A Policy Framework for Collaborative Web Service Customization

Haiqi Liang, Wei Sun, Xin Zhang, Zhongbo Jiang
IBM China Research Lab
{lianghq; weisun; zxin; jiangzb}@cn.ibm.com

Abstract: A web service usually serves many consumers, with each consumer having its specific functional or nonfunctional requirements. The web service should be customizable to meet those requirements, as is especially apparent for complex web services with large granularity such as Software-as-Service (SaS). This paper proposes an approach to facilitating web service customization in the programatical way through the collaboration between service provider and consumer. A specification for declaring web service customization policy is defined based on the WS-Policy framework, through which service provider can declare the service's customization capabilities. A service consumer can make customization requests within the scope defined by the customization policies. The customization requests are raised in structuralized customization directives. A web service consumption process and a supporting framework accompanied with customization policy are introduced accordingly. Finally, a case study is present.

1 Introduction

Web service is gaining widespread adoption in modern applications. As modular and self-described application, a web service with certain functionality and quality properties can be easily discovered and invoked over the Internet which can satisfy specific consumer requirements [UD]. A single web service implementation usually serves multiple service consumers; a popular web service serves thousands or even millions of consumers, such as the web service provided by Amazon [AM], eBay [EB], and Google [GO]. There is a trend that more and more business applications are delivered as web services, e.g. CRM (Customer Relationship Management) services [SF, SI] and ERP (Enterprise Resource Planning) services [IT]. These web services are named as Software as Service (SaS) by the industry, and usually have larger granularity with complex functions. When subscribed, these services can be accessed by consumers through Internet to support their business requirements. As different consumers usually have special requirements for the business application in terms of user interface, business process and data, it is critical for the SaS services to provide service consumers easy personalization and customization capabilities according to their own requirements. This has been demonstrated by leading SaS services, for example, Salesforce.com CRM service

provides CustomForce utility, Intacct.com ERP service provides CustomERP utility.

Both personalization and customization techniques can support to provide specific service consumer tailored functions on top of a base version service. Personalization [PC] technique enables web service provider to gather user-information during interacting with the consumer, and then deliver appropriate content and services to the consumer based on the collected user-information; Customization technique can empower the consumer actively change the functions of the service according to certain process or policy. There are various technologies introduced to support web service personalization [KT01, KS03, JR03, BM01, JR04]. This paper focuses on the technology which can streamline web service's customization, especially the collaboration between service provider and consumer.

2 Related work

There has been some research work to support web service customization in different ways. Ardissono L. [AG04] introduced a conversation framework and conversation flow language to support the management of communication between web service user and providers, which aims to include user in the loop of provisioning of customized web service. S.Abiteboul [AA03] proposed to use intension XML document as the mechanism to enable web service customization. Some data on the input/output is given explicitly, while other parts are defined by embedded "programs call" that dynamically generates the relevant data for the specific user. These are different technologies to enable realization of web service customization. As users usually actively involve into the customization process of web service according to their own requirements, it is very key for web service provider to share its customization capability to its consumers which can guide them to perform the customization process within the scope supported by service provider. Currently there are many web service customization should be performed by human user supported by necessary tools, for example, CustomForce [SF] utility provided by Salesforce.com CRM web service. However, nowadays web service customization capabilities are described in free format documents like "user manual" or "developer guide". Customizing a web service in a programatical way can not be easily achieved.
This issue has been noticed in some web service standards related activities. WSIA [OA] defines the framework for creating interactive web services. According to the WSIA requirement documentation, WSIA web service producers may associate adaptation metadata with a property describing how that property may be changed by the consumer. Adaptation description metadata defines "permissions" which control what aspects of the property are open for change by consumer, which includes deleting data element, adding new data elements, overriding of values of a given data property, adding or modifying relationships across elements in the prop-

erty, etc. However there is no specification design yet and there is no implementation as well. WS-Policy [WP] provides a flexible and extensible grammar for expressing capabilities, requirements, and general characteristics of entities in an XML web service-based system. WS-Policy defines a framework and a model to express these properties as policies. The specification will identify several basic service attributes, including privacy attributes, encoding formats, security token requirements, etc. Although WS-Policy can be used to describe those nonfunctional capabilities of a web service, the customization capability that is addressed in this paper is mostly about functional part of the web service which will have its uniqueness when described with WS-Policy.

