Specifying Conflict of Interest Assertions in WS-Policy with Chinese Wall Security Policy

PATRICK C. K. HUNG[1] AND GUANG-SHA QIU[2]
[1]CSIRO Mathematical and Information Sciences, Canberra ACT, Australia
[2]Faculty of Engineering, University of Western Australia, Perth WA, Australia
Patrick.Hung@csiro.au, cuey@tartarus.uwa.edu.au

A Web service is defined as an autonomous unit of application logic that provides either some business functionality or information to other applications through an Internet connection. Web services are based on a set of XML standards such as Simple Object Access Protocol (SOAP), Universal Description, Discovery and Integration (UDDI) and Web Services Description Language (WSDL). The benefits of adopting Web services over traditional business-to-business applications include faster time to production, convergence of disparate business functionalities, a significant reduction in total cost of development, and easy to deploy business applications for trading partners. However, Web services architectures are built on an insecure, unmonitored and shared environment, which is open to events such as security threats. Security concerns are the major barrier that prevents many business organizations from implementing or employing Web services. This article discusses one of the classical security policies that deal with conflict of interest - the Chinese wall security policy. The article then extends this concept into specifying and implementing conflict of interest assertions in the newly developed WS-Policy. WS-Policy is an XML representation that provides a grammar for expressing Web services policies, to allow service locators to have a common interpretation of security requirements in the matchmaking process.

Additional Key Words and Phrases: Conflict of interest, Chinese wall security policy, WS-Policy, WS-PolicyAttachment, Matchmaking, Delegation, Security policy, Security assertion, Service locators.

1. INTRODUCTION

A Web service is defined as an autonomous unit of application logic that provides either some business functionality or information to other applications through an Internet connection. Web services are based on a set of XML standards such as Simple Object Access Protocol (SOAP), Universal Description, Discovery and Integration (UDDI) and Web Services Description Language (WSDL). Some studies [Holland 2002] show that the Web services market is expected to grow to USD$28 billion in sales in the coming three years. In particular, Grid technologies and infrastructures increase the need for sharing and coordinating the use of Web services for different business processes in a loosely coupled execution environment. A business process contains a set of activities which represent both business tasks and interactions between Web services. Web services applications are growing in popularity. It is believed that early adopters of Web services may include several industries that involve a set of diverse trading partners working closely together in a highly competitive market such as insurance, financial services and high technology industries [Ratnasingam 2002]. Traditional business-to-business applications connect trading partners through a centralized architecture. A major drawback is that setting up an additional connection with another trading partner is costly and time consuming.

In contrast, the benefits of adopting Web services include faster time to production, convergence of disparate business functionalities, a significant reduction in total cost of development and easy to deploy business applications for trading partners [Ratnasingam 2002]. Another difference between traditional business-to-business applications and Web Services is a secure environment versus an exposed environment. Although WSDL and UDDI address the issues of discovery for trading partners (e.g., Web services), they do not discuss any approach to detecting conflicts, dissatisfaction and mistrust among trading partners [Ratnasingam 2002].

In the past few years, business process or workflow proposals relevant to Web services are proliferating in the business and academic world [W3C]. Most of the proposed XML languages are based on WSDL service descriptions with extension elements. For example, the Business Process Execution Language for Web Services (BPEL4WS) [IBM 2002a] is a formal specification of business processes and interaction protocols. BPEL4WS defines an interoperable integration model that facilitates the expansion of automated process integration in an intra-corporate and inter-corporate environment. No matter which specific XML language is used to define a business process by a modeler, each of the activities in a business process must be executed by an appropriate Web service. In this scenario, the role of service locators is to assign an appropriate Web service for each activity. This assignment process is called matchmaking. Furthermore, value-added Web services are required to be enacted by long duration multi-step activities. Thus Web services may also delegate some sub-activities that are decomposed from the assigned activities to other Web services. This assignment process is called delegation [IBM 2002a]. In general, both assignment processes are expected to use the service registry to find the most appropriate Web service to satisfy activities' or even sub-activities' requirements.

