IEEE TASK FORCE ON ELECTRONIC COMMERCE

NEWSLETTER

NO 2, ISSUE 1 2003

Editor: Liang-Jie Zhang

Associate Editor: Zongwei Luo

CONTENT

EXECUTIVE SUMMARY

The purpose of the TFEC Newsletter is to provide dated information on Electronic Commerce activities in a timely manner. Also, we will accept short articles written by TFEC members. The plan is to initially "publish" four issues a year. The editor is soliciting items pertaining to Electronic Commerce for the following categories:

- Announcements about TFEC Activities and Achievements
- Conference Announcements
- Call for Papers - Conferences and Journals
- New Publication Announcements on Electronic Commerce
- Special Issues on Electronic Commerce in Journals
- Workshops, Tutorials and Book Announcements
- New Web Sites and Research Group on Electronic Commerce
- **Short Articles on the latest development of Electronic Commerce (8 pages)**

Please send items to Dr. Liang-Jie Zhang at zhanglj@ieee.org.

In this issue, in addition to TFEC activities, news, there are two academic papers accepted and included. We continue to encourage any kind of contributions that will make this newsletter successfully.

IEEE TASK FORCE ACTIVITIES

The 2003 Workshop on e-Commerce, e-Business, and e-Service (*EEE'03*)

IEEE Task Force on Electronic Commerce is proud of being one of the workshop sponsors!

Source: http://www.iis.sinica.edu.tw/~eee03/

The 2003 Workshop on e-Commerce, e-Business, and e-Service (*EEE'03*) provides a public forum for researchers in academics, IT industry and government agencies to explore and to identify emerging issues on e-technology. *EEE'03* will feature keynote speeches by internationally renowned researchers, interdisciplinary invited talks, focused panel discussions, and tutorials covering important areas of electronic commerce and web services. The main focus of the workshop is on the enabling technologies to facilitate next generation, intelligent e-Business, e-Commerce and e-Government.

This workshop is open to all individuals who are interested in this exciting area. Due to space constraint, however, we may not be able to accommodate everyone who shows interests. Admissions are granted on a first-come-first-served basis, so please register early. All workshop attendees are encouraged to participate in the open discussion. Technical sessions will include invited presentations by distinguished researchers on e-business management, computer and network systems, agent and AI technologies, and data management. No proceedings will be published, but all presentation materials will be made available online and during the workshop.

IEEE Conference on Electronic Commerce (CEC 03)

http://tab.computer.org/tfec/cec03/

The accepted Paper and advance program will be published on the Web site.

IEEE TFEC members are encouraged to attend this conference in June.

NEWS

“Grid Computing can be defined as applying resources from many computers in a network—at the same time—to a single problem; usually a problem that requires a large number of processing cycles or access to large amounts of data.

At its core, Grid Computing enables devices—regardless of their operating characteristics—to be virtually shared, managed and accessed across an enterprise, industry or workgroup. This virtualization of resources places all of the necessary access, data and processing power at the fingertips of those who need to rapidly solve complex business problems, conduct compute-intensive research and data analysis, and engage in real-time.” – www.ibm.com/grid

Girding for Grid

“As financial institutions expanded into derivatives and fixed income over the years, they built complex, distributed-computing environments, networking high-end Unix servers to meet the needs of traders and risk managers in the front and middle offices. But times have changed. “

IBM and Butterfly to run PlayStation®2 games on global grid

“Butterfly Grid Will support massive numbers of video gamers online
Hong Kong, February 28, 2003 -- IBM and Grid pioneer Butterfly.net announced that Butterfly has signed a PlayStation®2 Tools and Middleware agreement with Sony Computer Entertainment Inc. to provide technology and networking services for the PlayStation®2 computer entertainment system. As part of the agreement, IBM and Butterfly activated a first-of-its-kind computing grid that allows online game developers to take advantage of the advanced capabilities of PlayStation®2 by more efficiently provisioning resources to meet the demands of console gamers.”

IBM, United devices and Accelrys aid U.S. department of defense in search for smallpox cure

“Smallpox Research Grid Project to link more than two million computers

ARMONK, N.Y., AUSTIN, TEXAS, and SAN DIEGO, CA, February 5, 2003— IBM, United Devices and Accelrys today announced a project supporting a global research effort that is focused on the development of new drugs that could potentially combat the smallpox virus post infection. The project will be powered through a massive computing "Grid" that will enable millions of computer owners worldwide to contribute idle computing resources with the goal of developing a wide collection of potential anti-smallpox drugs. ”

IBM drives Grid Computing for commercial business with ten new Grid offerings

“Grid Results At Financial Services Firm Reduces Processing Time To Seconds

ARMONK, NY, January 27, 2003— IBM is driving the benefits of Grid computing beyond its academic and research roots and into business enterprises with the introduction today of ten Grid offerings targeting key industries – aerospace, automotive, financial markets, government and life sciences. “

PAPER

In this issue, we are proud to present two academic papers. We continue to encourage our IEEE TFEC members and friends to send your submissions to us.

From State Charts Diagrams to Service Charts Diagrams

ZAKARIA MAAMAR, BOUALEM BENATALLAH

The growth of the Internet and the Web is revolutionizing the way businesses interact with their partners and customers. Indeed, users are becoming more and more demanding on businesses to provide them with relevant and up-to-date information. An increasing number of users are visiting the Web for various reasons: checking weather forecast, getting the latest stock quotation, etc. **Service-based applications [3]** are gaining a considerable momentum as a paradigm for effective discovery and delivery of services that fulfill users' needs. By service (also called Web service or e-service), we mean a semantically well-defined abstraction that allows users to access functionalities offered by Web applications.

Specifying Security Assertions in Web Services Endpoint Properties

PATRICK C. K. HUNG

A Web service is 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). A business process contains a set of activities, and service locators assign an appropriate Web service for each activity. This assignment process is called matchmaking. Further, 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. In addition to the services descriptions in WSDL, Web services are also described through endpoint properties that may include elements such as time thresholds, quality of services, cost of services and legal aspects. Like many other applications, security issues are also important endpoint properties of Web services. Recently IBM proposed a new XML language called Web Services Endpoint Language (WSEL) for describing endpoint properties of Web services. A WSEL document for an activity describes the expected endpoint properties that a Web service should have for executing the activity. For illustration, this paper extends the WSEL to specify the Conflict of Interest (CIR) activity requirements for the processes of matchmaking and delegation as security assertions in Web services endpoint properties.

FROM STATE CHART DIAGRAMS TO SERVICE CHART DIAGRAMS

Zakaria Maamar
College of Information Systems
Zayed University
PO Box 19282, Dubai, UAE
zakaria.maamar@zu.ac.ae

Boualem Benatallah
School of Computer Science & Engineering
The University of New South Wales
Sydney NSW 2052, Australia
boualem@cse.unsw.edu.au

1 INTRODUCTION

The growth of the Internet and the Web is revolutionizing the way businesses interact with their partners and customers. Indeed, users are becoming more and more demanding on businesses to provide them with relevant and up-to-date information. An increasing number of users are visiting the Web for various reasons: checking weather forecast, getting the latest stock quotation, etc. **Service-based applications [3]** are gaining a considerable momentum as a paradigm for effective discovery and delivery of services that fulfill users' needs. By service (also called Web service or e-service), we mean a semantically well-defined abstraction that allows users to access functionalities offered by Web applications.

To exemplify service-based applications, consider the following scenario. John is planning for his summer vacation; he wants to book a domestic flight, as well as an accommodation; he, also, wants to find some attractions for visit; he would like to rent a car if the major attraction is far from the booked accommodation. To be able to satisfy the request of John, multiple services are needed: **flight reservation, hotel booking, attraction, searching, and car rental**. All these services have to be connected to each other according to a certain flow of control. First, flight reservation is completed. Then, hotel booking and attraction searching are triggered.

Current Web service approaches assume a passive mode of interactions where services are explicitly invoked through operations defined in their interfaces. A passive approach is limited when services are autonomous and may have to take part to transient partnerships. A transient partnership does not assume any *a priori* arrangement. A service has to dynamically discover the right partners to team up with in the preparation and provisioning of a high-level business process. To facilitate the establishment of on-demand and real-time alliances among services, it is necessary to cater for the creation of **dynamic** and **transient** interactions among services.

