Research on Trustworthiness Evaluation Method of Software Resources Based on Fuzzy Sets

Yonghao Wang
School of Computer & Communication Engineering, University of Science & Technology Beijing, Beijing100083, China
Department of Computer, Beijing Electronic Science & Technology Institute, Beijing 100070, China
Email: wyh15@sohu.com

Guangping Zeng, Qi Wang and Qingchuan Zhang
School of Computer & Communication Engineering, University of Science & Technology Beijing, Beijing100083, China
Email: zgping20012002@yahoo.com.cn, wangdadingkou@163.com, zqc1982@126.com

***Abstract*— With the increasing demands on software functions, the software systems are becoming larger and more difficult to be managed. The damages caused by system failures are more serious, so the software trustworthiness has become a focus that the international experts and scholars pay close attention to. In the paper, the intension of software trustworthiness has been discussed, and an evaluation method based on fuzzy sets is presented. Finally, a case study is made and the result shows that the designed evaluation method helps to solve the problem about trust evaluation of software resources in open and dynamic environment.**

***Index Terms*—software trustworthiness, trust evaluation, trust evidence model, trust requirement model, fuzzy sets**

I. INTRODUCTION

The information infrastructure centered on communication, storage and computation has been applied to political, economic, cultural and all aspects of social life. It has become a powerful motive force for development of modern productivity and the progress of human civilization. Software is the soul of the information infrastructure. Along with the increasing demands on software functions, the software systems are becoming larger and more difficult to be managed. It is also difficult to avoid the defects and loopholes, which makes the systems more fragile. The frequent failures directly or indirectly cause loss to users. That is to say, the software systems are not always trusted. This is the so-called software trustworthiness problem.

The concept of software trustworthiness was first proposed by Anderson in the early 1970s, which drew wide attention from the field of academia and industry [1]. In papers [2-7], the intension of software trustworthiness is mainly discussed in two aspects: One is the objectivity, referring to the quality of service (Qos) of the software system with excellent characteristics of safety, reliability and availability. The other is the subjectivity, referring to subjective identification of users to the software. In a word, software trustworthiness means not only the comprehensive evaluation of those characteristics, but also the degree of consistence between behaviors of the software and expectations of users. However, how to evaluate the trustworthy level of software resources is a hot issue.

At present, some solutions to the trustworthy evaluation of software resources has been put forward. The trust management proposed by M.Blaze has been defined as adopting an unified method to describe and explain the security strategy, security certificates and trust relationship used to directly authorize key safety operation. Based on this definition, the trust management includes the definition of security policy, the access to security certificate, and the judgment on the safety certification set complying with the relevant security policy [8]. Meanwhile, many other trust management systems have also been developed, such as PolicyMaker, KeyNote and REFEREE [9]. The trust management is based on the rational judgment, namely, the safety certification. However, it is obviously difficult to meet the application demand of large-scale resource sharing and integration in Internet. It involves lots of anonymous and mobile software resources when the resources are shared and integrated in the open coordination environment.

The other solution is the trust evaluation, which argues that the trust is a kind of subjective and irrational behavior and the presentation of experiences. Therefore, the trustworthy level of software resources can be evaluated by constructing the scientific trust model based on the past collaborative experiences. This method is more suitable for practical application of current Internet. The method of trust evaluation needs to solve the following key problems. Firstly, how to get

Manuscript received August 22, 2013; revised September 27, 2013; accepted October 1, 2013.
Corresponding author: Yonghao Wang.

doi:10.4304/jsw.8.12.3214-3221

collaborative data of the software comprehensively and accurately in the past? Secondly, how to construct the scientific trust model and evaluate the objective trust level based on these collaborative data? Thirdly, how to compare the behaviors of software resources with expectations of users?

Currently, some institutes have proposed some trust models used to evaluate the trust level of software resources, typically including Beth model [11], Jøsang model [12], and SSTEM model [13], etc. But these models are mainly based on the probability statistics or arithmetic statistics. It means that the subjectivity and uncertainty of trust is regarded as randomness, and this is obviously inconsistent with the inherent characteristics of trust. According to the definition by Tyrone Grandison, the trust means the confidence of independent, safe and reliable executing ability in a particular context [14]. So the trust is a kind of subjective judgment and expectation of possibility, and has features of subjectivity, uncertainty and fuzziness. Although [15] takes the subjectivity and fuzziness of trust into consideration, it doesn't reflect the expectation of the user and the endorsement of software quality.