2. WEB SERVICES SECURITY

Web services architectures are built on an insecure, unmonitored and shared environment, which is open to events such as security threats. This may result in conflicts since the open architecture of Web services makes it available to many parties, who may have competing interests and goals [Ratnasingam 2002]. For example, a party's commercial secrets may be released to another competing company via the Web services execution. As is the case in many other applications, the information processed in Web services might be commercially sensitive so it is important to protect it from security threats such as disclosure to unauthorized parties. Since security is an essential and integral part of many business processes, Web services have to manage and execute the activities in a secure way. However, the research area of Web services security is challenging as it involves many disciplines, from authentication/encryption to access management/security policies. Security concerns and the lack of security conventions are the major barriers that prevent many business organizations from implementing or employing Web services. An Evans Data Corporation survey of 400 enterprise development managers, conducted in January 2002, showed that 45.5% of the managers regarded security and authentication issues to be the biggest obstacle to Web services implementation [Fontana 2002].

There are also XML languages proposed for describing security assertions in Web services. These XML languages restrict access to Web services to authorized parties only and protect the integrity and confidentiality of messages exchanged in a loosely coupled execution environment. For example, ebXML [Hofreiter et al. 2002] is an XML language used to enable the global use of electronic business information in an interoperable, secure and consistent manner by all parties. Specifically there are two well-known formats for XML-based security tokens: the Security Assertions Markup Language (SAML) [OASIS 2002] and the eXtensible rights Markup Language (XrML) [ContextGuard 2002]. SAML is used to define authentication and authorization decisions in Web services. Web services providers submit SAML tokens to security servers for making security decisions. In addition, a Java-based toolkit called JSAML [NETEGRITY 2001] is developed for supporting SAML in e-business applications. SAML is an XML-based framework for exchanging security credentials in the form of assertions about subjects. Similarly, XrML assists the owners of Web services to specify the rights of authorized users or parties and to identify the terms and conditions under which those rights may be exercised by those authorized users or parties. Lastly, WS-Security [IBM 2002b] describes enhancements to SOAP messaging to provide quality of protection through message integrity, message confidentiality and single message authentication. From another point of view, WS-Security is the messaging language and SAML is the security language. Similar to ebXML, WS-Security mainly focuses on secure communication.

Based on the WS-Security, WS-Policy [IBM 2002b] provides a grammar for expressing Web services policies. The WS-Policy includes a set of general messaging-related assertions defined in WS-PolicyAssertions [IBM 2002b] and a set of security policy assertions related to supporting the WS-Security specification defined in WS-SecurityPolicy [IBM 2002b]. In addition to the WS-Policy, WS-PolicyAttachment [IBM 2002b] defines how to attach these policies to Web services or other subjects such as service locators. WS-Authorization [IBM 2002b] defines how Web services manage authorization data and policies. The eXtensible Access Control Markup Language (XACML) [IBM 2002b] defines the fine-grained authorization and entitlement policies between subjects and resources. WS-Trust [IBM 2002b] defines methods for issuing and exchanging security tokens for establishing the presence of trust relationships. Lastly, WS-SecureConversation [IBM 2002b] defines a security context based on security tokens for secure communication. In conclusion, none of these languages proposes any security assertion for the matchmaking process between activities and Web services as discussed above. This is the major motivation of this article.

3. CONFLICT OF INTEREST AND CHINESE WALL SECURITY POLICY

By convention, security threats are usually thought to come from outsiders. In many cases, however, security problems arise in a well-control environment from authorized insiders. This means that a secure environment must not only ensure that Web services are trusted but must also deal with other security threats such as conflict of interest. Webster's Dictionary defines "conflict of interest" as a conflict between the private interests and the official responsibilities of a person in a position of trust. In fact, the concept of conflict of interest has been studied in other research fields for some time [Coombs and Avrunin 1988]. In bargaining games, conflict of interest is a property of the preferences of the participants and the structure of the situation in which they find themselves. In the example of a loan application process in a financial institution, the loan applicant should not hold a position of authority, such as manager, in the financial institution which approves his own application. There is a concern that the applicant may be likely to influence the decision for the loan approval. This is clearly an illustration of role/user conflict of interest between the self-interest of the applicant and the correct responsibilities of the manager, which is to approve loan applications only if they satisfy the criteria. By convention, if the applicant is a manager or any position of authority in a financial institution, then the applicant's loan application has to be carefully processed by other persons in that financial institution.