In this paper, we propose service chart diagrams as a means to model services that have the ability to autonomously engage in conversations with other services. Service chart diagrams are viewed as an extension to the traditional state chart diagrams of UML. More precisely, the use of service chart diagrams enables to state **who** provides the service (**organization perspective**), with **whom** the service engages in conversations (**flow perspective**), **what** the service contributes (**information perspective**), and **where** the service execute its operations (**location perspective**).

Section 2 presents an overview of services and explains state chart diagrams. Section 3 introduces service chart diagrams and explains their use with running examples. Section 4 explains the role of conversations in composing services. Section 5 concludes the paper.

2 BACKGROUND ON SERVICES

2.1 CONCEPTS AND DEFINITIONS

A Web service is a Web-accessible application that can be automatically discovered and invoked by other applications (and humans). We adopt a definition which considers an application as a Web service if it is [2]: (i) independent as much as possible from specific platforms and computing paradigms; (ii) developed mainly for inter-organizational situations rather than for intra-organizational situations; and (iii) easily composable (i.e., its composition with other Web services does not require the development of complex adapters). In general, by an identifier (e.g., URI), a set of attributes, and a set of operations specify a Web service. The attributes of a service provide information that is useful to the service's potential consumers.

Maamar et al. introduce the concept of **M-services** (M for Mobile) and suggest two definitions [6]. The "weak" definition is to trigger remotely a Web service from a mobile device for execution. In that case, the Web service is an M-service. The "strong" definition is to transfer wirelessly a Web service from its hosting site to a mobile device where its execution takes place. In that case, the Web service is an M-service that meets the following requirements [6]: 1) transportable through wireless networks; 2) composable with other M-services; 3) adaptable according to the computing features of mobile devices; and finally 4) runnable on mobile devices. In this paper, we focus on the M-services that comply with the "strong" definition. Indeed, we are mainly interested in devising a new generation of Web services to be provisioned to users of mobile applications.

In the rest of this paper, we use the term E-service to designate a conventional Web service (which is not an M-service). We also use the term **composite service** to denote the list of component services, whether E-services or M-services, that are involved in a composition.

2.2 STATE CHARTS FOR SERVICE COMPOSITION

The state chart formalism has been integrated into UML [7], as the foundation of many process-modeling constructs. Succinctly, a state chart is made up of states and transitions. Transitions are labeled by ECA (Event Condition Action) rules. The occurrence of an event fires a transition if (1) the machine is in the source state of the transition, (2) the type of the event occurrence matches the event description attached to the transition, and (3) the condition of the transition holds. When a transition fires, its action part is executed and its target state is entered. The event, condition, and action parts of a transition are all optional.

3 SERVICE CHART DIAGRAMS

3.1 CONCEPTS AND DEFINITIONS

Service chart diagrams are based on UML state chart diagrams. This time, the focus is on the context surrounding the execution of a service rather than on the states that a service takes. Services are represented from five perspectives. Besides the state perspective that includes the states of a service, the flow perspective corresponds to the execution chronology of the connected services. Here, the flow is conversation-based. The organization perspective identifies the business that supplies a service. The information perspective identifies the data that are exchanged between services. These data are identified during conversations and packaged into XML documents. A service that completes its execution may have to leave certain data to the next services that are due

for execution, so they could resume pending operations. Finally, the location perspective identifies the current site of a service. A service can be in one of the two sites: 1) business site waiting to be selected and inserted into a composite service; or 2) execution site under performance. According to Section 2.1, an execution site corresponds to a business site for an E-service or client site for an M-service.

A service chart diagram enhances a state chart diagram with details obtained from the above-cited perspectives (Fig. 1). Therefore, the service chart diagram of a composite service consists of connecting the service chart diagrams of all the services that constitute that composite service. Table 1 summarizes the three layers of a service chart diagram. Interesting is Layer 2; it contains the states that a service takes (Section 2.2). These states constitute themselves a state chart diagram that is wrapped in the four-other perspectives. It should be noted that the states of layer 2 integrate both normal and ad-hoc operating of a service. Ad-hoc operating corresponds to the exceptional cases that may occur (e.g., execution failure). Thus, back-up states or extra services can be requested to handle the exceptional cases.

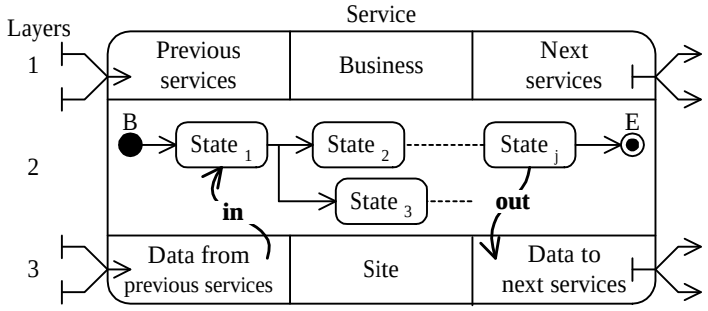

Figure 1: Representation of a service chart diagram

Table 1: Layers of a service chart diagram

Layer	Field	Perspective
1	Previous services Next services	Flow (composite service level)
	Business	Organization
2	States	State (service level)
3	Data from previous services Data for next services	Information
	Site	Location

In Table 1, next services field represents the list of services that are due for execution after a service completes its execution. This list is an expression that combines services over logical operators (**AND**, **OR**, and **NOT**). For instance, services that are connected with an AND operator have to be triggered in a concurrent way. A similar description applies to **OR** and **XOR** operators. Each service that appears in next services field is also annotated with the following elements:

1. The protocol that enables the invocation of the service. **SOAP** over **HTTP** is among the protocols that can be used [8].
2. The conditions to check before the service is invoked. The elements of a condition are obtained from the states of the service that is under execution.

3.2 APPLICATION OF SERVICE CHART DIAGRAMS

According to the location perspective, a service (i.e., an instance) can be in one of the following two sites: business site or execution site. Both types of site influence the content (in term of states)

and shape (in term of chronology) of a service chart diagram. In what follows, the conceptual description of the service chart diagram of Fig. 1 is applied according to the features of each site. Flight reservation service is used for illustration purposes.

A. BUSINESS SITE

Flight reservation service takes stand by state waiting, first to be selected among multiple services by the composition process and then, connected to other services. Fig. 2 is the service chart diagram of the service in business site. In this diagram, certain fields of Table 2 (e.g., business, next services, and site) are filled with values. In addition, the service takes stand by, preparation, and transfer states. Different activities are undertaken within each state. Transfer state only applies to M-services. Indeed, the execution of an M-service takes place in a different site to the business site. Flight reservation service is followed by two services: hotel booking and attraction searching. Both services are triggered in case a flight reservation is confirmed.

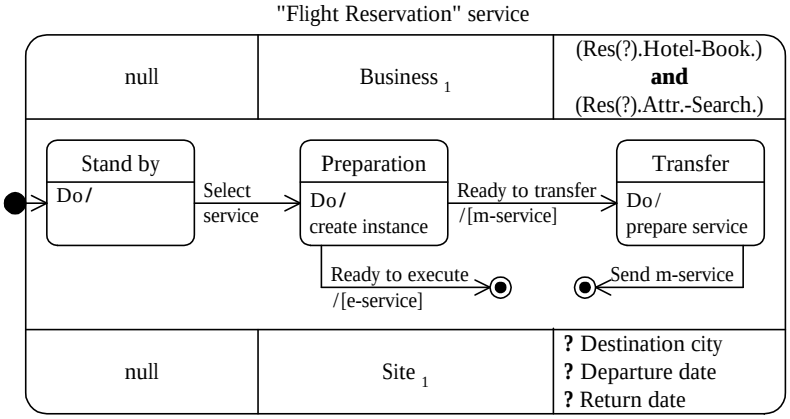

Figure 2: Service chart diagram - Business site

Table 2: State chart diagram details - Business site

Field	Value
Previous services	Null
Next services	(Reservation(?).Hotel Booking) **and** (Reservation(?).AttractionSearching)
Business	$Business_1$ (offers the service)
States	Stand by, Preparation, /Transfer
Data from previous services	Null
Data for next services	?Destination_city, ?Departure_date, ?Return_date
Site	$Site_1$ (where the service is now located)

B. EXECUTION SITE

At this time, the service is due for execution. This execution occurs either in the business site for an E-service or client site for an M-service. Fig. 3 is the service chart diagram of flight reservation service in execution site. Preparation state only applies to M-services; they need to be checked and installed upon arrival from the business site to the site of user which is a mobile host. Table 3 illustrates how the fields of Table 1 are instantiated according to execution site. After terminating the execution of flight reservation service, the relevant information such as date of departure and date of return are obtained and then, supplied to the next services.