To tackle the issues addressed above, this paper proposes an approach to enabling collaborative web service customization between web service provider and consumer. Based on WS-Policy framework, this paper defines a WS-CustomizationPolicy schema to capture the customization capabilities supported by the service provider. The collaborative web service customization framework and related tools can empower the provisioning of WS-CustomizationPolicy from service provider to consumer, controlling the customization requirements generation in the consumer side according to the policy, creating the customization directives described by predefined instruction set to support the realization of customization.

The rest of this paper is organized as follows. Section 3 analyzes a typical web service consumption process with web service policy involved, and studies the revised consumption process where the web service customization policy is involved. The policy framework of web service customization is then proposed in section 4, including the customization policy definition, and the customization requirements specification through customization directives. Section 5 demonstrates the proposed customization framework with a case study. Section 6 summaries and concludes this paper.

3 Collaborative web service customization

As the demand for customization becomes more and more appealing during web service consumption lifecycle, different service providers begin to develop their own portal or schema to enable the customization of their services. These customization capabilities are in vendor specific mode, and without common understanding between the service provider and the service consumer. The service provider may or may not define some constrain and policy through their customization portal or schema; even if it has been defined, the service consumer may not acknowledge it. This may cause potential issues that make the service consumption less effective.

In this paper we propose to define the web service customization policy, which works as a specification or protocol between the service provider and service consumer. The service consumption process should be accelerated by enabling the

customization policy description and sharing between service provider and service consumer.

3.1. Web service consumption with WS-Policy

WS-Policy provides a general purpose model to communicate policy assertions regarding the service, for example, how requesters will be authenticated, the preferred transport, what QoS is offered, and the terms of the SLA.
Figure 1 illustrates a typical web service consumption process following various web service protocols. After a web service is published to a service registry by service provider, it is then discovered by service consumer using UDDI, WSIL, or WS-Discovery, which specifies where to locate the service. From that location, the policies governing the use of the service are retrieved using WS-Policy. Then the description of the service itself can be retrieved in the form of WSDL script. This can be accomplished by browsing the web address, for example, http://paydins.com/regjourney?WSDL. The WSDL is used by service consumer to build the service requesting application by identifying the to-be invoked operations and the supporting request messages. Service consumer will take actions to make sure the service requesting application will conform to the policy assertions defined in WS-Policy. The service is requested by addressing a SOAP message to the appropriate endpoint provided in the WSDL.

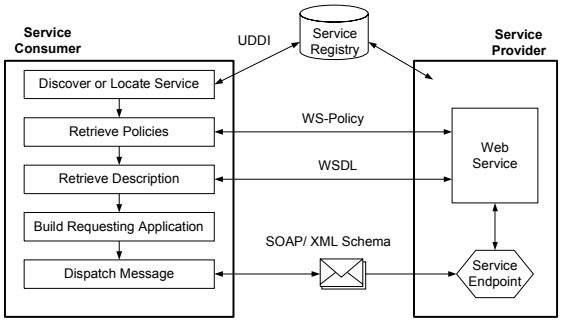

Figure 1: Web service consumption process without customization capability

3.2. Web service consumption with customization policy

As we discussed before, customization is a general requirement for web service consumption. Figure 2 illustrates the structure of a typical WSDL script. A service implementation binds to a specific portType; each port type implements some operations. The input or output messages of each operation is defined as message types, which is separately defined in XML Schema. This paper tackles with the

customization of message type, but the approach can be extended to other aspects of the web service.

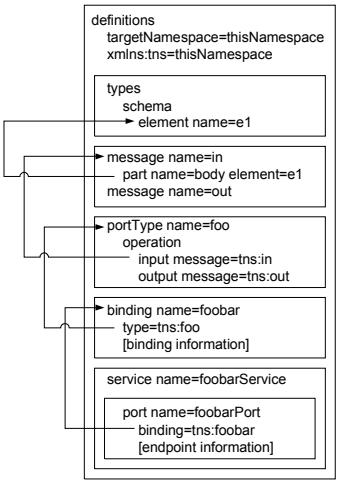

Figure 2: Web service description language structure

To provide customization support, service provider can declare its customization capabilities through customization policy. However, web service consumption process with customization policy will be different from that of general web service policy, where the policy will not result in update of the web service specification, such as WSDL and WS-SecurityPolicy [WSP]. General web service policy is used by service consumer to communicate with the service instance at service usage time. While the service customization policy proposed in this paper is used by service provider to share its web service's customization capability to service consumer, service consumer can then propose web service customization request within the scope defined by the policy and then invokes the updated service instance later as usual. As a result, the web service consumption process with customization policy will be as Figure 3.