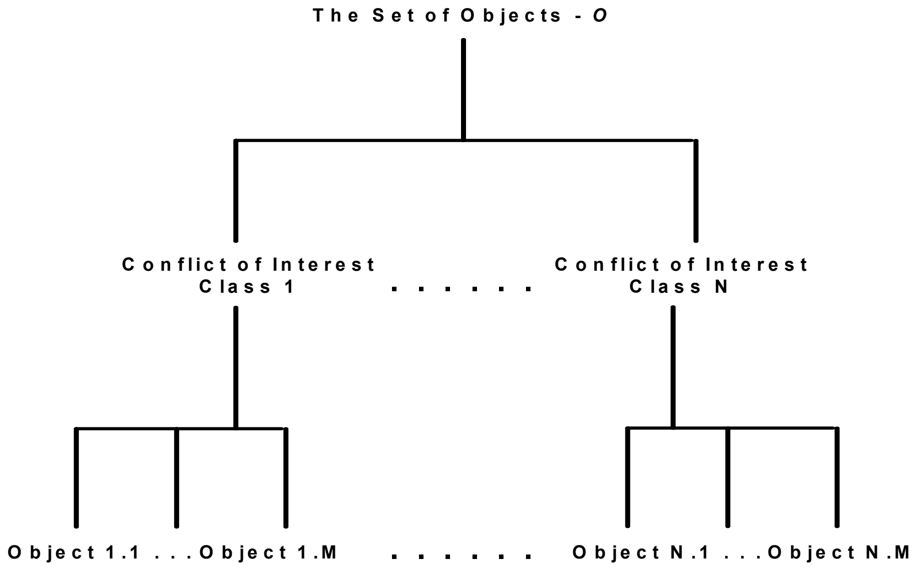

The Set of Objects - O
Conflict of Interest Class 1
......
Conflict of Interest Class N
Object 1.1 . . . Object 1.M
......
Object N.1 . . . Object N.M

Fig. 1. The Conceptual Model of Chinese Wall Security Policy.

A Web service security model must support protocol-independent, declarative security policies that the service locator can enforce, and descriptive security policies attached to the service locator that it can use in order to securely assign activities to Web services. A security policy is a set of rules and practices that specify or regulate how a system or organization provides security services to protected resources. A security assertion is typically scrutinized in the context of security policy [Hondo et al. 2002]. In general, the engineering of a security policy starts with risk analysis and ends with a set of security assertions that is ready for integration into the security architecture of a subject such as a service locator. Risk analysis identifies security threats in a business process and forms a set of security assertions, which refer to rules and practices to regulate how sensitive or activity information is managed and protected within a loosely coupled execution environment. A security policy is often formalized or semi-formalized in a security model that provides a basis for a formal analysis of security properties. One of the classical security policies that deal with conflict of interest is the Chinese wall security policy, which is a real-world security policy in the commercial sector [Brewer and Nash 1989]. The major objective of the Chinese wall security policy is to prevent information flows which cause conflict of interest for individual consultants. The Chinese wall security policy contains a set of access control rules such that no person can ever access data on the wrong side of the wall. The original application of Chinese wall security policy is illustrated as follows [Brewer and Nash 1989].

Organization information is stored in datasets. Initially, each consultant has the potential to access any dataset. Once a consultant has access to a dataset of a particular organization, that consultant is not allowed access to datasets of any other competing organization. Figure 1 illustrates the conceptual model of Chinese wall security policy in. A set of objects *O* is classified into a set of conflict of interest classes (e.g., from 1 to N), and then each class contains a set of competing objects (e.g., from 1 to M), where each object from *O* can only belong to one class. Referring to the example discussed before, the dataset is represented by the set of objects. Each subject (consultant) can only access at most one dataset under each class. However, each subject may access multiple datasets from different class. In the context of a matchmaking process, the set of activities/Web services is also represented by the set of objects/subjects respectively. Then we can classify those conflicting activities into a conflict of interest class. This means that each appropriate Web service can be assigned no more than one activity from each class. In reality, conflict of interest information may be valid only over a certain period of time, so that after some point there is no conflict. For example, some sensitive information is only treated as confidential within a certain period of time. Therefore, there is a need to permit the specification of a session (timeframe) for the conflicting activities.