Therefore, the paper proposes a trust evaluation method of software resources based on fuzzy set theory. The theory was proposed by L.A.Zadeh in 1965, which had the features of describing and solving fuzzy concepts and objects [16]. The fuzziness of trust can be analyzed quantitatively by using the membership and fuzzy linguistic variables in fuzzy sets, which makes the trust model more realistic. In addition, the paper describes the detail of how to acquire the collaborative information in past and how to compare with the expectation of the user.

II. Description of Trust

Software trustworthiness refers to the trust level of software resources and has features of fuzziness and uncertainty. So it can be described by using the membership and fuzzy linguistic variables in fuzzy sets.

Definition 1 *Let the domain be a non-empty set X, and x is the element in X. For each $x \in X$, mapping is given as follows:*

$$X \to [0,1], x \mapsto \mu_A(x) \in [0,1] \qquad (1)$$

Then for each $x \in X$, the sequence pair set $A = \{x | \mu_A(x)\}$ is called the fuzzy sub-set in X, or fuzzy set. $\mu_A(x)$ is called the membership function of x to A. For a specific x, $\mu_A(x)$ is called the grade of membership of x to A.

If $\mu_A(x)$ is closer to 1, it means that the extent of x belonging to A is higher. If $\mu_A(x)$ is closer to 0, it indicates that the extent of x belonging to A is lower.

Definition 2 *The fuzzy linguistic variable regards the words or sentences of natural or artificial language as the variable of value domain. Although the words or sentences aren't more accurate than number, the fuzzy linguistic variables can be used to describe these concepts which are fuzzy, complicated, or uncompleted in definition, or unable to be described in precise terminology.*

Let L be the name of linguistic variables, such as age and deviation, U_L be the domain of L, $S(L)$ be the value set of linguistic variable L, G_L be the grammar rules used to generate the value of L, and M_L be the semantic rules used to generate the membership function of the fuzzy subsets, then the fuzzy linguistic variables is descripted as a five tuples $(L, S(L), U_L, G_L, M_L)$ [17].

According to the above definition, the fuzzy linguistic variable trust can be described in table I:

TABLE I.

DEFINITION OF FUZZY LINGUISTIC VARIABLE TRUST

Linguistic Variable T	**Trust**
Domain U_T	{ 1,2,3 }
Linguistic Variable Value Set $S(T)$	{Distrust, Basic Trust, Trust}
Grammar Rules G_T	The rules of connecting the fuzzy linguistic variable trust with the qualifiers denoting level such as not, basic etc.
Semantic Rules M_T	Determining the membership function of fuzzy set represented by each linguistic value.

Among them, the domain U_T represents the different levels of trust by using the discrete numerical value set, and the numeric relationship of size in U_T reflects the degree of trust levels. The linguistic value set can be endowed with linguistic variables T intuitive and practical implication.

Because the fuzzy linguistic variable trust has three subsets, which means it is unable to accurately judge ascription relationship among them, namely the relationship among sub sets is not the exclusive relationship of one or the other, the trust vector consisting of membership degree of each fuzzy set can be used to represent this object. Formal trust vector can be expressed as V={ $v_0, v_1, \ldots, v_M$ }, where v_j denotes the membership degrade of x_i to T_j.

III. Collection of Trust Evidences

Trust is an embodiment of experience, and the experience is based on the interaction in the past. The interaction is also known as trust evidence. The

trust evidences are the basis of the trust evaluation on software resources. Only when the comprehensive and accurate trust evidences are obtained, the objective evaluation result can be guaranteed. Trust evidences can be obtained from various sources [18]. For example, the evidences of software resources according with setting goal are collected from the standardized process of the production, designation and management during the development procedure; the evidences of credible features are collected from analysis, testing and verification tools after submitting the software; the evidences of evaluation and feedback are collected in the process of using software resources, etc. But the paper focuses on the evidences during using the software, including the quality of service (QoS) information in runtime and the user feedback information. The former can objectively reflect the real situation of software in runtime, but the information that can be obtained is limited. The latter mainly depends on the subjective judgment of users, but it is uncertain whether these feedbacks can reflect the quality of software accurately and objectively. Therefore, it helps to be complementary mutually and evaluate the software more objectively and accurately if the objective and subjective evidences are introduced simultaneously.

