PROPAGATING TRUST IN THE WEB SERVICES FRAMEWORK

Anitta Thomas and Lucas Venter

School of Computing

UNISA

thomaa@unisa.ac.za

ventelm@unisa.ac.za

ABSTRACT

In this paper, we discuss one possible way of establishing trust between a user entity and a web service. We will assume that the user entity is machine-based (i.e. some software, a mobile agent or even perhaps a web service).

After a brief introduction, we will consider the concept of Trust and the Web Services framework in two separate sections. In the case of Trust, we will show that two aspects are important: on the one hand it is possible to quantify trust in order to establish a trust relationship, and on the other hand we stress the relevance of the context for which you wish to establish the trust relationship. One possible context includes security. In the case of the Web Services framework, we will show that a web service can have an extremely complex structure, which complicates the propagation of trust.

The paper will conclude with a description of one possible model, which could be used to establish a trust relationship between a user entity and a complex web service. The model makes use of the underlying XML structure of Web services. It allows the user to specify parameters, which reflects the trust context as well as the users' interpretation of the various aspects of a trust relationship.

KEY WORDS

Web Services, Trust, Trust establishment

PROPAGATING TRUST IN THE WEB SERVICES FRAMEWORK

1 INTRODUCTION

In recent years, managing Trust in the interaction between Human and machine has become more important than ever before. This is evidenced by the establishment of the iTrust initiative (See [15]). **iTrust** is an Information Society Technologies Working Group, which started on 1st of August, 2002. The Working Group is being funded as a Concerted Action / Thematic Network by the Future and Emerging Technologies unit of the IST programme. Two international conferences have been quite successful, and a third conference on this subject is envisaged to take place in 2005.

In this paper, we will consider the concept of Trust within the Web Services framework. We will show that establishing trust within this framework is not a trivial matter. In the case of Trust, we will show that two aspects are important: on the one hand it is possible to quantify trust in order to establish a trust relationship, and on the other hand we stress the relevance of the context for which you wish to establish the trust relationship. One possible context includes security. In the case of the Web Services framework, we will show that a web service can have an extremely complex structure, which complicates the propagation of trust.

We then turn our attention to a model for establishing trust. Since the model is not yet fully developed, we give only a brief description. The model can be specified in the underlying XML structure of the web service, thereby ensuring a common understanding of trust. Since establishing trust is based on some information, we then turn our discussion to an analysis of some factors that can influence trust. Our aim is not to present an exhaustive list, but to show how these factors can influence trust. The paper concludes with a discussion of possible future work.

2 TRUST

Trust is a complex concept and has been studied and presented in different perspectives in the literature. Most of the studies of trust focus on the application of trust in information security [1, 5, 6, 10, 16].

For instance, in a distributed system, key certificates can be used to bind the identity of an entity with its public key. A set of trusted entities is used to issue key certificates. This means that users of the system will trust that a certain public key belongs to a given entity, if that entity presents its (trusted) certificate. At the same time, such a certificate does not ensure any other characteristic of its owner. Using such a certificate it is not feasible to determine whether an entity is malicious or not. It is up to the entity or the application that receives the certificate to make a decision about the suitability of the public key owner for the intended purpose [1, 6, 7]. In general, such systems rely on the fact that the certificate issuers are trusted, but do not specify how this trust relationship is established [16]. At the same time, understanding trust, ways of building trust and its practical implications are not given high emphasis. The role of trust in other contexts, other than information security is also not emphasised.

Different perspectives of trust have lead to different definitions of trust [2, 5, 10, 14, 16]. For the purpose of this discussion, *Trust* is defined as a quantified belief, based on some set of guidelines, by an entity (*belief holding entity*) of another entity (*target entity*), to have a desired property or to behave in a specific way, for a specific purpose. The two key aspects of this definition are the quantification of trust and the application of trust in any context. These aspects will be discussed in the following sections.

Returning to the example above, the target entity could present its certificate to two different belief holding entities. One of these might accept the certificate, since it trusts the issuer, while the other might not know the issuer, and hence reject the certificate. The first belief holder can trust that the public key belongs to the target entity. It cannot deduce any further information from this trust relationship; for instance, it cannot trust the ability of the target entity to perform a certain task efficiently.