Table 3: State chart diagram details - Execution site

Field	*Value*
Previous services	Null
Next services	(Reservation(y).Hotel Booking)/ $Pro._{HB}$ **and** (Reservation(y).AttractionSearching)/$Pro._{AS}$
Business	$Business_1$ (offers the service)
States	/Preparation, /Transfer
Data from previous services	Null
Data for next services	Value(Des_city, Dep_date, Ret_date)
Site	$Site_1$✍$Site_{user}$

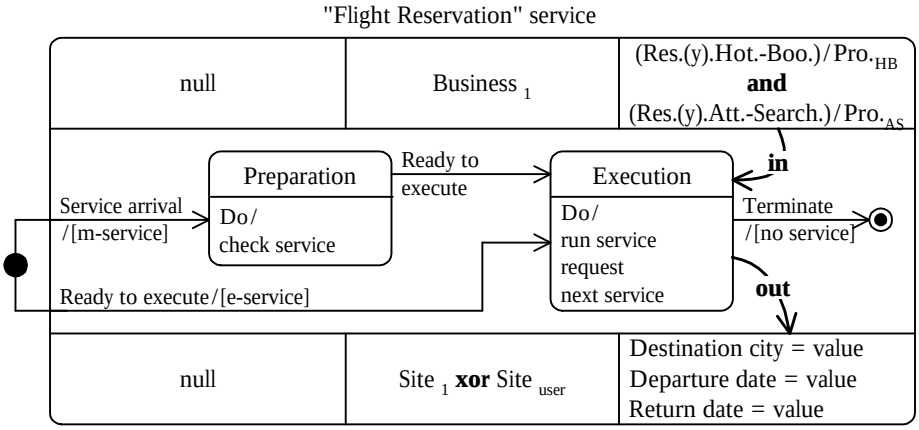

Figure 3: Service chart diagram - Execution site

Note: In Table 3, $Protocol_{HB}$ and $Protocol_{AS}$ are respectively the protocols that trigger Hotel Booking service and Attraction Searching service.

4 CONVERSATION-DRIVEN SERVICE COMPOSITION

In Section 3.2, we pointed out that a service is initially in a selection stage (i.e., business site) and then, enters an execution stage (i.e., execution site). We advocate that services must be able to talk to each other before they decide to join a composite service, what states they take according to the outcome of conversations, and what activities they perform within these states. Conversations are based on a Conversation Language (CL) and are of different types (e.g., representatives, directives, and commissives) [4]. When services engage conversations, they need *a priori* to agree on the exchange protocol to communicate with each other.

We aim at using conversations to leverage services into autonomous components that will be able to make independent decisions [5]. This aids in building composite services at runtime instead of design time. What is interesting to note is the concurrency that exists between the selection and execution stages of a service in an execution site. When a service is under execution, it has at the same time to initiate conversations with the services that are due for execution after approval (next services field). The purpose of these conversations is twofold: invite the services to join the composite service and make sure that the services are ready for execution after they agreed on joining the composite service. Since service chart diagrams of Fig. 2 and Fig. 3 do not include conversation states, we deemed

appropriate to complete these diagrams with the missing states.

A. BUSINESS SITE

Fig. 4 illustrates a light version (i.e., with no perspectives) of the new service chart diagram in business site after introducing the conversation state. The main difference with the service chart diagram of Fig. 2 is that now a service can either accept or reject joining a composite service. Without conversations, it was guaranteed that a service would take part to the composite service. A service can turn down an invitation to join a process of composing services for various reasons; e.g., the maximum number of the instances that can be deployed at the same time of that service has been reached.

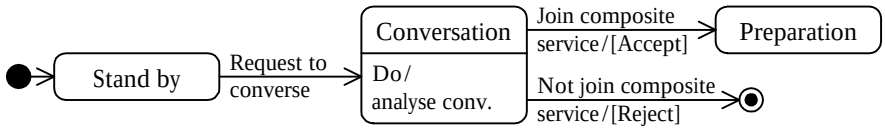

Figure 4: Updated service chart diagram - Business site

B. EXECUTION SITE

Fig. 5 illustrates a light version of the new state chart diagram in execution site after introducing the conversation state. While a service is being executed, it engages conversations with the next services that are due for execution. It should be noted that execution and conversation states are concurrent.

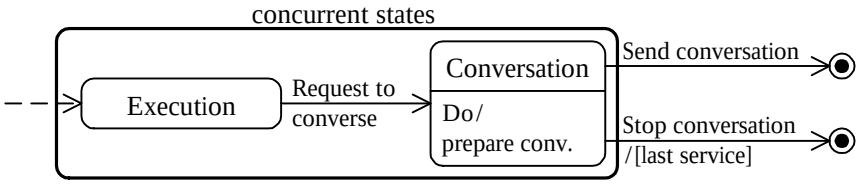

Figure 5: Updated service chart diagram - Execution site

Fig. 6 is a conversation-based interaction diagram that is established between service chart diagrams of two services. In the same Fig., rounded rectangles correspond to states while italic sentences correspond to conversations. We assume the existence of two services: $service_i$ and $service_j$. $Service_i$ is under execution (Fig. 5). While, $service_j$ is under selection, which means due for execution (Fig. 4). Initially, $service_i$ takes two concurrent states; execution state where certain activities are carried out and conversation state where certain activities to select the next services are carried out, too. We mainly focus on the conversation state of $service_i$. Numbers in Fig. 6 correspond to the chronology of conversations.

We recall that services make decisions regarding their intention to join the development of a composite service ($service_{1, .., i, j, ...n}$). In Fig. 6, the first conversation consists of sending a request from $service_i$ to $service_j$ to join the composite service (1). This composite service is decomposed into three segments. The first segment corresponds to the services that have completed their execution ($service_{1, ..., i-1}$). The second segment corresponds to the service that is currently under execution ($service_i$). Finally, the third segment corresponds the services that are under preparation ($service_{j, ..., n}$). $Service_j$ is in stand by state, waiting to receive invitations of joining a composite service. When it receives an invitation, $service_j$ enters the assessment state. Within that state, $service_j$ considers its

constraints and makes a decision to: a) decline the invitation, b) delay its making decision, or c) accept the invitation.

- Case a) In case $service_j$ declines the invitation (2.1), a conversation message is sent from $service_j$ to $service_i$ for notification. Thus $service_i$ enters again the conversation state, asking for another $service_{k\ k?j}$ to join the composite service. It is assumed that there will always be a positive response to the request of joining a composite service.
- Case b) In case $service_j$ cannot make a decision before the deadline of response that $service_i$ has set, $service_j$ requests to extend the deadline (2.2). $Service_i$ has two alternatives: 1.) Refuse the request of $service_j$; this means $service_i$ has to look again for another service (Case a.), or 2.) Accept the request of $service_j$; this means $service_j$ will get informed about the acceptance of $service_i$. For alternative 2, $service_j$ has been offered more time to make a decision. $Service_j$ may request to extend the deadline for several reasons, e.g., it cannot commit additional instances of $service_j$ until other instances complete their execution.
- Case c) In case $service_j$ accepts to join the composite service it notifies its acceptance to $service_i$ (2.3). At the same time, $service_j$ enters the preparation state to get itself ready for execution. It is noted in Fig. 6 that Accept to join/[yes] link between assessment and preparation states of $service_j$ plays two roles: conversation message to notify $service_i$ and transition to enter the preparation state.