In order to describe the collected evidence information, it is necessary to define the evidence model, which helps to provide uniform management mechanism for evidence collection. The evidence model includes two aspects. One is about evidence attributes contained in the model. Different users and applications have different requirements to the trust evidences, with the application area of software resources constantly enlarging, the trust attributes from users will also be expanded. So it is difficult to construct a model containing all requirements. The other is about the methods of collecting the evidences for each attribute. The collection methods are different for different evidence attributes. For example, the QoS information must be collected in real time, and to be necessarily quantized through the formulas. At present, the quantitative methods about reliability, availability, throughout, response time and reusability have been studied in papers [19-26]. However, the subjective evidence information is generally evaluated through the scoring method. To specify the value range of relevant attributes from 0 to 100, users can make evaluation according to the software being used. Furthermore, due to the influence of time, environment or user, there can be several quantitative methods for the same QoS attribute. For example, the quantitative method of availability includes the method based on the time repairing [24], calling numbers [25] and time range [26] etc. Each of these methods can be applied to different scenarios. So the most important thing is not to list quantitative methods of all attributes, but to provide an extended method which allows users to customize the quantitative methods according to the requirement. Therefore, an evidence model is described as a three tuples: < *Attributes*, *Rules*, *Values* >.

·*Attributes* represents evidence attribute, and is described by three tuples <*id, name, parentId*>. Among them, the *id* and *name* are respectively used to identify the uniqueness and name of evidence attribute. The *parentId* illustrates the parent attribute of this evidence attribute, thus a hierarchical relationship among attributes is formed.

·*Rules* represents the collection rule of attribute evidences, and is described by the four tuples <*id, name, attrId, rule*>. Among them, the *id* and *name* are respectively used to identify the uniqueness and name of collection rule. The *attrId* means the evidence attribute this rule belongs to. The *rule* represents the specific collection method of this rule.

·*Values* represents the storage manner, and is described by the seven tuples <*id, name, rulesId, attrId ,softwareId, time, value* >. Among them, the *id* and *name* are respectively used to identify the uniqueness and name of collected evidences. The *rulesId* means the collection rule of the evidence. The *attrId* means the evidence attribute the collected data belongs to. The *softwareId* means the software that the collected evidence belongs to. The *time* means the collection time of the evidence. The *value* means the collected instance value of the evidence attribute.

The model allows users to extend the evidence attributes and quantitative methods according to the requirement, and obviously has good scalability.

III. Model of Trust Requirement

After collecting the trust evidences, it is necessary to evaluate these evidences. A standard of evaluation is required. Because there are different requirements in trust evidences for different applications, it is necessary to allow users to define the standard of evaluation, including the evaluated attributes, evaluation standard of every attribute, namely the expectation of user for the attribute, and the weight of every attribute. It is also known as the trust requirement of users.

In order to better describe the trust requirement and the divergence caused by the differences in application fields and tasks, a customizable trust requirement model is presented in the paper as follows.