2.1 Quantified trust

The behaviour of an entity is influenced by many factors; some might be unknown, and others might be unpredictable or uncertain [22]. For this reason, trust is conveyed as a belief; we believe that the entity will behave in some way, but we are unable to predict or control this behaviour with a high degree of certainty. Obviously, the quantified belief that outlines trust is based on some information about the target entity. It is also obvious that any change in the information collected can alter the trust value. For instance, as more experience is gained in interaction with the target entity or more recommendations are collected about the target entity, the trust value could change positively or negatively. Thus trust based on knowledge cannot be static and may need to be dynamically updated [5, 14, 26, 29]. The information collected to quantify trust is specified by the belief holding entity. As each entity can have its own set of guidelines, different belief holding entities can have different trust believes of the same entity, even in the same context and for the same purpose [2, 16, 27]. Even if two entities are guided by similar guidelines the trust value can still be different based on the differences in the information collected, or the purpose for which the trust value will be used [11].

Quantification of trust is utilised in various trust models and different trust computation algorithms are used for the quantification process [1, 2, 5, 3, 14]. The trust computation techniques that quantify trust varies in its complexity, in terms of the variables used and thus the input parameters required for the computation. Two factors that are inclusive in the majority of the trust computation schemes are direct experiences of the belief holding entity with the target entity and / or recommendations of other entities about the target entity. Other information such as digital credentials, reputation of the target entity, system guarantees, system certifications, system specifications and assessments or evaluations made by other entities [1, 2, 3, 5, 11, 14, 16, 17, 20, 22, 26, 27, 29, 30, 31] can also be used. Based on the computation scheme, these factors are mapped into different trust values and various calculations are made available to combine these trust values. The models that use these quantified values use a threshold value (trusted / distrusted) or a range of values (highly trusted / trusted/ distrusted / highly distrusted) to base a trust based decision. There are also trust computation schemes that include factors like utility, risks, benefit and role of an entity in the computation schemes [28, 2, 27, 20].

2.2 Context

A trust relationship is always dependant on a certain context. Two entities might exchange public keys using a certification scheme. They now trust one another's public keys, and can hence exchange information securely. Hence, the context of this relationship is security. However, the trustworthiness of the information is not guaranteed by the current trust context. In order to establish the validity of the information, a different trust relationship would have to be established. It is immediately clear that a variety of trust contexts can be specified, such as competence, reliability, honesty, security, dependability, availability, reusability, scalability, and so on. The context depends on the purpose for which interaction needs to be established, and will determine the type of information necessary to quantify the trust.

3 WEB SERVICES

The Web Services framework aims to provide a structure to support the collaboration between diverse software components on the Internet, by providing standardised interfaces and interactions

of the components. The framework also supports a structure for publishing services provided by software components over the network [12, 23]. These framework standards and technologies are based on open XML standards and the Internet protocols [9, 13, 24, 25]. The Web Services framework specifies the standards such as Simple Object Access Protocol (SOAP), which is the standard communication protocol of the framework, Web Service Definition Language (WSDL), which is the service description language, Universal discovery, description and integration (UDDI), which specifies the web service registry, and so on [9, 12, 25]. The general web services architecture primarily consists of three components namely, the service provider, the service requester and the service registry. The figure below illustrates general architecture in conjunction with some of the framework's standards.