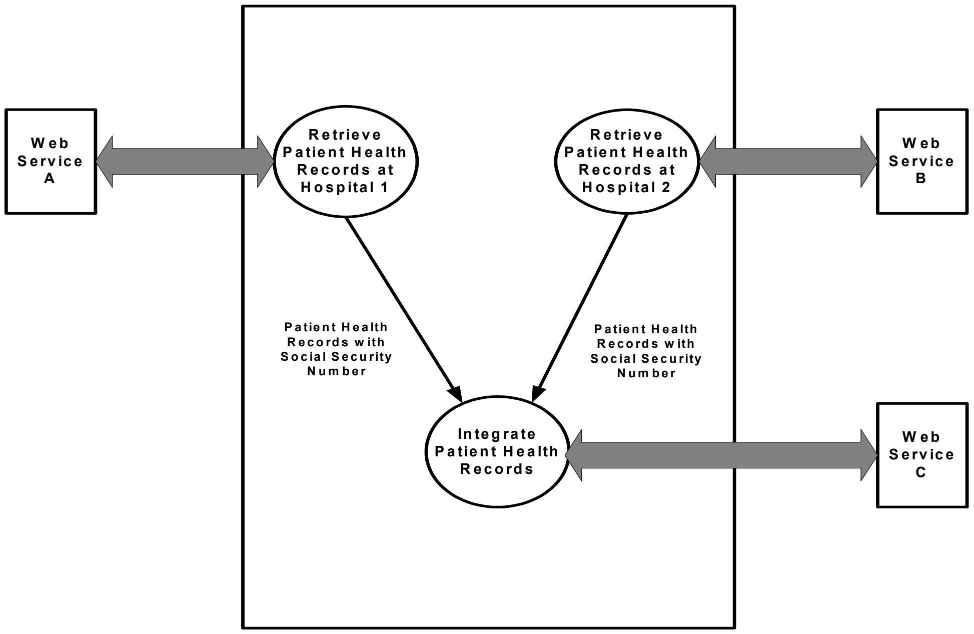
4. SPECIFYING CONFLICT OF INTEREST ASSERTIONS IN WS-POLICY

Service locators are required to ensure that the commercial secrets of clients do not leak via Web services execution. We apply the Chinese wall security policy to deal with this security requirement. Figure 2 shows a business process called "Patient Health Records Integration Process" that includes three activities "Retrieve Patient Health Records at Hospital 1," "Retrieve Patient Health Records at Hospital 2" and "Integrate Patient Health Records." The patient health records often contain sensitive and identifiable information about patients. The sensitive or identifiable attributes from those patient health records should have been protected properly, e.g., encrypting the patient identifiers in the dataset. Assume that there exist two Web services A and B, which can provide the services to retrieve the patient health records at hospital 1 and 2 respectively. Also assume that both Web services, A and B, can provide the services to integrate the patient health records from both datasets by a common key such as Social Security Number. Note that neither Web service A nor B is allowed to release those patient health records to the other to protect patient's privacy. If Web Service A or B can access the aggregate of patient health records (from both hospital 1 and 2), it may be possible for Web service A or B to infer a patient's identity by using data mining techniques (i.e., either patient health records at hospital 1 and 2). Therefore, neither Web service A nor B is allowed to execute the consequent "Integrate Patient Health Records" activity because conflict of interest arises among these three activities. As a result, a third party is required, such as Web service C, to execute the "Integrate Patient Health Records" activity, i.e., (Web service A *Conflicts-With* Web service C) $\wedge$ (Web service B *Conflicts-With* Web service C) $\wedge$ (Web service A *Conflicts-With* Web service B). Referring to the Chinese wall security wall policy discussed before, we can create a conflict of interest class for these three conflicting activities, i.e., *Class1* = {"Retrieve Patient Health Records at Hospital 1," "Retrieve Patient Health Records at Hospital 2," "Integrate Patient Health Records"}. This means that no Web service can execute more than one activity from the *Class1*.

There are two approaches to making conflict of interest associations in the context of a matchmaking process. The security assertions can be made as part of the definition of activities such as the WSEL [Hung 2002]. The other way is to define them independently and associate them through an external binding to the service locators [IBM 2002b]. In this article, we specify the security assertions in the WS-Policy that will allow service locators to have a common interpretation of security requirements in a matchmaking process [Hondo et al. 2002]. As discussed before, WS-Policy is used to specify policy information on a broad range of service requirements, preferences, and capabilities. The WS-Policy is represented by a policy expression that is an XML Infoset representation of one or more policy statements. A policy statement is a group of security assertions. Based on the example discussed before, we propose the corresponding security assertions shown in Figure 3.