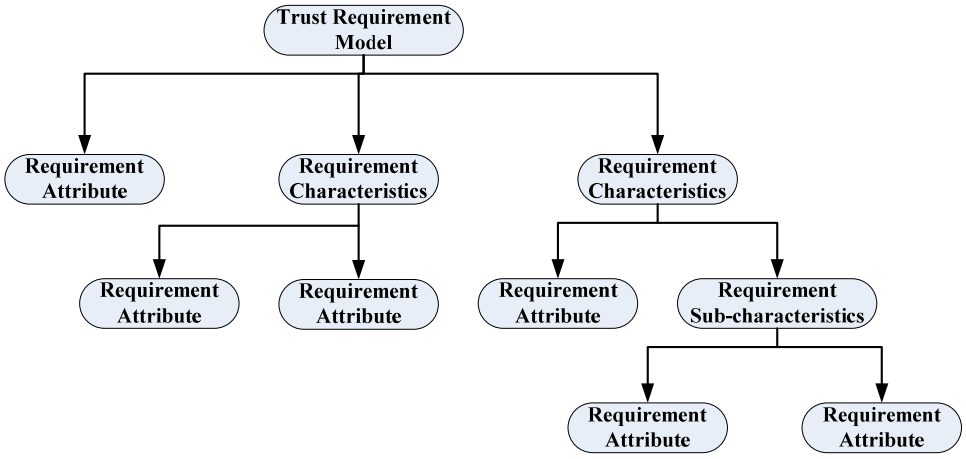

Figure 1. The model of trust requirement

As shown in figure 1, the model has an extended and hierarchical tree structure and can be described as the following ways:

<Model>::=(<Requirement_Attribute>)|(<Requirement_Characteristics>)**
<Requirement_Characteristics>::=<Characteristics_Id><Characteristics_Name>
(<Requirement_Attribute>)|(<Requirement_Sub-characteristics>)**
<Requirement_Sub-characteristics>::=<Sub-characteristics_Id><Sub-characteristics_Name>
*(<Requirement_Attribute>)**
<Requirement_Attribute>::=<id><name><Attribute_Id>< Weight> <Expect_Value_Low> <Expect_Value_Up>

Each of the trust requirement models can contain zero or more requirement attributes and zero or more requirement characteristics which are uniquely identified by *Id*. The requirement attribute is atomic and indivisible, while the requirement characteristics can contain zero or more requirement attributes and zero or more requirement sub-characteristics. The requirement sub-characteristic may contain one or more requirement attributes. As the indivisible atomic attribute, the requirement attribute also contains *Weight*, *Expect_Value_Low* and *Expect_Value_Up* apart from Id. Among them, *Weight* represents the weight of the attribute in the requirement model, *Expect_Value_Low* indicates the floor of expected value for the attribute, and *Expect_Value_Up* indicates the ceiling of expected value.

Because users can not only customize the requirement attributes according to their application requirements, but also can specify weights and expected value of the attribute, therefore, the model has good customizability and flexibility.

However, the model is constructed from the perspective of users and reflects the expectation of users for the software, while the evaluation basis of software is from evidence model. It is necessary to establish the relationship between requirement model and evidence model. The relationship is mainly reflected in the attributes of requirement model which must be contained in the evidence model. Otherwise, it is impossible to collect the evidence instance data, or to evaluate the attribute. Therefore, *Attribute_Id* is added to the requirement model, indicating the attribute corresponding to which attribute of evidence model.

V. EVALUATION OF TRUST

In order to help users to exactly find the right software, it is necessary to establish a comprehensive evaluation system of software trustworthiness, which is used to classify the software with the same model. In the paper, the fuzzy comprehensive evaluation method is used to analyze the trust quantitatively. The method is a quantitative evaluation model on the basis of fuzzy mathematics. It makes a comprehensive evaluation on the software using the fuzzy set theory and considers the comprehensive influence on trust from all the factors associated with the software. The basic steps of the method are as follows:

Step1 Determining the evaluation factor set of trust, $E=\{e_1,e_2,\ldots,e_n\}$. Because there are different requirements on software resources for different users and applications, the number and type of evaluation factors should be determined according to specific requirements.

Step2 Determining the evaluation set of trust, $D=\{d_1,d_2,\ldots,d_M\}$. The subscript of M can be set to three, which means to define three sub-sets as *{trust}*, *{basic trust}*, *{distrust}*. It also can be redefined according to actual requirement.

Step3 Establishing the factor evaluation matrix. By building the fuzzy map between factor set E and evaluation set D, the matrix element r_{ij} can be gotten.

Step4 Determining the weight allocation of various factors, $W=\{ w_1,w_2,\ldots,w_n \}$. Different weights must be given under the influence of various factors at the extent of the object. The weight should comply with the objectivity, orientation and measurable principles.

Step5 Executing the fuzzy comprehensive evaluation, and getting the trust vector. The formalized representation is as follows:

$$(v_0, v_1,\ldots, v_M)=(w_1,w_2,\ldots,w_n)\circ (r_{ij})_{n\times M} \quad (2)$$

Where "$\circ$" represents the fuzzy transformation. Operators can be determined according to the

specific situation, and common operators include Zadeh, Einstein operator, etc.