In Figure 3, we assume that a customization policy of the web service is provided by service provider. Compared with the usual case in Figure 1, where general service policies (such as security policies) are conformed to, Figure 3 shows the revised communication between service consumer and service provider because of the introduction of service customization policy. In Figure 3, service consumer retrieves policy definition and default WSDL script first, and then service consumer will customize the service definition (WSDL script) in the scope defined by the policy. The customization operations will finally result in a customization request to service provider. The customization request is intercepted by a customization engine at service provider side. The customization engine will realize the ser-

vice customization and most likely build a new service implementation (customized web service). The customized web service is rarely published to public service registry because it is only implemented for certain consumer. Service consumer then retrieves the updated WSDL script of the service, builds its service requesting application, and invokes the service as usual to satisfy its specific requirements.

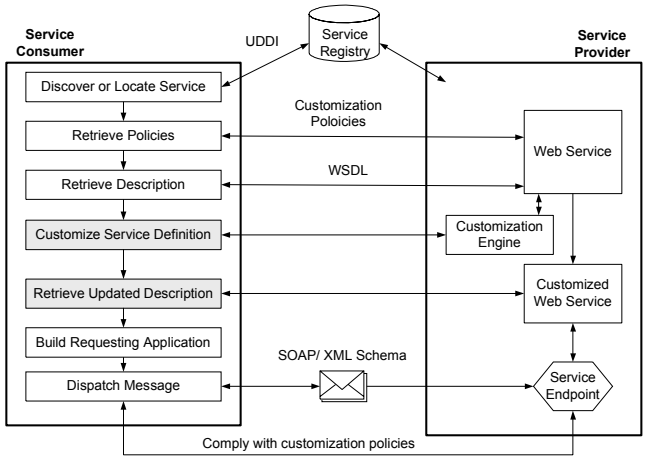

Figure 3: Web service consumption process with customization policy

As we can see in above process, customization policy is the protocol between service provider and consumer which specifies what customization capabilities are supported by service provider (or customization engine), and what constraints that service consumer should abide by when customizing the service definition.

4 A framework for collaborative web service customization

According to the web service consumption process explained above, this paper proposes a general collaborative web service customization framework to facilitate web service customization, as Figure 4 shows.
In this framework, service provider will utilize Customization Policy Builder tool to define the customization capabilities of its published web service. The generated customization policy is published on service registry along with its WSDL script. At the service consumer side, service consumer will utilize a Customization Controller to customize the WSDL script, and then generate a set of customization directives. The customization directives are sent to the customization engine of service provider to realize the customization.

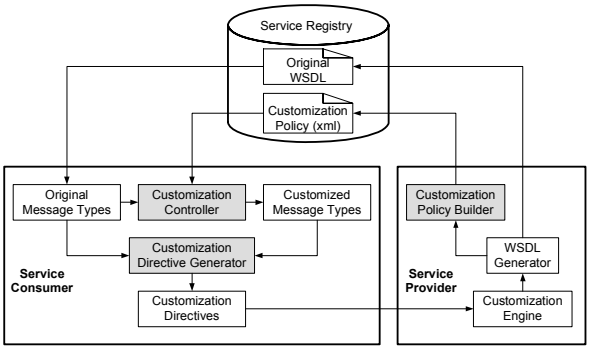

Figure 4: Collaborative web service customization framework

We have developed a set of tools to implement this framework. As we can see, there are two key issues in this framework: one is the customization policy definition; the other is the customization directive specification. The following of this section will focus on these aspects.

4.1. Exposing customization capabilities with customization policy

WS-Policy defines a framework for allowing web services to express their constraints and requirements. Such constraints and requirements are expressed as policy assertions. This paper defines a set of customization policy assertions in the WS-Policy framework, which capture customization features discussed in previous sections; we call it WS-CustomizationPolicy. As data customization is the primary requirement, the WS-CustomizationPolicy only defines the data related customization capabilities, i.e., the XML schema defined in the <types> of WSDL script. Figure 5 shows the demonstrative XML schema structure and the main relationship between XML schema elements.