Beside those standard Web services utility (wsu) and policy (wsp) namespaces defined at http://schemas.xmlsoap.org/ws/2002/07/utility and http://schemas.xmlsoap.

org/ws/2002/12/policy [IBM 2002b], we assume that there is a schema for a Web services matchmaking process (wsmp) located at http://schemas.xmlwsmp.org/wsmp /2003/01/matchmaking, as well as a schema for the business process of the "Patient Health Records Integration Process" stored at the http://schemas.xmlbp.org/bp/2003 /01/hri. Then, the <wsp: Policy> element (lines 1, 2, 3, 4, 5 and 16) is the top-level container for a policy expression with all the namespaces. In addition, the wsu:id attribute is used to indicate a fragment ID in arbitrary containing elements.


Web Service A
Retrieve Patient Health Records at Hospital 1
Retrieve Patient Health Records at Hospital 2
Web Service B
Patient Health Records with Social Security Number
Patient Health Records with Social Security Number
Integrate Patient Health Records
Web Service C
Fig. 2. Patient Health Records Integration Process.

The primary operator in this policy expression is <wsp:All> (lines 6 and 15), which indicates that all of its child assertions are combined to form a policy statement. The <wsp:All> element indicates that all of the contained security assertions must be met. The wsp:Usage attribute is used to qualify the semantics of the leaf elements as applied to a policy subject such as the service locator. In this example, the value wsp:Required means that the assertion must be applied to the service locator. If the service locator does not meet the criteria expressed in the assertion, a fault or error will occur. The wsp:preference attribute is to specify the preference of this policy statement. The higher the value of the preference is, and the greater the weighting of the expressed preference is.

We propose the security assertions for specifying conflict of interest in the leaf elements (lines 7, 8, 9, 10, 11, 12, 13 and 14). The <wsmp:ConflictOfInterestSession> element (lines 7 and 8) is used to indicate the assertion validity period of the conflict of interest class *Class1* (wsmp:ClassName) from wsmp:Start to wsmp:End. The consequent security assertions <wsmp:ConflictOfInterest> are used to specify those three conflicting activities in the *Class1*:

- wsmp:ActivityName="bp:RetrievePatientHealthRecordsatHospital1" (lines 9 and 10)
- wsmp:ActivityName="bp:RetrievePatientHealthRecordsatHospital2" (lines 11 and 12)
- wsmp:ActivityName="bp:IntegratePatientHealthRecords" (lines 13 and 14)

```
(01) <wsp:Policy wsu:Id="Policy1"
(02)  xmlns:wsu="http://schemas.xmlsoap.org/ws/2002/07/utility"
(03)  xmlns:wsp="http://schemas.xmlsoap.org/ws/2002/12/policy"
(04)  xmlns:wsmp="http://schemas.xmlwsmp.org/wsmp/2003/01/matchmaking"
(05)  xmlns:bp="http://schemas.xmlbp.org/bp/2003/01/hri" >
(06)  <wsp:All wsp:Usage="wsp:Required" wsp:Preference="100">
(07)   <wsmp:ConflictOfInterestSession wsmp:ClassName="Class1"
(08)    wsmp:Start="01/01/2003" wsmp:End="01/01/2004">
(09)   <wsmp:ConflictOfInterest wsmp:ClassName="Class1"
(10)    wsmp:ActivityName="bp:RetrievePatientHealthRecordsatHospital1" />
(11)   <wsmp:ConflictOfInterest wsmp:ClassName="Class1"
(12)    wsmp:ActivityName="bp:RetrievePatientHealthRecordsatHospital2" />
(13)   <wsmp:ConflictOfInterest wsmp:ClassName="Class1"
(14)    wsmp:ActivityName="bp:IntegratePatientHealthRecords" />
(15)  </wsp:All>
(16) </wsp:Policy>
```

Fig. 3. An Example of Conflict of Interest Assertions in WS-Policy.