VI CASE STUDY

There is a component providing air ticket subscription service in software resource base, and the user wants to reuse the component. Therefore, the user puts forward a trust requirement model, as shown in figure 2. The weight and expected value of each trust attribute is shown in table Ⅱ.

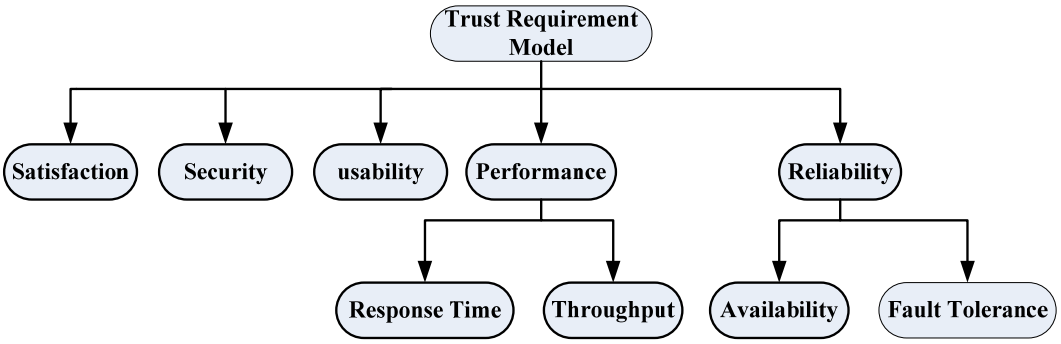

Figure 2. The trust requirement model of air ticket subscription component

TABLE Ⅱ.
THE WEIGHT AND EXPECTED VALUE OF TRUST ATTRIBUTES

Attribute Name		Weight	Expected Value Floor	Expected Value Ceiling
Performance	Response Time	0.2	0.9s	1s
	Throughput	0.2	36 times/min	40 times/min
Reliability	Fault Tolerance	0.05	80%	90%
	Availability	0.1	90%	98%
Satisfaction		0.2	80	90
Security		0.15	90	98
Usability		0.1	80	90

In the past reused situations of this component, the collected evidences of the related trust attributes are shown in table Ⅲ.

In the example, the trust levels of software are classified as three fuzzy sets: *{Trust}*, *{Basic Trust}* and *{Distrust}*. If the collected value of evidence attribute is between the floor and ceiling of expected value of user, the value is classified to the fuzzy set of *{Basic Trust}*. If the value is worse than the floor of expected value, value is classified to the fuzzy set of *{Distrust}*. If the value is better than the ceiling of expected value, the value is classified to the fuzzy set of *{Trust}*. According to the method, the statistical number belonging to each of the fuzzy sets is shown in table Ⅳ.

TABLE Ⅲ.
COLLECTED VALUE OF EVIDENCE ATTRIBUTES

Attribute Name		Value							
Performance	Response Time	1.03	1.08	0.83	1.05	0.86	0.89	0.95	0.92
	Throughput	33	35	38	37	41	36	45	42
Reliability	Fault Tolerance	80%	95%	93%	77%	76%	92%	89%	91%
	Availability	93%	99%	88%	90%	100%	99%	100%	95%
Satisfaction		86	98	93	97	88	96	90	100
Security		93	100	96	99	100	99	97	99
Usability		77	88	86	90	83	94	93	91

TABLE Ⅳ.
STATISTICS OF EVERY FUZZY SET

Attribute Name		Trust	Basic Trust	Distrust
Performance	Response Time	$3(C_{11})$	$2(C_{12})$	$3(C_{13})$
	Throughput	$3(C_{21})$	$3(C_{22})$	$2(C_{23})$
Reliability	Fault Tolerance	$4(C_{31})$	$2(C_{32})$	$2(C_{33})$
	Availability	$4(C_{41})$	$3(C_{42})$	$1(C_{43})$
Satisfaction		$5(C_{51})$	$1(C_{52})$	$2(C_{53})$
Security		$5(C_{61})$	$3(C_{62})$	$0(C_{63})$
Usability		$3(C_{71})$	$4(C_{72})$	$1(C_{73})$

Then the trust level of the air ticket subscription service can be evaluated as follows by taking advantage of the fuzzy comprehensive evaluation method.