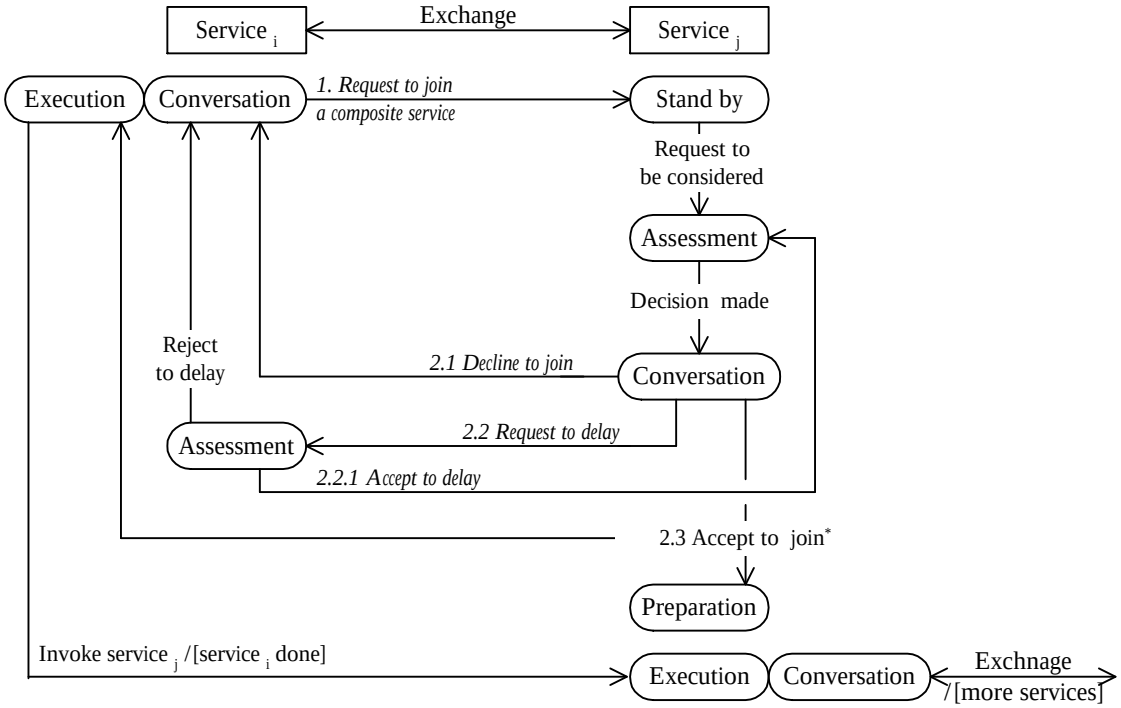

Figure 6: Conversation-based interaction diagram

When $service_i$ finishes its execution, it invokes $service_j$ according to the agreement that was established (Case c.). Therefore, $service_j$ enters the execution state and at the same time, initiates conversations with the next services. $Service_j$ adopts the aforementioned approach.

5. CONCLUSION

In this paper, we presented an approach for designing composite services. Service chart diagrams constitute the backbone of the approach by leveraging the traditional state chart diagrams of UML.

Additional elements are added to state chart diagrams, such as the organization that offers a service and the execution location of a service. Before connecting services together, they engage in conversations to decide if they join a composite service or not. Conversations enable to raise services to the level of autonomous components. As stated in [1], the services that are capable of engaging intelligent interactions would be able to discover and negotiate with each other, mediate on behalf of their users, and compose themselves into more complex services.

One of the main issues that need to be dealt with during conversations is scalability. If a service is requested by a great number of services, a bottleneck situation may happen. In fact, the requested service has to engage conversations with each service, which could definitely take time and require computing resources.

REFERENCES

[1] S. Akhil, M. Vijay, S. Mehmet, L. Li Jie, and C. Fabio. Automated SAL Monitoring for Web Services. *Technical Report HPL-2002-191*, HP Laboratories, Palo Alto, California, USA, 2002.

[2] B. Benatallah, M. Dumas, M. Sheng, and A. Ngu. Declarative Composition and Peer-to-Peer Provisioning of Web Services. In *Proceedings of The 18th IEEE International Conference on Data Engineering (ICDE'2002)*, San Jose, California, 2002.

[3] F. Casati, M.-C. Shan, and D. Georgakopoulos (Editors). Special Issues on E-services. VLDB Journal, 24(1), 2001.

[4] B. Chaib-draa and F. Dignum. Trends in Agent Communication Language. *Computational Intelligence*, 2002.

[5] M. Huhns. Agents as Web Services. *IEEE Internet Computing*, 6(4), July/August 2002.

[6] Z. Maamar, W. Mansoor, and Q. H. Mahmoud. Software Agents to Support Mobile Services. In *Proceedings of the First International Joint Conference on Autonomous Agents & Multi-Agent Systems (AAMAS'2002)*, Bologna, Italy, 2002.

[7] J. Rumbaugh, I. Jacobson, and G. Booch. *The Unified Modeling Language Reference Manual*. Addison-Wesley, 1999.

[8] Simple Object Access Protocol (SOAP). http://www.w3.org/TR/SOAP/, Visited July 2002.

SPECIFYING SECURITY ASSERTIONS IN WEB SERVICES ENDPOINT PROPERTIES

PATRICK C. K. HUNG
CSIRO Mathematical and Information Sciences
GPO Box 664, Canberra, ACT 2601, Australia
Patrick.Hung@csiro.au

A Web service is 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). A business process contains a set of activities, and service locators assign an appropriate Web service for each activity. This assignment process is called matchmaking. Further, 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. In addition to the services descriptions in WSDL, Web services are also described through endpoint properties that may include elements such as time thresholds, quality of services, cost of services and legal aspects. Like many other applications, security issues are also important endpoint properties of Web services. Recently IBM proposed a new XML language called Web Services Endpoint Language (WSEL) for describing endpoint properties of Web services. A WSEL document for an activity describes the expected endpoint properties that a Web service should have for executing the activity. For illustration, this paper extends the WSEL to specify the Conflict of Interest (CIR) activity requirements for the processes of matchmaking and delegation as security assertions in Web services endpoint properties.

Additional Key Words and Phrases: Matchmaking, Delegation, Security Assertions, Web Services Endpoint Properties, Conflict of Interest.

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). A business process contains a set of activities, and activities represent both business tasks and interactions between Web services providers. In this environment, service locators assign an appropriate Web service for each activity. This assignment process is called matchmaking. In particular, 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. In general, both binding processes are expected to use the service directory (i.e., UDDI) to find the most appropriate Web service that can provide the services to satisfy activities' or even sub-activities' requirements.

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. The major barrier that prevents many business organizations from implementing Web services is the security concerns [Yang 2002]. In particular, this paper mainly focuses on tackling the security threat of unauthorized disclosure of information in Web services to unauthorized conflicting parties [Varlamov 2002]. In addition to the services descriptions in WSDL, Web services are also described through endpoint properties that may include different elements such as time thresholds, quality of services, cost of services and legal aspects. Like in many other applications, security issues are definitely important endpoint properties of Web services. Recently IBM proposed a new XML language called Web Services Endpoint Language (WSEL) for describing endpoint properties of Web services. A WSEL document for an activity describes the expected endpoint properties that a Web service should have for executing the activity. For illustration, this paper extends the WSEL to specify the Conflict of Interest (CIR) activity requirements for the processes of matchmaking and delegation as security assertions in Web services endpoint properties. The remainder of this paper is organized as follows: Section 2 discusses the related work in the literature. Next, Section 3 presents the security assertions for CIR in Web services endpoint properties. Then, Section 4 describes the conclusions and future research.

2. RELATED WORK

Recently workflow or business process proposals relevant to Web services are proliferating in the industry and academic world [W3C]. All of the proposed XML languages are based on WSDL service description with extension elements. For example, Thatte [2001] describes XLANG as a notation for the specification of message exchange behaviors (i.e., interactions) among participating Web services in business processes. XLANG describes the behavior of the services as a part of a business process. In XLANG, the behavior of each Web service is specified independently, and the interactions between Web services is only through message exchanges expressed as operations in WSDL. Next, the Web Service Choreography Interface (WSCI) [SUN 2002] describes the flow of messages exchanged by a Web service participating in interactions with other Web services. WSCI describes the dynamic interface of the Web service participating in a given message exchange by means of reusing the operations defined for a static interface. Similarly, the Business Process Execution Language for Web Services (BPEL4WS) [IBM 2002] 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. In addition, BPEL4WS highlights the necessary notions of service policies, endpoint properties, and message contexts. Though BPEL4WS recommends that business process implementations use Web Services Security (WS-Security) [IBM 2002] for secure messaging, none of these languages discussed above provides any security assertion for Web services endpoint properties.