We assume that the policy expression is stored at http://www.xmlpolicies123.com/wsmp #Policy1. The mechanism for associating policy with one or more subjects (e.g., service locators) is referred to as policy attachment. We now demonstrate how to attach the policy to the service locator. Figure 4 shows a policy expression to be associated with a resource independent of its definition and representation, using a <wsp:PolicyAttachment> element (lines 1 and 12). The <wsp:AppliesTo> element (lines 2 and 9) defines one domain expression by the <wsp:EndpointReference> (lines 3, 4 and 8) element. In this example, we assume that the service locator is located at http://www.xmlservices123.com. Then, the child assertions contain three elements:

- <wsp:ServiceName> (line 5) indicates the name of the service locator.
- <wsp:PortType> (line 6) indicates the type of the service locator.
- <wsp:Address> (line 7) indicates the Uniform Resource Identifier (URI) of the service locator.

```
(01) <wsp:PolicyAttachment>
(02)  <wsp:AppliesTo>
(03)   <wsp:EndpointReference
(04)    xmlns:services="http://www.xmlservices123.com">
(05)    <wsp:ServiceName Name="services:LocatorService" />
(06)    <wsp:PortType Name="services:LocatorPortType" />
(07)    <wsp:Address URI="http://www.xmlservices123.com/servicelocator" />
(08)   </wsp:EndpointReference>
(09)  </wsp:AppliesTo>
(10)   <wsp:PolicyReference
(11)    wsp:URI="http://www.xmlpolicies123.com/wsmp#Policy1" />
(12) </wsp:PolicyAttachment>
```

Fig. 4. An Example of Related WS-Policy Attachment to Figure 3.

The <wsp:PolicyReference> element (line 10 and 11) references the policy expression that is being applied to the service locator. The wsp:URI attribute references a policy using its URI. Note that this policy expression also constrains delegation models at those appropriate Web services. In this example, this means that no Web service in any delegation model can execute two or more sub-activities decomposed from more than two conflicting activities in the *Class1*.

Figure 5 presents a technical framework for supporting the matchmaking process in a loosely coupled Web services execution environment. The Web service "WS-Policy and WS-PolicyAttachment Editor" is used to specify related XML documents for the matchmaking process. The WS-Policy document is submitted to the Web service "WS-Policy Parser" and then the WS-PolicyAttachment document is used to bind the WS-Policy document to a particular entity, i.e., the service locator. The Web service "WS-Policy Parser" classifies different policy assertions from the WS-Policy document into different groups and then distributes each group to the relevant Web services such as the Web service "Conflict of Interest Service (CIRService)" and other services such as authentication and authorization. On the other side, the Web service "Service Locator" interacts with the Web services "Business Process Generator" (e.g., BPEL4WS) and "Service Registry" (e.g., UDDI) for matching each activity from the business process to an appropriate Web service. During the matchmaking process, the "Service Locator" has to consult those services such as the "CIRService." As a result, the "Service Locator" returns an appropriate matchmaking pattern for each business process. A matchmaking pattern is defined as a set of assignments between activities and Web services. Note that all the interactions between Web services are driven by SOAP messages.

