Composition of Trustworthy Web Services

***Karl Quinn, Declan O' Sullivan, Dave Lewis,**
Vincent Wade.
Knowledge and Data Engineering Group,
Trinity, College Dublin, Dublin 2, Ireland.
Tel: 00 353 1 608 1355
Email: firstname.lastname@cs.tcd.ie

Abstract: An ontology based trust model that can be used to semantically annotate web services was designed and developed to facilitate the computation of trust values for service selection and composition. This paper describes the research areas and presents an analysis of the salient issues experienced throughout a three month R&D project.

Acknowledgement: This work was carried out in the Ericsson Ireland Research Department as part of an Internship by the lead Author during 2004. The project was a joint effort between Ericsson Ireland and the Knowledge & Data Engineering Group (KDEG), Trinity College Dublin. The lead author is funded through an Irish Research Council for Science, Engineering, and Technology (IRCSET) scholarship.

I. Introduction

Definitions of trust generally use synonyms or trust inspiring terms. "Belief" [i], "Credibility or Reliability" [ii], "Confidence or Faith" [iii], "Reputation" [iv], and "Competence and Honesty" [v] have all been used in this way. Definitions generally try to convey that trust has a specific, quantitative, and directed (i.e. A to B, not B to A) value. Increasingly the definition of trust values and their calculation are seen as important elements of an overall security framework. Determination of an end-to-end trust value for a particular service can be brought about by reasoning over trust metadata associated with each of the individual service components. Web services will benefit from the use of trust metadata and management as it can aid in the automatic discovery or composition of trustworthy web services.

In parallel, a major direction of web service research is towards service collaboration and semantic annotation through ontologies [vi]. Ontology provides a means to describe and define terms, concepts, and relationships specific to a knowledge domain such as trust and services. The development of technologies for the Semantic Web [vii] has produced ontology languages such as the W3C's OWL (Ontology Web Language) [viii] which can be reasoned over at runtime. It is argued that trust and service metadata described with reference to such ontologies provide for improved reasoning because of its semantic basis. Trust management research is also exploring models for managing trust on an internet scale outside of the more traditional centralized systems. Furthermore trust management research has also exploring the use of policy based management [ix, x, xi, xii].

Given the above directions, our research focuses on combining ontology and policy based approaches for trust management. This involves the annotation of web services with trust metadata with reference to an ontology for trust and services. In addition users will be enabled to specify policies that determine how the management system should reason over the trust metadata for trustworthy service selection and composition.

This paper provides an overview of the key concepts of Semantic Web services, policies, and trust. It highlights important related work, and introduces our current experiment in the area and presents and assesses our experiences while implementing the project.

II. Key Concepts

II.1 Semantic Web Services

Web services can be defined as "self-contained, modular applications that can be described, published, located, and invoked over a network—generally, the World Wide Web" [xiii].Web services enable businesses, governments and individuals to easily integrate many of their applications that may be from heterogeneous environments, which can be both internal and external to the organization.

However a significant difficulty with the current web is that the majority of the resources have been primarily designed for use by humans. The idea behind the Semantic Web initiative is to provide tools and techniques to allow resources on the web to be augmented with information that would allow for greater interpretation and processing by computer applications directly.

Semantic Web services are the combination of web services and Semantic Web. In this approach, ontology languages from the Semantic Web community are used to describe the properties and capabilities of web services in an unambiguous, machine-interpretable form. Currently, the composition of web services relies more on manual hard-coding than on automated service orchestration. The promise of Semantic Web services is that they allow for the automation of web service discovery, invocation, interoperation, composition and execution monitoring. Early research has shown [xiv, xv] and confirmed [xvi, xvii, xviii] the availability of these kinds of automation.