Web Services Flow Language (WSFL) [Leymann 2001] is also an XML language for the description of Web services interactions. WSFL specifies the appropriate usage pattern of a collection of Web services in order to achieve a particular business goal, and WSFL also specifies the interaction pattern of a collection of Web services. In addition to WSFL, Leymann [2001] proposes an XML language called Web Services Endpoint Language (WSEL) for describing Web services endpoint properties, where it matches the expectations from WSFL to the promises from WSDL. Leymann [2001] mentions the importance of security assertions in WSEL as a future work and the current version of WSEL provides a provision for security assertions. Thus the proposed security assertions in this paper will be demonstrated in the context of WSEL.

On the other hand, there are some XML languages proposed for describing security assertions. For example, ebXML [Hofreiter et al. 2002] is a XML language to enable the global use of electronic business information in an interoperable, secure and consistent manner by all parties. Next, OASIS proposes an XML language called Security Assertions Markup Language (SAML) [OASIS 2002] for making authentication and authorization decisions at Web services. Web services providers submit SAML to security servers for requesting authorization decisions. In addition, a Java-based toolkit called JSAML [Netegrity 2001] is developed for supporting SAML in e-business applications. SAML considers authentication and authorization in Web Services. Lastly, Web Services Security (WS-Security) [IBM 2002] describes enhancements to SOAP messaging to provide quality of protection through message integrity, message confidentiality, and single message authentication. Similar to ebXML, WS-Security mainly focuses on secure communication. In a word, most of them focus on the authorization and authentication in a secure communication environment. None of these works considers any security assertion for Web service endpoint properties.

3. SECURITY ASSERTIONS IN WEB SERVICES ENDPOINT PROPERTIES

Figure 1 shows an example of matchmaking model, delegation and flow model for Web services interacted activities. The “Preventive Cancer Study Process” (the flow model) is to investigate the demographic and geographical factors of the people (i.e., smokers) who died of cancer. In the flow model, an activity is represented as a circle. The flow model is used to describe and depict the execution sequence of the activities as specification of the flow of control and data between Web services. The “Retrieve Census Data” and “Retrieve Health Data” are two activities to retrieve

relevant data from different outside sources. The "Retrieve Census Data" activity is assigned to a Web service at "National Bureau of Statistics," and the "Retrieve Health Data" activity is assigned to a Web service at "National Health Center." In particular, the Web service at "National Health Center" (within a delegation model) involves a hierarchical composition of Web services. The delegation model provides a description of how the composed Web services interact with each other as links between operations of the Web services' interfaces. In this scenario, the Web services provider at any point may not be the ultimate destination for the assigned activities. In fact, a Web services provider may also act as a Web services requestor by decomposing an assigned activity into a set of sub-activities, sending sub-activities to multiple Web services for execution, and then gathering the results from the Web services to build an aggregated result. Referring to Figure 1, the Web service at "National Health Center" has to retrieve the relevant data from each state (e.g., from *1* to *N*). Next, each Web service at health data centers in each state is composed of a set of Web services at different hospitals (e.g., from *1* to *M*) of that state. The aggregated result is composed of the partial results from all of these Web services involved in the delegation model. Once the data from both activities (i.e., "Retrieve Census Data" and "Retrieve Health Data") are received, the consequent "Data Analysis" activity is executed by a Web service at "National Center of Health Statistics." Then, the Web service generates a set of customized tables. Finally the "Publish Reports" activity is executed by a Web service at the "National Media Press." Overall, all these assignment processes are assembled as a matchmaking model.

Modeling security enforcement is an issue of paramount importance in many application domains such as office automation, planning, medical diagnosis, and manufacturing systems that need interactions between humans and systems, and also among humans. For example, Web services requestors do not want their confidential information to be released to any unauthorized party. Therefore, Web services requestors will use the security elements of a service description to find service endpoints that meet their policy and security requirements. Based on the prior research, here are four basic assumptions for the security properties at the Web service endpoints:

A1: the Web service is trusted as a means of injecting multilevel security into applications such as command, control, and intelligence systems [Hinke 1989].

A2: the communication channels along the Web services requestors and providers are point-to-point secure against security threats.

A3: the activity is executed in a secure manner by the Web service.

A4: the information at the Web service is protected against security threats.

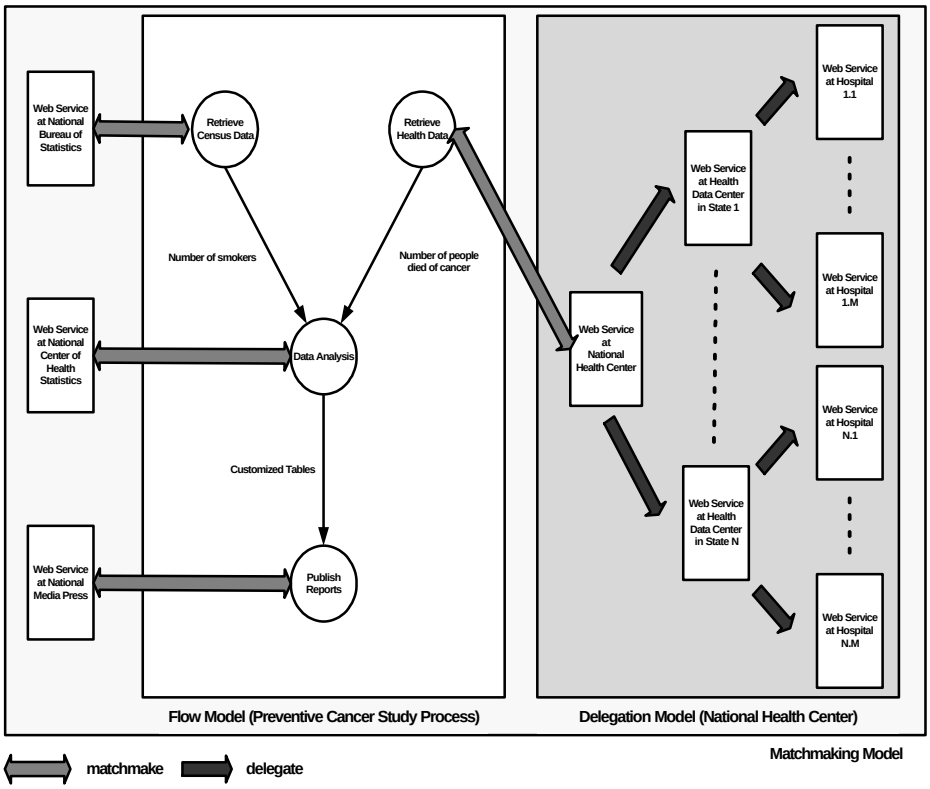

Figure 1. An Example of Matchmaking, Delegation and Flow Model.

In a delegation model, the intermediary Web services providers may also invoke other Web services. As a result, the final Web services providers accept delegation from other Web services providers and the final Web services providers make authorization decisions. The security properties at the endpoints in a delegation model are inherited along the structure of delegation. Referring to Figure 1, the security properties at the Web service at "Health Data Center" in state *1* to *N* are inherited from the Web service at "National Health Data Center" and so on. In addition to those four assumptions (i.e., *A1*, *A2*, *A3* and *A4*), this paper proposes two additional assumptions in a delegation model:

A5: the Web services requestors are eligible to know whether delegation is performed by the Web services being used.

A6: the Web service provides delegation on behalf of requestors (i.e., it passes along the requestors' identity) if and only if it is authorized.

Without doubt, those assumptions are not adequate to ensure a secure execution at the endpoints. Though some Web services are capable to execute certain activities, there may have certain constraints that prohibit them executing a particular activity in a particular situation. To understand this work, this paper develops an abstract model for a business process as follows. A business process is represented into a flow model (FM) that contains a partially ordered set of activities (A) that is coordinated by a set of data/control flows. The order of activity execution is orchestrated by matching the input and output flow(s) of each activity. Each activity represents a piece of work (i.e., a sequence of operations) that needs to be done by a Web service. Each Web service may have a delegation model that contains a set of Web services (WS) in a hierarchical or peer-to-peer structure. In a loosely coupled Web services execution environment, let sets of activities (A) and Web services (WS), and a flow model (FM), respectively, be:

? A = $\{a_1, a_2, ?\ , a_m\}$ is the set of *m* activities.

? WS = {ws_1, ws_2, ? , ws_n} is the set of *n* Web services.

? FM = {a_1, a_2, ? , a_p} is the set of *p* activities that is decomposed from the flow model, where FM ? A.