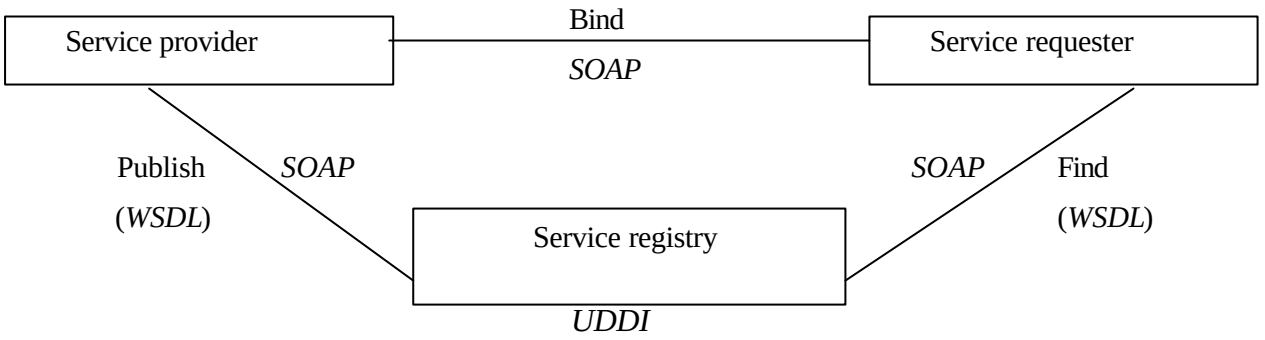

Figure 1

An organisation wishing to supply web services takes the role of a service provider. The service provider publishes the services in the service registry so that the availability of the service is visible to the intended web service users. This service description includes details of the interface, its implementation, network location, and so on. Thus, service descriptions should include sufficient information for a service requestor to access the service [12, 23, 21]. The service registry is a look-up registry that holds the list of services published by the service providers. Hence the service registry is the key entity that facilitates the discovery of services.

The software components conforming to this framework are autonomous, modular and intend to provide a specific functionality over the network [12, 13]. The Web Services framework allows these services to be utilized by other software components or services and thus allowing the incorporation of different services over the network. The software components integrated using the Web Services framework can be implemented and maintained by different organisations, and hence the framework aim to facilitate the interaction of heterogeneous software components across organisational boundaries [8].

As an example, consider an application to implement a service to find the availability of seats for a bus journey. Assume that there are three bus services namely 'Luxury', 'Speedy' and 'Express' in a particular location. These bus services have their own software components implementing own business logic and these components are exposed as web services. This implies that they adhere to the specifications of the Web Services framework and their functionalities can be utilised by other software components over the network. In this example, consider another web service developed by another organization, which is capable of interacting with these three bus web services. The combined bus web service component will implement the logic of the application with necessary coordination controls. Ultimately, it provides the service of finding the availability of seats from these three bus services. An illustration of the interactions in this service is given below.

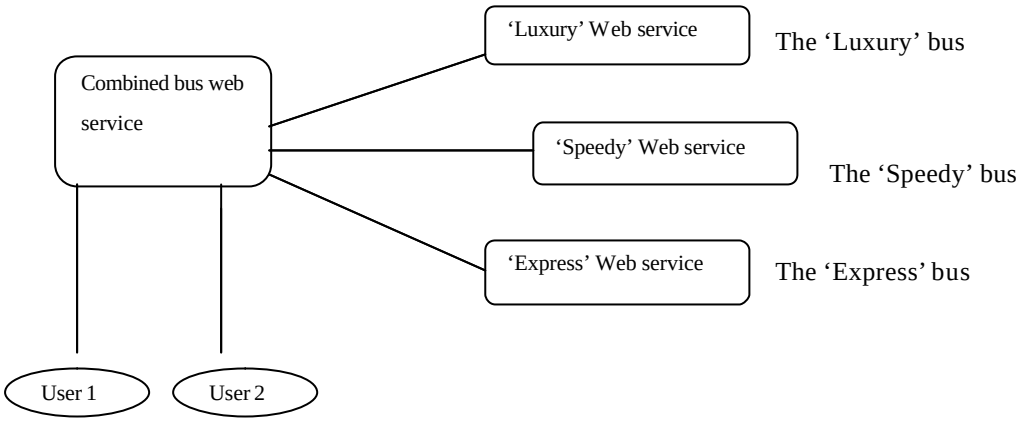

Figure 2

As illustrated in the figure, the application is accomplished using four different services, which are provided by four different service providers. Such a service can be referred to as a *composite service* as it utilizes the services of multiple services and indeed, this composite service can form a component in other composite services [4]. Thus composite services can become complex, consisting of any number of services and multiple interactions among the services.

In a composite service, each of the constituent services can provide a set of interfaces for utilising their functionalities. At the same time a composite service can provide interfaces for utilising the complete service, which can hide the interaction details and the differences among its constituent services. The interfaces provided by the composite service are also capable of abstracting different multileveled interactions in a composite service. A user of such a composite service need not be aware of such multileveled interactions. The composite services can also be dynamic considering the possibility of being able to add, remove and substitute components in a service. Such a change may not affect all the interactions in the service.