II.2 Trust

Trust has various definitions that are applicable to different areas of computer security. Many definition use synonyms or trust inspiring terms such as "Belief" [i], "Credibility or Reliability" [ii], "Confidence or Faith" [iii], "Reputation" [iv], "Competence and Honesty" [v] have all been used in this way. Definitions generally try to convey that trust has some quantitative value associated with it such that A trusts B, but only by so much. Trust is multidirectional in that B may not trust A at all. Trust can be made specific even more by stating that A trusts B in relation to car maintenance but not with regards to medical procedures. Trust and its synonyms can be applied to many facets of the A and B relationship. It can be interpreted that A trusts the information that B (BBC News service) provides if A finds that the BBC's information is both credible and reliable. From this A can assign some level of confidence to the BBC's information and act in good faith upon it. The BBC can build up its reputation (expectation of behaviour based on past observations/information) by using competent reporters so that A has a directed and weighted belief in the BBC and its statements. The BBC can at the same time hold (to some degree) confidence in its audience.

Trust is a difficult issue to contemplate because it is such a human idea that has so many uses and general meanings. It is for this reason that the scoping of trust (much like security) can enable a more specific meaning that can be more efficiently translated into mission goals and statements. For the purposes of our research trust can be seen as the aggregation of many of the synonyms used above in conjunction with the ideals that they convey. Web services must be reliable and inspire confidence in their users. Their information must be credible (or from credible sources) and honest so that it can be believed. Together all these elements can create a reputation of trust that can help in such areas as the orchestration of composite web services. In the real world we tend to make choices about who we buy services like health insurance from based on trust because it gives us peace of mind, so it seems plausible that we should have similar notions of trust in the computing world in order to help make choices about online service composition.

II.3 Policy

A policy can be seen as a form of rule, which may change the behaviour of an entity. Generally, a policy is expressed in the form of the triple: Event; Condition; Action, where the event triggers an evaluation of the policy rule. The conditions are a set of stipulations that must be met in order for the policy to be enacted. If a policy is to be enacted then the actions state what is to be performed. Policy languages are typically vendor specific and proprietary [xix]. In generally, policy languages are split between access control and resource management languages [xx].

Policies are useful for applying a common set of rules to a large set of distributed nodes, services or users. Policies capture the business goals of an organization, which will allow these policies to govern how the organization's or individual's system operates. REFEREE [xxi] is a system that allows users to specify rules as policies in order to provide trust management for web applications. [xxii] uses policies to specify privacy rules for selecting web services.

III. Related Work

III.1 Semantic Web Services & Trust

The creation and utilization of domain specific ontologies, such as a trust ontology, on the Semantic Web will empower a richer semantic environment in which more powerful expression and reasoning may occur. Semantic Web services will take advantage of ontologies by using them in their advertisements and so enable more sophisticated and accurate service discovery and matching to occur.

Ontologies can be used by systems in order to express and calculate trust. The TRELLIS [xxiv] project purports to enable users to express their trust in a source (and the sources statements) so that an individual's trust can be combined into an overall assessment of trust. It is presented as an information analysis tool that enables users to annotate how they analyze and use information when making some decision. Friend-Of-A-Friend (FOAF) [xxv] is a project that utilizes the RDF [xxvi] vocabulary that allows users' to describe information about herself and her friends, which includes statements that can be used to build a web of acquaintances. A users' privacy is optionally maintained by use of signed files that specify anonymity or their true identity. The Advogato [xxvii] project also automatically calculates trust using group assertions. Interestingly, the Advogato metric for calculating trust is highly attack resistant. This allows the system to cut out portions of the network that are subsequently identified as 'bad'.

Advogato, FOAF and TRELLIS all provide a platform for the basis of a trust network that is commonly referred to as a 'Web of Trust', which is one of the ultimate goals of the Semantic Web.

There are a limited number of ontologies that are already defined for trust and reputation (a sub element of trust). In [iv] an extension is made to the FOAF ontology that allows for the assignment of a reputation value to a person. This extension to allow for the reputation value is similar to Golbeck's earlier work [ii] where the value assigned was for trust. Both [ii] and [iv] describe how trust/reputation can be applied to a person for a specific subject area. For example, Bob can state that he trusts Dan in relation to a certain area by such a degree or that Dan has a certain reputation in a specific area. The levels of trust used by Golbeck are defined at [xxix] where trust values are measured on a scale from one to ten, one being absolute distrust and ten being absolute trust.