The relationships among different entities in the FM are the following:

? M: A ? WS is a one-to-one mapping that gives a Web service that is assigned to execute an activity in the FM. To illustrate, $M(a_i) = ws_i$ is the Web service that is assigned to execute the activity a_i, where ws_i ? WS and a_i ? FM.

? D: WS ? BOOLEAN (i.e., true or false) tells whether a Web service contains a delegation model or not. To illustrate, $D(ws_i)$ = "true" means that the Web service ws_i contains a delegation model, where ws_i ? WS.

? DM: WS ? WS gives a set of Web services invoked in a delegation model if and only if the Web service contains a delegation model, i.e., $D(ws_i)$ = "true." To illustrate, $DM(ws_i) = \{ws_{i1}, ws_{i2}, ? , ws_{ik}\}$ is the set of *k* Web services invoked in the delegation model of ws_i, where $DM(ws_i)$? WS.

? MM: FM ? WS gives a set of Web services invoked in the FM. To illustrate, $MM(FM) = \{ws_1, ws_2, ? , ws_k\}$ is the set of *k* Web services invoked in the matchmaking model of the FM. To illustrate, $MM(FM) = \{? \ ws_i \mid M(a_i) = ws_i$ where a_i ? FM}, where MM(FM) ? WS.

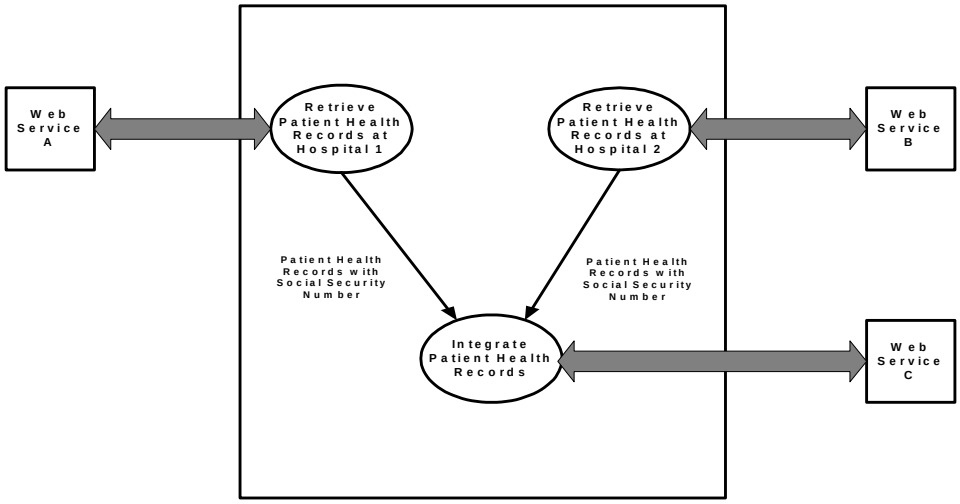

Figure 2. Patient Health Records Integration Process.

Referring to Webster [Webster Dictionary], the terminology "conflict of interest" in a general situation means that there is a conflict between the private interests and the official responsibilities of a person in a position of trust. Conflict of Interest (CIR) also exists in a loosely coupled Web services execution environment because Web services may work for many clients but some of them may compete with each other. In consequence, Web services may obtain sensitive information on competing clients. Web services are required to ensure client's commercial secrets do not leak via activity execution [Brewer and Nash 1989]. Figure 2 shows a business process (a flow model) 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 of the patients. Though, the sensitive or identifiable attributes from those patient health records should have been protected properly. If Web Service *A* or *B* can access the aggregate of patient health records (from both hospital *1* and *2*), Web service *A* or *B* may be possible to infer the patients' identities by using the data mining techniques [Glymour et al. 1996] with their information at their disposal (i.e., either patient health records at hospital *1* and *2*). Note that neither Web service *A* nor *B*

is allowed to release those patient health records to each other because of patient's privacy act. Therefore, either Web service *A* or *B* is not allowed to execute the consequent "Integrate Patient Health Records" activity because there exists CIR among these activities or roles. Thus it is required to have another third party such as Web service *C* to execute the "Integrate Patient Health Records" activity, i.e., Web service *A* ? Web service *B* ? Web service *C*.

In this paper, CIR is represented as a sequence of notation in the format of first order predicate calculus. Referring to the example of "Patient Health Records Integration Process" in Figure 2, the business process modeler can identify CIR in the matchmaking process as follows:

$$(\mathbf{M(a_1) = ws_a} \; ? \; \mathbf{(M(a_2)} \; ? \; \mathbf{ws_a} \; ? \; \mathbf{M(a_3)} \; ? \; \mathbf{ws_a)} \; ?$$
$$\mathbf{(M(a_2) = ws_b} \; ? \; \mathbf{(M(a_1)} \; ? \; \mathbf{ws_b} \; ? \; \mathbf{M(a_3)} \; ? \; \mathbf{ws_b)} \; ?$$
$$\mathbf{(M(a_3) = ws_c} \; ? \; \mathbf{(M(a_1)} \; ? \; \mathbf{ws_c} \; ? \; \mathbf{M(a_2)} \; ? \; \mathbf{ws_c)}$$

where ws = Web service, a_1 = "Retrieve Patient Records at Hospital 1," a_2 = "Retrieve Patient Records at Hospital 2" and a_3 = "Integrate Patient Health Records." This means that if a_1 is assigned to ws_a, then a_2 and a_3 cannot be assigned to ws_a and so on. This paper discusses CIR into a multi-lateral relation with exclusive-or (XOR) logic. For a particular Web service ws_i, an activity a_i and a set of *k* conflicting activities a_{i1}, a_{i2}, ? , a_{ik} where *k* ? 1, CIR can be represented as a binary relation in the format of first order predicate calculus as follows:

$$\mathbf{(? \; ws_i \; ? \; WS) \; (? \; a_i \; ? \; A) \; (? \; a_{set} \; ? \; A \; ? \; a_i \; ? \; a_{set})}$$
$$\mathbf{(? \; CIR(ws_i, a_i, a_{set}) \; ? \;\; ? \;\; ? \; \{M(a_i) = ws_i \; ? \; M(a_{ij}) = ws_i \; | \; ?_j \; a_{ij} \; ? \; a_{set}\})}$$

where XOR is represented with the symbol "? " and $a_{set} = \{a_{i1}, a_{i2}, ? , a_{ik}\}$. For simplification, CIR can be exempted if the following expression is enforced in the matchmaking process for the activity a_i:

$$\mathbf{M(a_i) = ws_i} \; ? \; \mathbf{M(a_1) = ws_i} \; ? \; \mathbf{M(a_2) = ws_i} \; ? \;\; ? \;\; ? \; \mathbf{M(a_k) = ws_i}$$

The business process modeler can specify the security assertions for the activity a_i in Web services endpoint properties. To illustrate, this paper demonstrates this idea on the context of WSEL as shown in Figure 3. The element is named "matchmaking" and it contains an attribute "exclusive-set" that specifies a set of activities (i.e., a_1, a_2, ? , a_k) where CIR exists with a_i.

```
<activity name="ai">
    <wsel:duration limit="30" metric="minutes"/>

    <wsel:retry maxNumber="10"/>

    <wsel:escalate>
        <wsel:staff
              who="select PID from Person where skill > 15"
              Invoke="c:\programs\org_query.exe"/>
    </wsel:escalate>

    <wsel:observed>
        <wsel:staff
```

```
            who="select PID
                 from Flows
                 where FlowName= "TotalSupplyFlow" "
            Invoke="c:\programs\org_query.exe"/>
    </wsel:observed>

    <wsel:conflict-of-interest>
        <wsel:matchmaking
              exclusive-set="a1, a2, ? , ak"/>
        <wsel:matchmaking
              inclusive-set="a1, a2, ? , ak"/>
        <wsel:delegation
              exclusive-set="ws1, ws2, ? , wsk"/>
    </wsel:conflict-of-interest>
</activity>
```

Figure 3. Encoding in WSEL (Based on the example on pages 83-84 in Leymann [2001])