In figure 2, the users, which can also be services or software entities, interact with the combined bus web service to utilize the service. For the user, the interactions of the combined bus web service with the other web services are indirect. At the same time, for the individual bus web services the interactions with the users are indirect. In this example, the interactions are always mediated through the combined bus web service. Also the interactions of the user with the bus web services are hidden by one level. In a more complex composite service, the depth of the indirect interactions can be high. Referring back to figure 2, the software component providing a bus web service can be changed to a different software component. Such a change may not affect all the interactions in the composite service.

The services utilised in a composite service can have differences in their security, reliability, quality, performance, scalability, reusability, etc. [9, 23, 25]. The success of an application developed from different components is based on the functioning of its constituent components. As an example, reliability of such a service will depend on the reliability of its constituent components. In such a scenario, an application is built using different entities and the application depend on its constituent entities, which are not always under the control of the same organisation or same local network domain [24]. This may introduce uncertainties about the properties of the entities and the factors influencing the behaviour of an entity. As the number of components in an application increases, issues related to for instance security, reliability, quality and performance of the application gets complicated.

Entities functioning in a composite service can have different goals and the existence of malicious entities cannot be denied. As each entity is motivated by different goals and due to the difference in the functional and non-functional capabilities of these components, a composite

service cannot be viewed strictly as a single, uniform entity. As the result, for a user entity a composite service cannot be trusted like a single entity and there could be entities with undesirable properties. Thus analysing these entities in any context for the purpose of building trust becomes relevant.

4 BUILDING TRUST IN A COMPOSITE SERVICE

According to the context and the purpose for which trust is established, the information used to build trust will vary. At the same time, for each belief holding entity, the consideration of information sources and the significance of each information source from which the information is collected for the purpose of building trust can also differ. The threshold values used by the belief holding entity to make a trust based decision for a specific purpose can also vary.

Figure 3 represents a computation scheme that can be used as a basis for building a trust computation algorithm. The computation scheme utilises multiple information sources and assigns different weights to different information sources. These weights will reflect the value that the belief holding entity places on each of the different sources. The computation scheme illustrated do not include an exhaustive list of information sources that can be used to build trust, as each of the sources can still be sub categorised into more specific information sources.