III.2 Trust Management

Trust management systems for traditional computing environments such as PolicyMaker [x, xi] and KeyNote [xxviii] are two (similar) engines for granting authorization. Instead of the two step process of authentication and access control for processing a (signed) request these engines address the authorization problem directly. In the two step process the questions asked are "Is this person who they say they are?" and "does this person have the correct access control permissions for this request?". Even the reliable authentication of a user 'a priori' wouldn't help in deciding whether or not to execute the requested action of a user, if the user is unknown. Therefore, in directly answering the authorization problem the question asked becomes "is the key that signed this request authorized to take this action?" or as [x] puts it "Does the set C of credentials prove that the request complies with the local security policy P?". Any entity that responds to requests must have a (local) policy that acts as the ultimate source of authority on which decisions may be directly based upon. The policy can delegate this responsibility to credential issuers that it trusts and that have the required domain expertise as well as relationships with potential requesters. This delegation process enables entities to grant authorization to users' that it doesn't know 'a priori'. KeyNote builds upon PolicyMaker by adding two design goals; standardization and easy integration into applications.

Vigil [xxx] is a trust management system for pervasive computing environments that uses an ontological approach for the development of a Role Based Access Control like system. According to this research an ontological approach enables the system to extend trust based on the user's role, where roles can be changed based on a users actions or context. For example someone in a meeting room using the projector can be assumed to be the speaker and therefore the use of the computer could be granted to them based on their speaker context. The 'Security Agent', among other things, manages trust and receives new or altered access rights information and enforces policies in the local space. A system of delegation is used to allow users with no access rights to access a particular resource so long as a user who has the correct access rights delegates this ability. A delegation system like this could be used to handle unknown users a priori so long as a known user delegates the appropriate rights.

III.3 Policy Languages

Three common policy languages are Ponder [xxxi], Rei [xxxii], and KAoS [xxxiii]. Rei is an ontology based policy language that allows for the specification, analysis, and reasoning of policies. It enables users to express and represent the concepts of 'rights and prohibitions' and 'obligations and dispensations'. The concepts are represented as an ontology, which allows for greater interoperability with other policy languages and enables users to extend the ontology as required. Kagal has combined the separate research areas of an ontology based policy language with ontology based semantic web services. [xxii] describes how privacy policies could be written as Rei policies that are used during the discovery phase of service selection.

IV. Experimentation

IV.1 Background

The current state of the art in trust ontologies is rather elementary with only the very basic concepts of trust and sub-elements of trust developed. Prior to the completion of our initial project, a definitive (or even semi-complete) trust ontology had not existed.

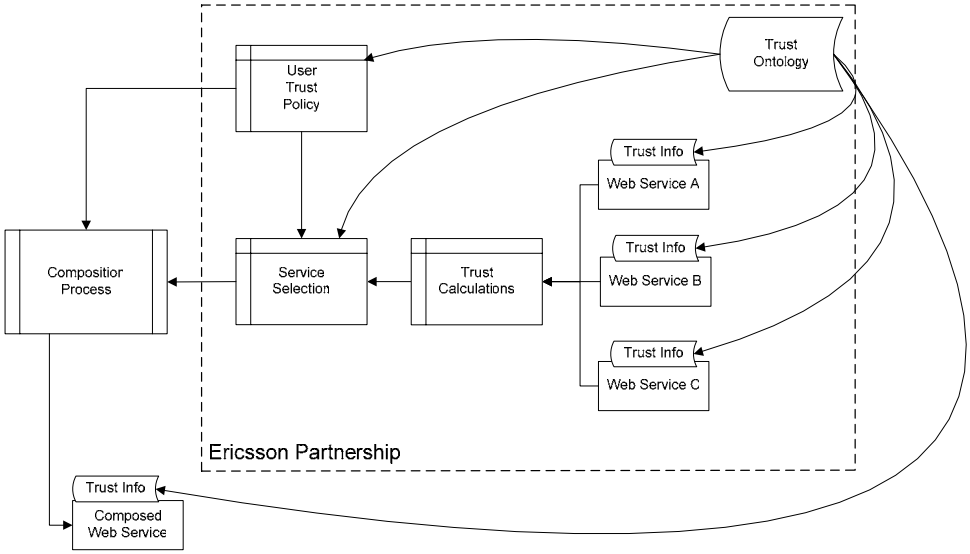

Figure 1: High Level Architecture

In a similar vein to Golbeck, but at a more complete level, our trust ontology was designed (using the Web Ontology Language OWL) to support the semantic annotation of web services in a fashion similar to Kagal's research efforts. Yet where Kagal offered privacy as a determining factor in the selection of a web service our experiment used trust in the selection and composition of web services. Like Blaze, Kagal, and the W3C, policies are used to describe security requirements, specifically trust, in order to locally enforce user defined rules.