Step 1 Determining the evaluation factor set of trust, $E=\{e_1,e_2,\ldots,e_7\}=\{$ *response time, throughput, fault tolerance, availability, satisfaction, security, usability*$\}$.

Step 2 Determining the evaluation sets of trust , $D=\{d_1,d_2,d_3\}=\{$ *trust, basic trust, distrust* $\}$.

Step 3 $C_{ij}(i=1,2,3,4,5,6,7;\ j=1,2,3)$ indicates the numbers of $e_i(i=1,2,3,4,5,6,7)$ belonging to $d_j(j=1,2,3)$.

$$r_{ij}=\frac{Ci_j}{\sum_{j=1}^{3} Ci_j}(i=1,2,3,4,5,6,7) \qquad (3)$$

Thereby, establishing the factors evaluation matrix as follows:

$$R=\begin{bmatrix} 0.375 & 0.25 & 0.375 \\ 0.375 & 0.375 & 0.25 \\ 0.5 & 0.25 & 0.25 \\ 0.5 & 0.375 & 0.125 \\ 0.625 & 0.125 & 0.25 \\ 0.625 & 0.375 & 0 \\ 0.375 & 0.5 & 0.125 \end{bmatrix}$$

Step 4 Determining the weight allocation of various factors , $W=\{\ w_1,w_2,\ldots,w_7\ \}=\{0.2,\ 0.2,\ 0.05,0.1,0.2,\ 0.15,\ 0.1\}$.

Step 5 Executing the normalization, and getting the final trust vector:

$$V=W_\circ\ R=\{v_1,v_2,\ v_3\ \}=(\ 0.48,\ 0.31,\ 0.21\ )$$

The result of the calculation means that the extent of trust is 48%, that of basic trust is 31%, and that of distrust is 21%. According to the principle of maximum membership, the component can be trusted.

VII. Implementation Framework of Trust Evaluation Method

According to the evaluation method of trust above, an implementation framework is provided, just as shown in figure 3. The framework includes the user interface layer, the evaluation and the monitoring layer and the running platform of online software resources. Among them, the user interface layer provides the