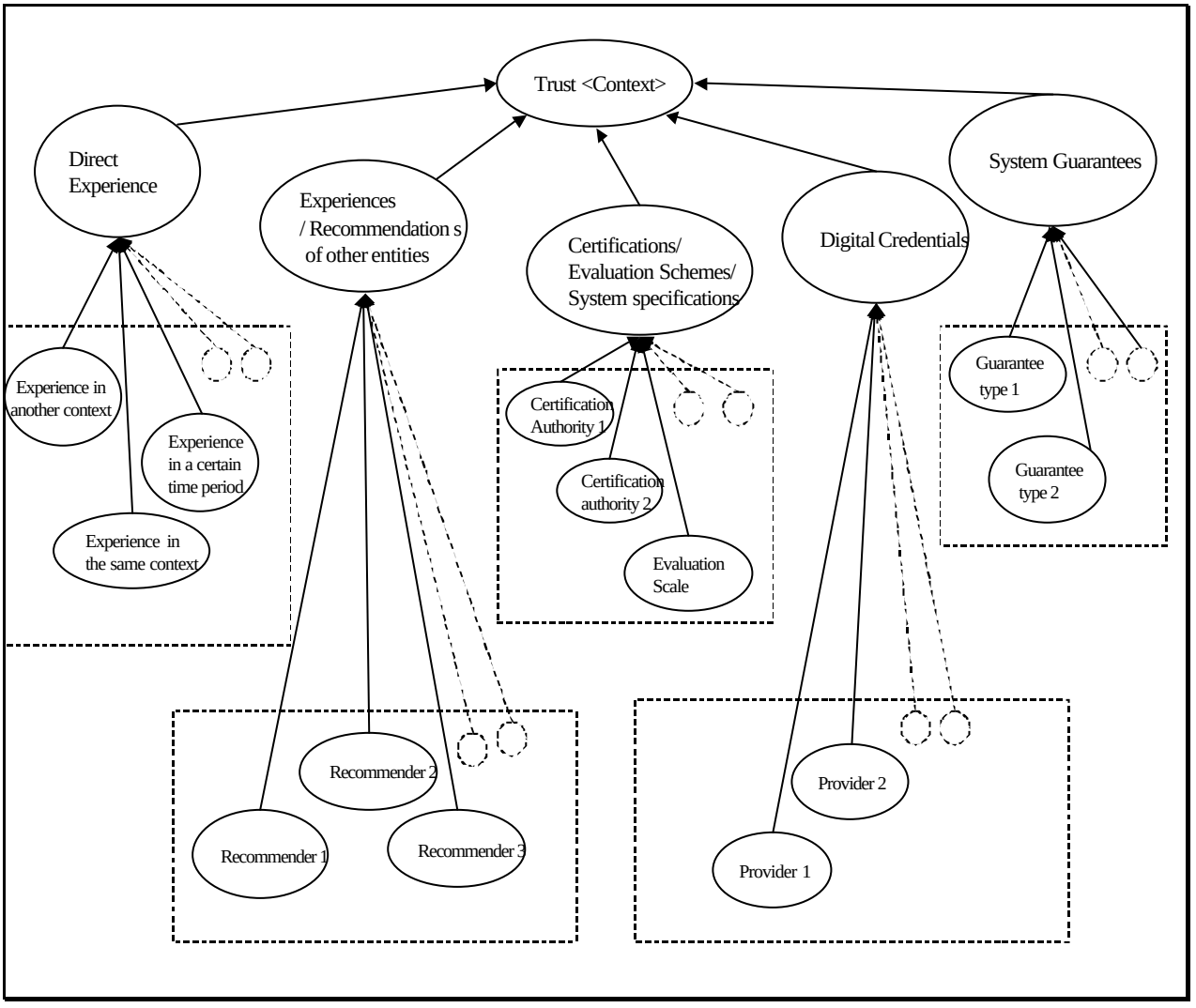

Figure 3

In the Web Services framework, such a computation scheme can be specified in the underlying XML structure. When a user entity intends to utilize a service it computes a trust value, by specifying the values of the different weights. The required input values can be gathered from the service itself or from other entities. The communication protocol SOAP can be tailored to request and carry trust related information among web services. When a trust related request is made by the user entity to a composite service the request must be propagated along different services in a composite service and this information must also propagate back to the user entity. According to the trust information request, a variety of trust related information could reach the user, which must also be combined appropriately to determine the trust value.

In the next section, we present some factors that can influence the trust computation, and which should be reflected in our model.

5 FACTORS INFLUENCING TRUST COMPUTATION IN COMPOSITE SERVICES

This section contains a list of some factors that can influence the trust computation, and which should be reflected in the computational model described above. This is not an exhaustive list; it simply highlights the way in which these factors can influence trust.

5.1 Accuracy of information

Much of the information needed to establish a trust relationship, can be supplied by the web service itself. This information could be accurate, or could be designed to mislead the user into falsely trusting the web service. Hence the user entity would want to establish the accuracy of the supplied information. For this purpose, the user can for instance use any of the recommender services described in the literature to support the information supplied by the web service [18, 19].

5.2 Complexity

A complex composite service can have any number of constituent services to realise the ultimate service. As the number of entities increase, the number of indirect interactions and the depth of the levels of interaction may increase. Hence, any request for trust building information from a user entity becomes a recursive process where the request is sent via the complex service to all relevant constituent services. When the complexity of a composite service increase, the depth of the path that the trust information request traverses down and trust information traverse back gets deeper.

When a service becomes complex, there could be links between constituent services that could infer unsolicited information regarding the user entity, which these constituent services cannot infer independently. Such a situation can be unacceptable to users.

5.3 Anonymity

In a composite web service, the identity of constituent services would in general may not be made known to the user entity. This might present a threat to the user, since some of the constituents might be malicious towards this specific user. Hence, the user might decide to request further information about the constituents. The trust computation will be influenced by the willingness of the service to supply this information.

The user entity might wish to remain anonymous to the constituent services. Hence it would have to trust the web service not to divulge its information to constituent services that does not require this information. When such information needs to be passed on to a constituent service, the user entity will have to be notified of this interaction. It can then decide whether it will trust this constituent service with its information, i.e. it can try to build a trust relationship with this constituent service.

5.4 Dynamic behaviour

Dynamic behaviour refers to the fact that constituent services could be added, deleted and substituted in the composite service at any time. When such changes occur, it can influence the trust values of different entities. Composite services have the capability to change dynamically in many different ways that could affect the trust values of different entities without changing any other interaction details.