To illustrate further, Figure 4 shows a business process called "Healthcare Quality Verification Process" [Stefanelli 2002] that includes three activities "Retrieve Patient Health Records at Hospital," "Verify Quality of Healthcare" and "Contact the Patients who Require Follow-up Services." The "Retrieve Patient Health Records at Hospital" activity retrieves the patient health records, and then the consequent "Verify for Quality of Healthcare" activity performs a data mining process on the patient health records to verify healthcare quality for each patient. Lastly the "Contact the Patients who Require Follow-up Services" activity contacts (e.g., by e-mail) those patients who require follow-up services based on the report generated from the previous activity. Again the sensitive and identifiable attributes of patient health records such as personal e-mail addresses are all protected in this scenario. It is obvious that there exists CIR between the first two activities because of the conflicting roles in these two activities as follows:

$$\mathbf{(M(a_1) = ws_a \; ? \;\; M(a_2) \; ? \; ws_a) \; ? \; (M(a_1) = ws_b \; ? \;\; M(a_2) \; ? \; ws_b)}$$

where ws = Web service, a_1 = "Retrieve Patient Health Records at Hospital" and a_2 = "Verify Quality of Healthcare." In particular, the "Verify for Quality of Healthcare" activity may identify certain patients who need further healthcare services to maintain the quality of services. Then, the "Contact the Patients who Require Follow-up Services" activity needs the patient's contact information (e.g., e-mail addresses) for contacting the patients who require follow-up services. However, CIR exists if the patient's contact information is passed to a third party by the hospital because of patient's privacy [Cushman 1996]. In this situation, an appropriate approach is that the Web service who executes the "Retrieve Patient Health Records at Hospital" activity should also be the Web service who executes the "Contact the Patients who Require Follow-up Services" activity, i.e., Web service *A* = Web service *C*. The purpose is trying to avoid passing the sensitive or identifiable information from one party to other parties. Again, the business process modeler can identify CIR for these two activities in the matchmaking process as follows:

$$\mathbf{(M(a_1) = ws_a \; ? \;\; M(a_3) = ws_a) \; ? \; (M(a_1) = ws_c \; ? \;\; M(a_3) = ws_c)}$$

where ws = Web service, a_1 = “Retrieve Patient Health Records at Hospital 1” and a_3 = “Contact the Patients who Require Follow-up Services.”

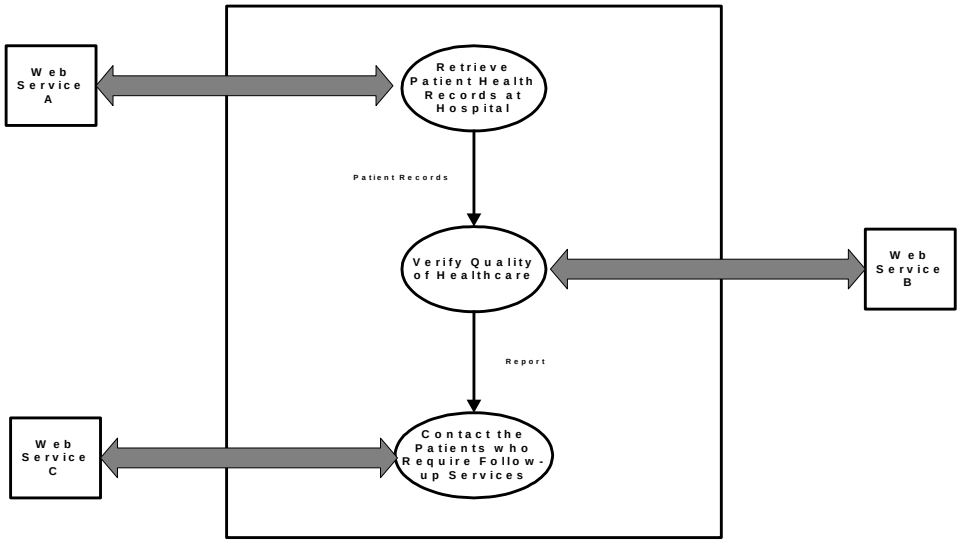

Figure 4. Healthcare Quality Verification Process.

This paper extends this bi-lateral relation into a multi-lateral relation with exclusive-and (XAND) logic. For a particular Web service ws_i, an activity a_i and a set of *k* conflicting activities a_{i1}, a_{i2}, ? , a_{ik} where *k* ? 1, CIR can be represented as a binary relation in the format of first order predicate calculus as follows:

$$(? \mathbf{ws_i} ? \mathbf{WS}) (? \mathbf{a_i} ? \mathbf{A}) (? \mathbf{a_{set}} ? \mathbf{A} ? \mathbf{a_i} ? \mathbf{a_{set}})$$
$$(? \mathbf{CIR(ws_i, a_i, a_{set})} ? \ ? \ ? \ \mathbf{\{M(a_i) = ws_i} ? \ \mathbf{M(a_{ij}) = ws_i} \mid ? _j \mathbf{a_{ij}} ? \ \mathbf{a_{set}\})}$$

XAND is represented with the symbol “? .” In Figure 3, the security assertion is named as “inclusive-set”. Similarity as above, CIR can be exempted if the following expression is enforced in the matchmaking process for the activity a_i:

$$\mathbf{M(a_i) = ws_i} ? \ \mathbf{M(a_1) = ws_i} ? \ \mathbf{M(a_2) = ws_i} ? \ \ ? \ \ ? \ \mathbf{M(a_k) = ws_i}$$

In a delegation model, there also exists CIR similar to those situations discussed in the matchmaking model. This paper assumes that the business process modelers may not have enough information about the private flow model of Web service [Leymann 2001] such as the interactions between those Web services involved in the delegation model. With incomplete information, it is reasonable to assume that the business process modeler is only able to define an exclusive set that identifies a set of Web services that is not allowable to execute any piece of work (sub-activity) of an activity. From another aspect, CIR exists when one of the Web services invoked in a delegation model is a competitor to the Web services requestor. Let the exclusive set for a delegation model be:

? $ES(a_i) = \{ws_{i1}, ws_{i2}, ? , ws_{ik}\}$ is the set of *k* Web services that are not allowed to execute any sub-activity decomposed from the activity a_i by any means, where $ES(a_i)$? WS.

For a particular Web service ws_i, an activity a_i and an exclusive set of *k* conflicting Web services ws_{i1}, ws_{i2}, ? , ws_{ik} where *k* ? 1, CIR can be represented as a binary relation in the format of first order predicate calculus as follows:

$$(? \; ws_i \; ? \; WS) \; (? \; a_i \; ? \; A) \; (? \; M(a_i) = ws_i \; ? \; D(ws_i))$$
$$(? \; CIR(ws_i, a_i) \; ? \;\; ? \;\; ? \; (DM(ws_i) \; ? \; ES(a_i) = ? \;))$$

It is interpreted as follows. If the Web service ws_i is assigned to execute the activity a_i and the Web service ws_i contains a delegation model, the delegation performed by the Web service ws_i is not allowed to invoke those Web services in the exclusive set. Again, this paper demonstrates this element as "delegation" and it contains an attribute "exclusive-set" as shown in Figure 3.

4. CONCLUSIONS AND FUTURE RESEARCH

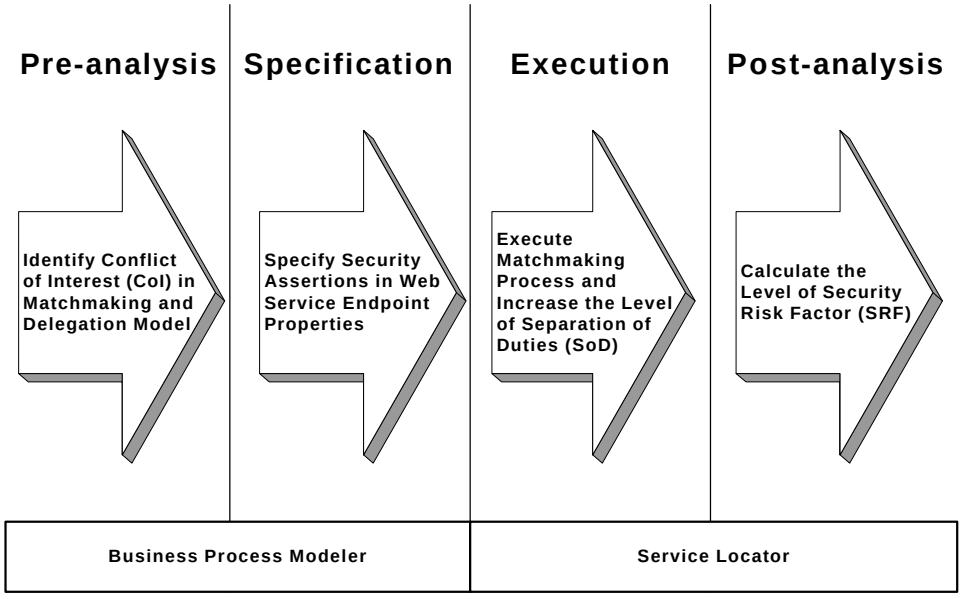

Figure 5. Four Phases for Web Services Matchmaking Process (WSMP).