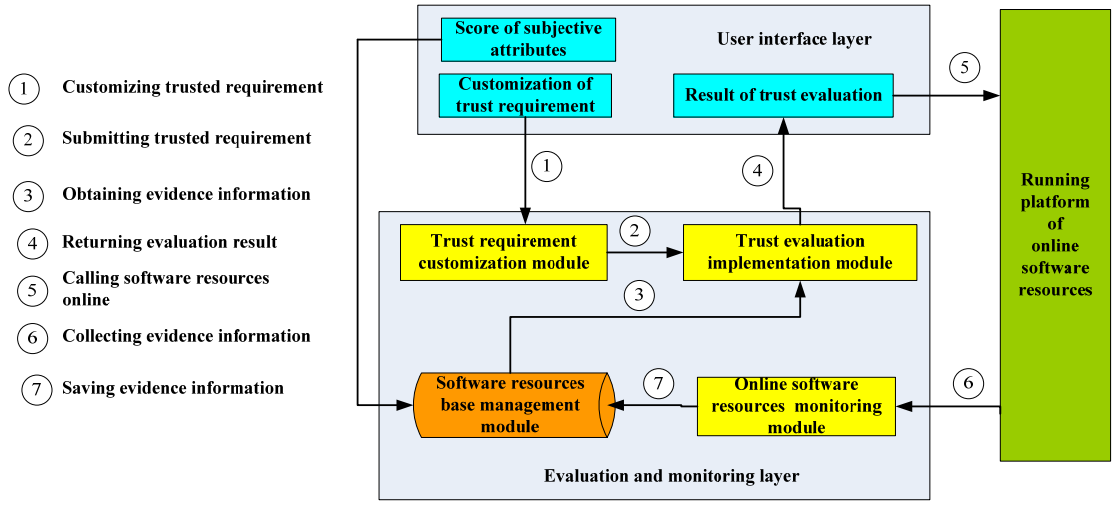

Figure 3. Implementation Framework of Trust Evaluation Method

services of customizing trusted requirement and scoring for the subjective attributes of software resources for the users, and submits the scores to the software resources base management system. Meanwhile, the layer also receives the result of trust evaluation from lower layer, and selects out the software resources from running platform of online software resources on base of the results. The evaluation and monitoring layer includes the modules of trust requirement customization, trust evaluation implementation, online software resources monitoring and software resources base management. The trust requirement customization module provides extended functions of customizing trust requirement such as trusted attributes selection and weight assignment for users. The trust evaluation module obtains the related evidence information from software resource base management system according to the customized model of trust requirement. Then it evaluates the trustworthy level of software resources according to the specific method of trust evaluation and returns the evaluated results to the user by the user interface layer. The software resources base management system is responsible for managing and monitoring collected evidence information. The online software resource monitoring module is responsible for collecting the runtime status information of software resources in running platform. Then it quantizes the information based on the defined quantitative rules in evidence model and provides the quantitative evidence information to the software resources base management system for storage and management.

Conclusion

Now the software resource bases provide the reused software resources for the software developer with the openly, publicly accessed and highly dynamic form. However, due to the lack of an effective software

evaluation mechanism, the users may download the incompetent software resources from the bases and therefore damage their interests.

Thus, the paper proposes a method supporting the trust evaluation of software resources and makes a detailed analysis of the method, such as the customization of trust requirement model, the collection of the trust evidence instances and the trust evaluation method based on the fuzzy sets etc. In addition, the paper also demonstrates how to use the method through the case and provides an implementation framework of the method. The method helps to solve the problem about trust evaluation of software resources in open and dynamic environment.

ACKNOWLEDGMENT

We thank the National High Technology Research and Development Program of China (863 Program) (No.2009AA01Z119) and the National Natural Science Foundation of China (No.60973065) for partially supporting our research.

REFERENCES

[1] Anderson J P. Computer Security Technology Planning Study. ESD-TR-73-51, Vol. I, AD-758 206, ESD/AFSC, Hanscom AFB, Bedford MA, October 1972.

[2] ISO/IEC, “Information technology-security Techniques-Evaluation Criteria for IT Security, Part 1: Introduction and General Model. (Available at URL: http://standards.iso.org/ittf/PubliclyAvailableStandards/c040612_ISO_IEC_154081_2005(E))” , 2005.

[3] Trusted Computing Group, “TCG Architecture Overview”, (Available at URL: https: //www.trustedcomputinggroup.org/specs/IWG/TCG_1_0_Architecture_Overview.pdf, accessed on April 14, 2006) , 2004.

[4] Gates Bill, “Trusted Computing”, Wired News, (Available at URL: http://www.wired.com/news/business/0,1367,49826,00.html, accessed on April 14, 2006), 2002.

[5] Guoyuan Lin, Yuyu Bie, Min Lei. Trust Based Access Control Policy in Multi-domain of Cloud Computing. Journal of Computers, 2013, VOL.8, NO.5, pp.1357—1365.

[6] Wang Huaimin, Tang Yangbin, Yin Gang. Trustworthiness principle of network software. Science in Chinese: Series E, 2006, Vol.36, No.10, pp.1156-1169.

[7] Mei Hong, Cao Donggang. Software trustworthiness: the challenge of Internet. Communications of the China Computer Federation, 2010, Vol. 6, No.2, pp.20-27.

[8] Blaze M., Feigenbaum J. and Lacy J. Decentralized Trust Management. IEEE Conference on Security and Privacy, 1996, Oakland, California, USA.

[9] Yang-Hua Chu, Joan Feigenbaunn, Brian LaMaeehia, PaulResnic k, and Martin Srtuass. REFEREE: Trust management for Web applications. Computer Networks and ISDN systems, 1997, 29(8-13), pp.953-964.

[10] Abdu1.Rahman A, Hailes S. A distributed trust mode1[C]// Proceedings of the’97 New Security Paradigms Workshop．Cumbria：ACM, 1997, pp.48—60.