The aim of our experimentation was to develop a platform that will allow Semantic Web service composition based on trust annotations and where policies will provide decentralized management for a user's trust requirements. The key challenges (as presented in figure 1) involved in this research are the development of an ontology specific to trust and services, the annotation of selected web services according to that ontology, and the creation of an application that reasons over the annotated services to establish end to end trust values. The initial steps in this experiment (with Ericsson) are highlighted within the dashed-line in figure 1.

IV.2 Experiment Overview
The initial project was scheduled over three months with the following critical paths:

(1) Design and develop an ontology based trust model
The trust model is encoded in OWL.
(2) Semantically annotation web services
Annotation information is stored in a service profile's service parameters.
(3) Develop trust calculations to compute trust values
Trust calculation sets are specific to the trust concepts within the trust model.

The most crucial aspect to the successful completion of the project was the design and development of the ontology based trust model. Without this model it would not have been possible to complete phase two and three as the trust calculations (3) are based on the trust structure (2), which is based on the trust model (1). The twelve week time frame saw six weeks devoted to the trust model, two to semantically annotating our chosen service, and the last four to the trust calculations, reporting, and sundry.

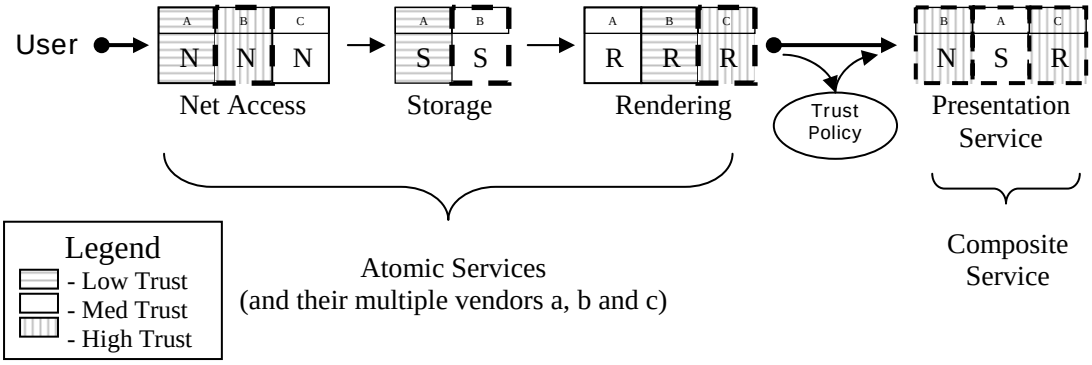

Figure 2: Service Selection and Composition

A typical use case scenario for the management application is presented in figure 2. In it our user wants to select three services (access, storage, and rendering) from different vendors (A, B, C) and compose a 'presentation service'. The services and/or vendors are unknown to our user and are therefore considered not to be trusted. The services have been annotated with trust structure information. Other users, who have used these service(s), store specific trust data about these services. The trust calculation uses both the service's trust structure and the user trust data to calculate a trust value. These values are the used to make a recommendation. Our user can then leverage these recommendations to decide what services to select in order to compose the presentation service so long as the trust values meets the user's trust policy.

The trust ontology is a rich semantic representation of the previously outlined trust concepts (reliability, credibility, etc) that are made up of properties specific to web services. For example, 'downtime' could be a factor in determining a service's 'reliability'. The trust concepts are linked together by different strength relationships. Figure 3 presents a snippet from the trust ontology, expressed in OWL. It can be seen that 'reliability' is disjoint with 'confidence', related to 'reputation', and has 'reliability properties'.

```
<owl:Class rdf:about="#Reliability">
  <owl:disjointWith rdf:resource="#Confidence"/>
  <owl:equivalentClass>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <owl:Restriction>
          <owl:someValuesFrom rdf:resource="#Reputation"/>
            <owl:onProperty>
              <owl:ObjectProperty rdf:about="#relationship"/>
            </owl:onProperty>
        </owl:Restriction>
        <owl:Restriction>
          <owl:someValuesFrom rdf:resource="#ReliabilityProperies"/>
            <owl:onProperty>
              <owl:ObjectProperty rdf:about="#hasProperties"/>
            </owl:onProperty>
        </owl:Restriction>
…
```