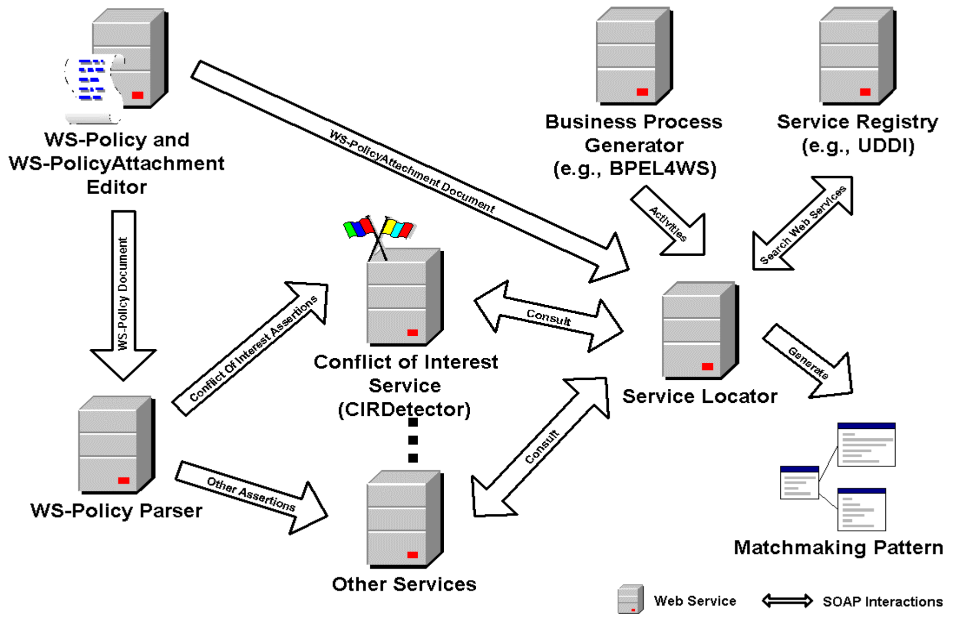

WS-Policy and WS-PolicyAttachment Editor
WS-PolicyAttachment Document
Business Process Generator (e.g., BPEL4WS)
Service Registry (e.g., UDDI)
Activities
Search Web Services
WS-Policy Document
Conflict Of Interest Assertions
Conflict of Interest Service (CIRDetector)
Consult
Service Locator
Generate
WS-Policy Parser
Other Assertions
Other Services
Consult
Matchmaking Pattern
Web Service
SOAP Interactions

Fig. 5. A Technical Framework for Web Services Matchmaking Process.

5. CONCLUSIONS

In comparison to traditional business-to-business applications that connect trading partners through a centralized architecture, Web services have the advantages of faster time to production, convergence of disparate business functionalities, a significant reduction in total cost of development and easy to deploy business applications for trading partners. Despite the numerous benefits attainable after adopting Web services, a major concern is the lack of security conventions after changing from a secure environment to an open and exposed environment. Many security problems do arise in well-controlled environments from authorized parties. This means that security is achieved not only by ensuring that Web services are trusted but also by the clearance of other security threats such as conflict of interest. Therefore, we proposed the Chinese wall security policy in analyzing conflict of interest in the Web services matchmaking process. Further, based on the WS-Policy, we demonstrated how to specify security assertions for preventing conflict of interest in the Web services matchmaking process. We have also implemented the first version of a prototype Web service "CIRService" by using C# on .Net framework.

REFERENCES

BREWER, DAVID F. C. AND MICHAEL J. NASH. 1989. Chinese Wall Security Policy. In *Proceedings of the Symposium on Security and Privacy*, 206-214.

CONTENTGUARD. 2001. eXtensible rights Markup Language (XrML), Version 2.0.

COOMBS, CLYDE H. AND GEORGE S. AVRUNIN. 1988. The Structure of Conflict, Lawrence Erlbaum Associates, Publishers.

FONTANA, J. 2002. Top Web Services Worry: Security. NetworkWorldFusion, January 2002, http://www.nwfusion.com/news/2002/0121webservices.html.

HOFREITER, B., C. HUEMER AND W. KLAS. 2002. ebXML: Status, Research Issues, and Obstacles. In *Proceedings of Twelfth International Workshop on Research Issues in Data Engineering: Engineering E-Commerce/E-Business Systems*, 7-16.

HOLLAND, P. 2002. Building Web Services From Existing Application. In *eAI Journal*, September 2002, 45-47.

HONDO, MARYANN, NATARAJ NAGARATNAM AND ANTHONY NADALIN. 2002. Securing Web Services. In *IBM Systems Journal*, vol. 41, no. 2, 228-241.

HUNG, PATRICK C. K. 2002. Specifying Conflict of Interest in Web Services Endpoint Language (WSEL). In *ACM SIGecom Exchanges*, vol. 3.3, 1-8.

IBM CORPORATION. 2002a. Business Process Execution Language for Web Services (BPEL4WS), Version 1.0.

IBM CORPORATION. 2002b. Security in a Web Services World: A Proposed Architecture and Roadmap, White Paper, Version 1.0. http://www-106.ibm.com/developerworks/library/ws-secroad/

NETEGRITY. 2001. JSAML Toolkit: Netegrity's Java Implementation of the Security Assertions Markup Language (SAML) Specification, Netegrity White Paper.

OASIS. 2002. SAML 1.0 Specification Set: Committee Specifications.

OASIS. 2002. OASIS eXtensible Access Control Markup Language (XACML), OASIS Standard 1.0, 11 December 2002.

RATNASINGAM, P. 2002. The Importance of Technology Trust in Web Services Security. In *Information Management & Computer Security*, vol. 10, no. 5, 255-260.

SUN MICROSYSTEMS. 2002. Web Service Choreography Interface (WSCI), Version 1.0.

UDDI ORGANIZATION. 2002. UDDI Version 3.0, Published Specification.

WORLD WIDE WEB CONSORTIUM (W3C): www.w3c.org