al. Negotiating trust in the web. IEEE Internet Computing, 6(6):30–37, 2002.

[19] W. Zhao, V. Varadharajan, and G. Bryan. Modelling trust relationships in distributed environments. In Lecture Notes

in Computer Science, volume 3184, pages 40–49. Springer-Verlag, 2004.
[20] T. Grandison and M. Sloman. A survey of trust in internet application. IEEE Communications Surveys, pages 2–16, Fourth Quarter, 2000.
[21] H. Skogsrud, B. Benatallah and F. Casati. Model-driven trust negotiation for Web services. IEEE Internet Computing, 7(6):45–52, 2003