This paper extends the WSEL by the appropriate extensibility elements to specify the Conflict of Interest (CIR) for the processes of matchmaking and delegation as some of the security assertions in endpoint properties. This work can be expanded in several directions. In particular, we are currently working on a formal model for Web Services Matchmaking Process (WSMP) with the security assertions proposed in the endpoint properties. This is well-known that, one of the major processes in a loosely coupled Web services execution environment is *matchmaking*, that is, an appropriate Web service is assigned to execute an activity by a service locator [Zhang and Zhang 2002]. Figure 5 shows the four phases in the proposed WSMP model. As discussed above, the business process modeler has to identify whether there exists any CIR in every activity assignment during the pre-analysis phase. If CIR does exist, the business process modeler has to specify it as security assertions at the endpoint properties for each activity during the specification phase. In the execution phase, the service locator executes the matchmaking process and tends to increase the level of Separation of Duties (SoD) [Gligor et al. 1998]. Traditionally, SoD is a well-known principle for preventing fraud by identifying conflicting roles and ensuring that the same individual can belong to at most one conflicting role [Ahn and Sandhu 1999]. In the post-analysis phase, the service locator calculates the level of Security Risk Factor (SRF) [Hung et al. 1999] based on the matchmaking

pattern generated from the previous phase. Essentially, the SRF measures the level of risk associated with a set of Web services executing a set of inter-dependent activities. The SRF value can be used as an indicator for adjusting and evaluating the matchmaking patterns.

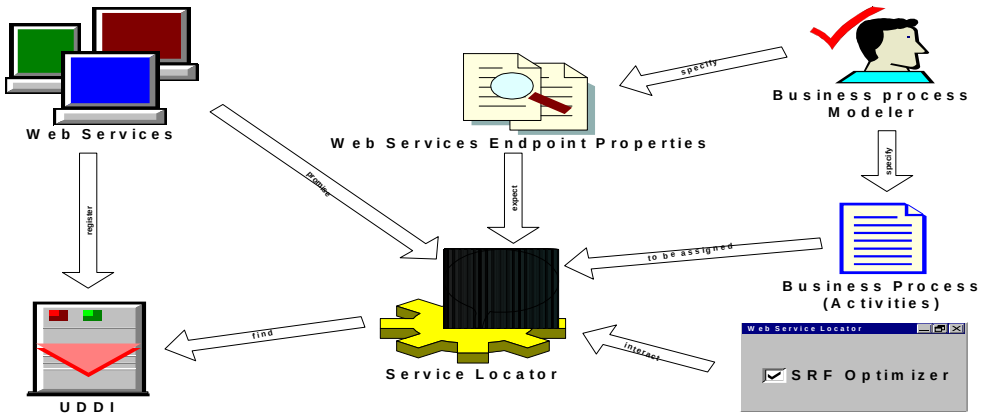

Figure 6. Multilateral Matchmaking

Based on the WSMP, Figure 6 shows the basic idea of building such a service locator. The matchmaking mechanism to this specific environment requires multi-participants in the match (i.e., Web services, activities and the security assertions at endpoint properties), which requires a multilateral matchmaking approach [Raman et al. 2000]. Lastly, we are working on the implement plan of a demonstrator for this demo system by using Web service technologies.

REFERENCES

AHN, GAIL-JOON AND RAVI SANDHU. 1999. The RSL99 language for role-based separation of duty constraints. Proceedings of the ACM Workshop on Role-Based Access Control, 43-54.

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

CUSHMAN, R. 1996. Information and medical ethics: protecting patient privacy. IEEE Technology and Society Magazine, vol. 15, 32-39.

GLIGOR, VIRGIL D., SERBAN I. GAVRILA AND DAVID FERRAIOLO. 1998. On the formal definition of separation-of-duty policies and their composition. Proceedings of 1998 IEEE Symposium on Security and Privacy, 172 -183.

GLYMOUR, C., D. MADIGAN, D. PREGIBON AND P. SMYTH. 1996. Statistical Inference and Data Mining. Communications of the ACM, vol. 39, no. 11, 35-41.

HINKE, THOMAS H. 1989. Trusted server approach to multilevel security. In Proceedings of Annual Computer Security Applications Conference, 335-341.

HOFREITER, B., C. HUEMER AND W. KLAS. 2002. ebXML: status, research issues, and obstacles. 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. eAI Journal, September 2002, 45-47.

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

HUNG, PATRICK C. K., KAMALAKAR KARLAPALEM AND JAMES GRAY III. 1999. Least Privilege Security in CapBasED-AMS. The International Journal of Cooperative Information Systems, vol. 8, no. 2 & 3, 139-168.

IBM CORPORATION. 2001. Web Services Conceptual Architecture (WSCA 1.0).

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

IBM CORPORATION. 2002. Web Services Security (WS-SECURITY), Version 1.0.

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

LEYMANN F. 2001. Web Services Flow Language (WSFL 1.0). IBM Corporation.

LEYMANN F., D. ROLLER AND M.-T. SCHMIDT. 2002. Web services and business process management. IBM Systems Journal, vol. 41, no. 2, 198-211.

LIN, T. Y. 1990. Chinese wall security policy - An aggressive model. Proceedings of Annual Computer Security Applications Conference, 282-289.

LIN, T. Y. 2000. Chinese Wall security model and conflict analysis. Proceedings of IEEE Computer Society's International Computer Software and Applications Conference, 122-127.

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

RAMAN, RAJESH, MIRON LIVNY AND MARVIN SOLOMON. 2000. Resource management through multilateral matchmaking. Proceedings of the Ninth International Symposium on High-Performance Distributed Computing, 290-291.

STEFANELLI, M. 2002. Knowledge Management to Support Performance-based Medicine. Methods of Information in Medicine 1/2002, 36-43.

THATTE, S. 2001. XLANG - Web Services for Business Process Design. Microsoft Corporation.

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

VARLAMOV, STAN. 2002. Security Strategies for EAI. eAI Journal, September 2002, 41-44.

WEBSTER DICTIONARY: http://www.webster.com

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

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

YANG, A. 2002. Web Services Security. eAI Journal, September 2002, 19-23.

ZHANG, ZILI AND CHENGQI ZHANG. 2002. An improvement to matchmaking algorithms for middle agents. Proceedings of the first international joint conference on Autonomous agents and multiagent systems, 1340-1347.

CALL FOR PARTICIPATION

IEEE TFEC Membership

Join Task Force on Electronic Commerce

The TFEC is always looking for members and volunteers to promote its activities. The TFEC *membership is FREE* and is open to all irrespective of whether your are IEEE/Computer Society member or not. However, IEEE or CS members enjoy special privileges including: discount on registration fee for TFEC sponsored events, right to vote, receive any printed newsletter. If you are not a member of IEEE or CS, It's time to become a member.

How to become TFEC member?

- **Online-Option (Please select Task Force- E-Commerce)**:

  https://newton.computer.org/correspo.nsf/signup?OpenForm

- **Second Option:** All you need to do is fill the **Application Form (Text Format)** and email OR print it out and fax/post directly to IEEE CS, USA as described in the application.

Any questions, please contact the following TFEC Co-Chairs:

Jen-Yao Chung ***(TC Co-Chair)***
IBM T. J. Watson Research Center
P. O. Box 218
Yorktown Heights, New York 10598
USA
Phone: +1 914 945 3422
Fax: +1 914 945 3242
Email: jychung@us.ibm.com

Kwei-Jay Lin ***(TC Co-Chair)***
Department of Electrical and Computer Engineering
University of California, Irvine
Irvine, CA 92697-2625
USA
Phone: +1 949 824 7839
Fax: +1 949 824 2321
Email: klin@uci.edu