5.5 Context

As mentioned earlier, the information required to build trust is based on the context. As an example, when the context of trust is the reliability of the service, this may require some information about the reliability of all the constituent services. When the context of trust is the security of the credit card transaction performed by a composite service, this context does not require relevant information regarding all the entities in a service, instead only information pertaining to a set of services that handle credit card transactions are required. This means, when a trust related request is made, the services must be able to understand the context, or at least the information requested as well as the number of entities that the information is pertaining to.

5.6 Example

Consider a situation in which a user entity interacts with the complex web service described in figure 2 above. The web service has requested credit card details from the user, and the user must decide whether it will trust the web service with this information. The context of the trust relationship is fixed as security, and hence security mechanisms can be used. The user would probably ask the web service to identify the constituent service (in the background) who will handle the credit card transaction. This could be done by asking for a digital certificate of that service. If the web service refuses this request, the user can terminate the interaction. If the request is granted and the user accepts the certificate, it can encrypt the credit card details using the public key of the credit card service, and in this way ensure that its identity and card details remains hidden from other constituent services in the background.

6 CONCLUSIONS AND FURTHER WORK

This paper deals with the establishment of a trust relationship between a user entity and a complex web service. In section 2, we gave a definition of trust, and showed that two aspects of trust are important. In the first place, trust can be quantified, and in the second place it is important to note that this quantification takes place within a certain context. In section 3, we gave a brief overview of the web services framework. We emphasize that a web service can be quite complex, and that this complex nature will lead to issues when establishing a trust relationship.

In section 4, we describe a possible framework that can be used to facilitate trust computations in the web services context. Our discussion does not give any details of the proposed model; this will be the topic of our future research. We will have to establish how to specify the trust computation in the underlying XML structure of the web service. Another aspect that needs further research is the factors that influence trust. We list a number of these in section 5, and we point out how they could influence the trust relationship. Our list is as yet not exhaustive. We will also have to find a way of quantifying these factors.

A grant from the Thuthuka fund in partial support of this work is gratefully acknowledged.

7 BIBLIOGRAPHY

[1] Abdul-Rahman A. and Hailes S.(1997). A Distributed Trust Model. *Proceedings of new Security Paradigm.* pp. 48-60.

[2] Abdul-Rahman A. and Hailes S.(2000). Supporting Trust in Virtual Communities. *Proceedings of the 33rd Hawaii International Conference on System Sciences*. pp. 1-9.

[3] Aberer K. & Despotovic Z.(2001). Managing Trust in a Peer-2-Peer Information System. *Proceedings of the Tenth International Conference on Information and Knowledge Management(CIKM01).* pp 310-317.

[4] Alonso G., Casati F., Kuno H. and Machiraju V.(2003).01.Web Services Concepts, Architectures and Applications. Springer-Verlag Europe.

[5] Au R., Looi M. and Ashley P. (2001). Automated Cross-organizational Trust Establishment on Extranets. *Proceedings of the IEEE Workshop on Information Technology for Virtual Enterprises*. pp. 1-9.

[6] Blaze M., Feigenbaum J., Ioannidis J. and Keromytis A.D.(1999) The Role of Trust Management in Distributed Systems Security. *Secure Internet Programming: Issues in Distributed and Mobile Object Systems*, Lecture Notes in Computer Science State-of-the-Art series Springer-Verlag. pp. 185 – 210.

[7] Blaze M., Feigenbaum J. and Lacy J. (1996). Decentralized Trust Management. *Proceedings of the 1996 IEEE Symposuim on Security and Privacy (SP'96)*. pp. 164-173.

[8] Bosworth A. (2001). Developing web services. *Proceedings of the 17t International Conference on Data Engineering (ICDE '01).* pp 477-484.

[9] Curbera F., Duftler M., Khalaf R., Nagy W., Mukhi N. and Weerawarna S. (2002). Unraveling the web services web. *IEEE Internet Computing*. pp 86-93.

[10] Della-Libera G., Dixon B., Garg P., Hallam-Baker P., Hondo M., Chris K., Maruyama H., Nadalin A., Nagaratnam N., Nash A., Philpott R., PrafullChandra H., Shewchulk J., Simon D, Waingold E. and Zolfonoon R. (2002). Web Services Trust Language (WS-Trust). Online. 29 pages. Available at http://www-106.ibm.com/developerworks/library/ws-trust. 27/08/2003.