Figure 3: Trust Model Snippet

The user policy document, trust structure, and trust data are also implemented using OWL. Figure 4 illustrates that a services 'reputation' must have a 'high' value.

```
<owl:hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string">high</owl:hasValue>
  <owl:onProperty>
  <owl:DatatypeProperty rdf:about="&trust;reputation"/>
```

Figure 4: Policy Snippet

V. Experience and Analysis

V.1 Ontology Development Tools

The ontology development environment used to create the specified model was the Protégé ontology editor and knowledge acquisition system [xxxiv]. Outside of the Integrated Development Environment the application development framework utilized was Hewlett Packard's Jena2 [xxxv]. It is our experience that Protégé empowers the user to not only create ontologies with greater ease but it also dynamically modifies them as necessary. Protégé's full installation allows the application to natively handle OWL, which was our chosen language for model implementation. Jena2 takes an OWL document and builds a model in memory that can be reasoned about at run-time. It was possible to write Java code that not only built the memory model from file but also allowed us to reason about the model.

We found that the combination of Protégé and Jena2 enables a developer to create an ontological model from specification and reason about that model as required. A significant challenge that we overcame presented itself when we tried to infer new knowledge from the knowledge that was already intrinsic to the ontology.

V.2 Development Issues

The OWL language itself has some weaknesses that created problems when implementing our model. In [xxxvi], Horrocks finds that OWL is a weak language for talking about properties. He cites the lack of a 'composition constructor' as the main reason why OWL can't combine the 'parent' and 'brother' properties to find 'uncle' properties. Horrock suggested an extension to OWL called the OWL Rule Language (xxxv), which is similar to the Semantic Web Rule Language (SWRL) [xxxvii]. Both SWRL and ORL allows these 'uncle' type relationships to be expressed.

In figure 5 the SWRL syntax is presented. The rules that are stated in figure 5 specify that if ‘x1’ ‘hasFather’ ‘x2’ and if ‘x2’ ‘hasBrother’ ‘x3’, then we can infer that ‘x1’ ‘hasUncle’ ‘x3’. Our model required that some similar set of rules be developed so that all the knowledge that we specified could be utilized by the application.

```
<ruleml:Imp>
    <ruleml:body rdf:parseType="Collection">
        <swrl:IndividualPropertyAtom>
            <swrl:propertyPredicate rdf:resource="hasFather"/>
            <swrl:argument1 rdf:resource="#x1" />
            <swrl:argument2 rdf:resource="#x2" />
        </swrl:IndividualPropertyAtom>

        <swrl:IndividualPropertyAtom>
            <swrl:propertyPredicate rdf:resource="hasBrother"/>
            <swrl:argument1 rdf:resource="#x2" />
            <swrl:argument2 rdf:resource="#x3" />
        </swrl:IndividualPropertyAtom>
    </ruleml:body>
<ruleml:head rdf:parseType="Collection">

    <swrl:IndividualPropertyAtom>
        <swrl:propertyPredicate rdf:resource="hasUncle"/>
        <swrl:argument1 rdf:resource="#x1" />
        <swrl:argument2 rdf:resource="#x3" />
    </swrl:IndividualPropertyAtom>
```

Figure 5: Partial SWRL syntax

Both SWRL and ORL are very recent developments and the technologies that allow developers to use SWRL in the manner that we wanted to are at present at a very early stage. Hoolet [xxviii] is such a technology, yet we found that its early stage of development didn’t allow us to use it effectively. Our solution to implementing the ‘uncle’ like rules came through the utilization of RDQL [xxxix] queries. We specified queries that firstly asked ‘is x1’s father x2?’, then asked ‘is x2’s brother x3?’, therefore we could conclude that ‘x1’s uncle is x3!’. This solution worked yet in doing so it moves the knowledge from an ontology-centric view to an application-centric view as the RDQL queries are stored on the application side. It would be possible to shift this knowledge back to an ontology-centric view by encoding the queries in OWL, which the application could import like it does the ontological model.