[11] Beth T, Borcherding M, Klein B. Valuation of trust in open network[C]// Proceeding of the European Symposium on Research in Security(ESORICS)．Brighton：Springer-Verlag, 1994, pp.3-18

[12] Audun Jøsang. An Algebra for Assessing Trust in Certification Chains[C]// Proceedings of NDSS’99, Network and Distributed System Security Symposium, The Internet Society, San Diego, 1999

[13] Ruizhong Du, Xiaoxue Ma. Trust Evaluation Model based on Service Satisfaction. Journal of Software, 2011, vol.6, No.10, pp.2001-2008

[14] Tyrone Grandison, Morris Sloman. A survey of Trust in Internet Applications. IEEE Communications Surveys and Tutorials, Fourth Quarter 2000.

[15] ZhengzhenZhou, YonglongLuo, LiangminGuo. A trust Evaluation Model based on Fuzzy Theory in P2P networks. Journal of Computers, 2011, VOL. 6, NO. 8, pp.1634-1638.

[16] L.A. Zadeh. Fuzzy sets. Information and Control, June 1965, Volume 8, Issue 3, pp.338-353.

[17] Zadeh L A. The Concept of a Linguistic Variable and Its Application to Approximate Reasoning[M]. Beijing: Science Press, 1982, pp.63-84.

[18] Wang Huaming, Tang Yangbin, Yin Gang, Li Lei. Trustworthiness of Internet-based software. Science in China—Series F: Information Sciences, 2006, Vol.49, No.6, pp.759−773.

[19] Zhao Junfeng, Wang Yasha, Xie Bing. A management framework of component supporting QoS of component. Chinese journal of electronics, 2004, Vol.32, No.12A, pp.165-168.

[20] Yanan Hao, Yanchun Zhang, Jinli Cao. A novel QoS model and computation framework in web service selection. World Wide Web, 2012, Volume 15, Issue 5-6, pp. 663-684.

[21] Shi Shuangyuan, Chen Zhaode. Research on the trustworthiness evaluation of business components. Computer engineering and science, 2009, Vol.31, No.5, pp.84-86.

[22] Mao Guobei, Li Xuejing, Ge Xiaokun. The metrics and application of software-based component quality model. Computer application and software, 2005, Vol.22, No.5.

[23] Cao Buqing, Liu Jianxun, Li Bing. A QoS-assured Trust-driven Semantic Web Service Discovery Method. Journal of Convergence Information Technology, 2011, Vol. 6, No. 7, pp. 162-171.

[24] Xavier Franch . systematic formulation 0f non-functional characteristics of software. In Proceeding of 3rd International Conference on Requirements Engineering(ICRE). USA : Colorado springs, IEEE Computer Society, 1998.

[25] M.Mikic-Rakic, S.M., N. Medvidovic. Improving availability in large, distributed component-based systems via redeployment. Component deployment. International Working Conference on, Grenoble, FRANCE, 2005. German: Springer, 2005, pp.83-98.

[26] Zeng L Z, Bemtalhh B, Dumas M. Quality driven web services composition. WWW2003[C]. Budapest, Hungary. 2003, pp.411-421.

Yonghao Wang received M.S. degree from Beijing University of Posts and Telecommunications in 2004. He is currently working toward the Ph.D. degree in the department of computer science and technology, University of Science and Technology Beijing, China. His research interests focus on information security and distributed computing.

Guangping Zeng received Ph.D. degree from University of Science and Technology Beijing, China, in 1999. He is currently a professor and Ph.D. supervisor of the School of Computer and Communication Engineering with University of Science and Technology Beijing, and Director of the Center of Smart System and Soft Computing (CSSSC). His research areas include Distributed and Migrating Computing, Linux Operating System and Embedded Systems, Intelligent Robotics and SoftManTechnology, Intelligent Network and Communications, and Smart Systems and Soft Computing.

Qi Wang is currently working toward the Ph.D. degree in the department of computer science and technology, University of Science and Technology Beijing, China. His research interests focus on artificial intelligence and services computing.

Qingchuan Zhang received M.S. degree in school of Information Science & Engineering from Hebei University of Science and Technology in 2008. He is currently working toward the Ph.D. degree in the department of computer science and technology, University of Science and Technology Beijing, China. His research interests focus on intelligence software and distributed computing.