[11] Essin J.D.(1997). Patterns of Trust and Policy. *New Security Paradigms Workshop*.pp.38-47.

[12] Gardner T.(2001). An introduction to web services. *Ariadne Issue 29*. Online. 4 pages. http://www.ariadne.ac.uk/issue29/gardner/intro.html. 27-02-2003.

[13] Gergic J., Kleindienst J., Despotopoulos Y.,Soldatos J., Polymenakos L. (2002). An approach to lightweight deployment of web services. *14th International Conference on Software Engineering and Knowledge.SEKE '02.*pp 635-640.

[14] Grandison T. and Sloman M.(2001). SULTAN-A Language for Trust Specification and Analysis. *8th Conference Proceedings HP-OVUA*. pp.1-9.

[15] iTrust, Trust Management. http://www.itrust.uoc.gr.

[16] Josang A.(1996). The right type of trust for distributed systems. *ACM new Security Paradigm Workshop*. pp. 119-131.

[17] Kalcklosch R. & Herrmann K. (2003). Statistical Trustability (Conceptual Work). Trust Management, *Proceedings of the First International Conference on Trust Management (iTrust 2003)*. Lecture Notes in Computer Science 2692 Springer-Verlag 2003. pp 271-274.

[18] Kinaterder M. and Rothermel K. (2003). Architecture and Algorithms for a Distributed Reputation System. Trust Management, *Proceedings of the First International Conference on Trust Management (iTrust 2003)*. Lecture Notes in Computer Science 2692 Springer-Verlag 2003. pp 1-16.

[19] Liu J. and Issarny V. (2004). Enhanced Reputation Mechanism for Mobile Ad Hoc Networks. Trust Management, *Proceedings of the Second International Conference on Trust Management (iTrust 2004)*. LNCS 2995 Springer-Verlag 2004. pp 48-62.

[20] Manchala D. W. (1998). Trust Metrics, Models and Protocols for Electronic Commerce Transactions. *18th International Conference on Distributed Computing Systems.* pp 312-321.

[21] McGregor C. and Kumaran S. (2002). Business Process Monitoring using Web Services in B2B e-commerce. *Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS'02).*pp 219b.

[22] Petrick A.S. (2002). Building Trustworthy Software Agents. *IEEE INTERNET COMPUTING. pp.* 46-53.

[23] Roy J. and Ramanujan A.(2001). Understanding Web services. *IT Pro.* pp.69-73.

[24] Shirky C. (2002). Web Services and Context Horizons. *Computer*. pp 98-100.

[25] Stal M.(2002).Web Services: BEYOND COMPONENT-BASED COMPUTING. *Communications of the ACM.* 45(10). pp 71-76.

[26] Swarup V. & Fabrega J.T. (1999). Trust: Benefits, Models, and Mechanisms. Secure Internet Programming, *Security Issues for Mobile and Distributed Objects.* Lecture Notes in Computer Science 1603 Springer-Verlag 1999. pp 3-18.

[27] Tan Y. & Theon W.(2000). Formal Aspects of a Generic Model of Trust for Electronic Commerce. *Proceedings of the 33rd Hawaii International Conference on System Sciences*. pp. 1-8.

[28] Varadharajan V. & Lin C. (2003). Modelling and Evaluating Trust Relationships in Mobile Agents Based Systems. *International Conference on Applied Cryptography and Network Security ACNS (2003)*. LNCS 2846 Springer-Verlag 2003. pp. 176-190.

[29] Winslett M., Yu T., Seamons K.E., Hess A., Jacobson J., Jarvis R., Smith B. and Yu L. (2002) Negotiating Trust on the Web. *IEEE Internet Computing*. pp. 30-37.

[30] Witkowski M. and Pitt J. (2000). Objective Trust-based Agents: Trust and Trustworthiness in a Multiagent Trading society. *Fourth International Conference on Multiagent Systems (ICMAS-2000)*. pp 463- 464.

[31] Yu T., Ma X and Winslett M. (2000). PRUNES: An efficient and complete Strategy for Automated Trust Negotiation over the Internet. *Proceedings of the 7th ACM conference on Computer and communications security CCS'00.* pp.210-219.