VI. Conclusions

We have presented the key concepts, related work, and the initial experiment involved in our Ericsson/KDEG trust project. Our experience in developing the project over a three month was presented with an analysis of the issues we encountered.

We have designed and developed an ontology based trust model specific to web services and have shown how it can be used to semantically annotate services with trust information from which trust values can be computed. Throughout the project we encountered and overcame several development issues and succeeded in realizing our goals.

VII. References

[i] McKnight, H.D., Chervany, N.L., 'The Meanings of Trust; Technical Report 94-04, Carlson School of Management, University of Minnesota', 1996.

[ii] Golbeck, J., Hendler, J., Parsia, B. 'Trust Networks on the Semantic Web', 12th International Web Conference (WWW03), Budapest, Hungary, May 2003.

[iii] Shadbolt, N., 'A Matter of Trust', IEEE Intelligent Systems, pp. 2-3 January/February 2002.

[iv] Golbeck, J., Hendler, J., 'Inferring Reputation on the Semantic Web', 13th International Web Conference (WWW2004), New York, NY, USA, May 2004.

[v] Grandison, T., Sloman, M., 'A Survey of Trust in Internet Applications', IEEE Communications Surveys, 3, pp. 2-16, Fourth Quarter 2000.

[vi] Zhang, L. J., Jeckle, M., 'The Next Big Thing: Web Services Collaboration', Proceedings of ICWS-Europe 2003, Springer, LNCS 2853, September 2003, pp 1-10.

[vii] Berners-Lee, T., 'Business Model for the Semantic Web', W3C Semantic Web Activity background, 29th October 2001

[viii] McGuinness, D.L., van Harmelen, F., 'OWL Web Ontology Language Overview', W3C Proposed Recommendation, 15th Dec 2003.

[ix] M. Blaze, J. Feigenbaum, J. Ioannidis, and A. Keromytis. The KeyNote Trust-Management System. Work in Progress, http://www.cis.upenn.edu/~angelos/keynote.html , June 1998.

[x] M. Blaze, J. Feigenbaum, and J. Lacy. Decentralized Trust Management. In Proc. of the 17th Symposium on Security and Privacy, pages 164{173. IEEE Computer Society Press, Los Alamitos, 1996.

[xi] M. Blaze, J. Feigenbaum, and M. Strauss. Compliance Checking in the PolicyMaker Trust-Management System. In Proc. of the Financial Cryptography '98, Lecture Notes in Computer Science, vol.1465, pages 254{274. Springer, Berlin, 1998.

[xii] Keromytis, A., Ioannidis, S., Greenwald, M.B., Smith, J.M., 'The STRONGMAN Architecture', In the *Third DARPA Information Survivability Conference and Exposition (DISCEX III)*, Washington, D.C. April 22-24, 2003.

[xiii] Gottschalk, K., et al, 'Web Services Architecture Overview: The Next Stage of Evolution for E-Business', http://www-106.ibm.com/developerworks/library/w-ovr

[xiv] Ruoyan, Z., Arpinar, B., Aleman-Meza, B., 'Automatic Composition of Semantic Web Services', The 2003 International Conference on Web Services (ICWS'03), June 2003.

[xv] Narayanan, S., McIlraith, S., 'Simulation, Verification and Automated Composition of Web Services', WWW2002, May 7-11, 2002, Honolulu, Hawaii, USA.

[xvi] Srivastava, B., Koehler, J., 'Web Service Composition - Current Solutions and Open Problems.', ICAPS 2003, Workshop on Planning for Web Services 10 June 2003, Trento, Italy.

[xvii] McIlraith, S., Son, T.C., Zeng, H., 'Semantic web services', IEEE Intelligent Systems, 16(2):46-53, March/April 2001.

[xviii] McIlraith, S. and Martin, D., 'Bringing Semantics to Web Services', IEEE Intelligent Systems, 18(1):90--93, January/February, 2003.

[xix] Carey, K., Feeney, K., Lewis, D., 'State of the Art: Policy Techniques for Adaptive Management of Smart Spaces', M-Zones Deliverable 1, June 2003, http://www.m-zones.org/deliverables/D1_1/Policy.pdf

[xx] Sloman, M., Lupu, E., (2002) 'Security and Management Policy Specification', IEEE Network, vol.16 No. 2 pp.10-19, March/April 2002.

[xxi] Chu, Y., Feigenbaum, J., LaMacchia, B., Resnick, P., and Strauss, Ma., 'REFEREE: Trust Management for Web Applications.', The World Wide Web Journal, 1997, 2(3), pp. 127-139.

[xxii] Kagal, L., Paoucci, M., Srinivasan, N., Denker, G., Finin, T., and Sycara, K., 'Authorization and Privacy for Semantic Web Services', AAAI 2004 Spring Symposium on Semantic Web Services, March 22, 2004.

[xxiii] RDFWeb: FOAF: 'the friend of a friend vocabulary', http://rdfweb.org/foaf/

[xxiv] Gil, Y., Ratnakar, V., 'Trusting Information Sources One Citizen at a Time.', Proceedings of the First International Semantic Web Conference (ISWC), Sardinia, Italy, June 2002.

[xxv] Dumbill, E., 'XML Watch: Finding friends with XML and RDF.', IBM Developer Works, http://www-106.ibm.com/developerworks/xml/library/xfoaf.html, June 2002.

[xxvi] Lassila, O., Swick, R.R., 'Resource Description Framework (RDF) Model and Syntax Specification', W3C Recommendation 22nd February 1999.

[xxvii] Levien, R., Aiken, A., 'Attack resistant trust metrics for public key certification.', 7th USENIX Security Symposium, San Antonio, Texas, January 1998.

[xxviii] M. Blaze, J. Feigenbaum, J. Ioannidis, & A. Keromytis. The KeyNote Trust-Management System. Work in Progress, http://www.cis.upenn.edu/~angelos/keynote.html , June 1998.

[xxix] http://trust.mindswap.org/ont/trust.owl

[xxx] Kagal, L., Undercoffer, J., Perich, F., Joshi, A, Finin, T., 'A Security Architecture Based on Trust Management for Pervasive Computing Systems', Proceedings of Grace Hopper Celebration of Women in Computing 2002.

[xxxi] Damianou, N., Dulay, N., Lupu, E. C., and Sloman, M., 'Ponder: A Language for Specifying Security and Management Policies for Distributed Systems (Version 2.3)', Imperial College of Science, Technology and Medicine, Department of Computing, 20 October 2000.

[xxxii] Kagal, L., Finin, T., Joshi, A., 'A Policy Language for a Pervasive Computing Environment', Proceedings of IEEE 4th International Workshop on Policies for Distributed Systems and Networks (POLICY 2003), Lake Como, Italy.

[xxxiii] Bradshaw, J. M., Dutfield, S., Benoit, P., and Woolley, J. D., 'KAoS: Toward an industrial-strength generic agent architecture', In J. M. Bradshaw (Ed.), Software Agents. (pp. 375-418). Cambridge, MA: AAAI Press/The MIT Press.

[xxxiv] Musen, M. A. , Tu, S. W. , Eriksson, H., Gennari, J. H. , and Puerta, A. R., 'PROTEGE-II: An Environment for Reusable Problem-Solving Methods and Domain Ontologies', International Joint Conference on Artificial Intelligence, Chambery, Savoie, France, . 1993.

[xxxv] Hewlett Packard, http://www.hpl.hp.com/semweb/jena.htm.

[xxxvi] Horrocks, I, Patel-Schneider, P.F., 'A proposal for an owl rules language', In *Proc. of the Thirteenth International World Wide Web Conference (WWW 2004)*, pages 723-731, ACM, 2004.

[xxxvii] Horrocks, I ., and Patel-Schneider, P.F., 'A Semantic Web Rule Language Combining OWL and RuleML', April 2004, http://www.daml.org/rules/proposal/.

[xxxviii] University of Manchester, http://owl.man.ac.uk/hoolet/.

[xxxix] Hewlett Packard, http://www.hpl.hp.com/semweb/rdql.htm.