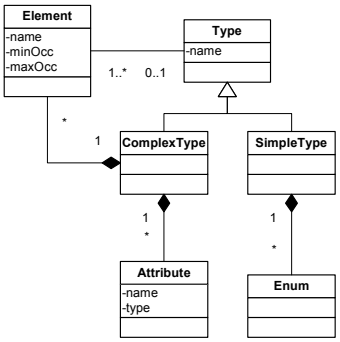

Figure 5: Demonstrative XML Schema structure

According to the relationship, following is a list of customizable aspect of XML schema and the operations allowable on them:

- *Message root*: addElement, removeElement, addType, removeType;
- *Element*: changeName, changeType, changeMinOcc, changeMaxOcc;
- *ComplexType*: changeName, addElement, removeElement, addAttribute, removeAttribute;
- *SimpleType*: addEnum, removeEnum;
- *Attribute*: changeName, changeType;

The WS-CustomizationPolicy schema is defined according to above research. This is an example customization policy:

```
<wsp:Policy xmlns:wsp=http://schemas.xmlsoap.org/ws/2004/09/policy
xmlns:wsc=http://schemas.xmlsoap.org/ws/2005/08/custom
xml:base="http://crl.ibm.com/policies" wsu:id="POType">
  <wsc:CComplexTP wsp:Optional="true">
    <wsc:Operation>addElement</wsc:Operation>
    <wsc:Operation>removeAttribute</wsc:Operation>
  </wsc:CComplexTP>
</wsp:Policy>
```

This example means for the original data structure, the complex type "POType" can be either added a new child element or be removed an attribute, or both. However, adding a new attribute or changing a child element name is not allowed according to this customization policy.

4.2. Customizing web service specification according to customization policy

In previous sections of this paper, we proposed and defined the web service customization policy which declares customization capabilities supported by service provider. Once customization requirements have been identified, service consumer need to submit a set of web service customization directives to the customization engine at service provider side for customization realization (see Figure 4). This section proposes an approach to describing customization directives for service consumer. The customization directives can formalize the comparison results of the customized WSDL message types and the original WSDL message types, such as:

```
<directive>
  <operation type="changeType">
  <parameters>
    <parent type="attribute" name="/PurchaseOrderType/{orderDate}" />
    <change from="xs:date" to="xs:string" />
```

```
  </parameters>
  </operation>
</directive>
```

As we can see, different entities of the XML schema have different operations on them (see Section 4.1), and accordingly the related information will varies with operations in the customization directives.

5 Case study

This section takes a fictitious example to demonstrate our approach of collaborative web service customization. HomeSys Corp. is a manufacturing company which wants to improve its management of customer related business operations. After evaluation, HomeSys decides to subscribe a CRM web service hosted by the CRMOnline.com, instead of deploying a CRM application in house. As there has already been an ERP application running in its Intranet, HomeSys definitely doesn't want the CRM web service be an information island. There're various kinds of information needs to be shared between the ERP application and CRM web service. In the following demonstration, we will take product information sharing as an example in this case.

A specific requirement of HomeSys is that the product information needs to be shared and synchronized between the CRM web service and ERP application, i.e., when the product manager of HomeSys adds or modifies the product information in ERP, the corresponding information should be populated to CRM system simultaneously. A system integrator, odRSI, takes the responsibility of integrating these two systems. The following schema shows the data portion in the WSDL script of the CRM web service that is related with customization declarations.

```
<types>
<xs:schema … PolicyURIs="http://crl.ibm.com/policies#messageroot">
  <xs:element name="product" type="ProductType">
    <xs:complexType name="ProductType"
        wsp:PolictURIs=http://crl.ibm.com/policies#producttype>
      <xs:ComplexContent>
        <xs:sequence>
          <xs:element name="ProductID" minOccurs="0" type="xs:string"/>
          <xs:element name="Name" minOccurs="0" type="xs:string" />
          <xs:element name="ProductDescription" minOccurs="0"
          type="xs:string" />
          <xs:element name="AvailableTime" minOccurs="0"
          type="xs:dateTime" />
          <xs:element name="ProductUnit" minOccurs="0"
          type="xs:string" />
          <element name="ListPrice" minOccurs="0" type="xs:double" />
        </xs:sequence>
      </xs:complexContent>
    </complexType>
```

```
  </xs:element>
<xs:schema>
</types>
```

This is the corresponding product related customization policy in the CRM web service:

```
<wsp:Policy … xmlns:wsc=http://schema.xmlsoap.org/2005/08/custom
base="http://crl.ibm.com/policies" wsu:id="messageroot">
  <wsc:CMessageRoot wsp:Optional="ture">
    <wsc:Operation>addType</wsc:Operation>
  </wsc:CMessageRoot>
</wsp:Policy>
<wsp:Policy … xmlns:wsc=http://schema.xmlsoap.org/2005/08/custom
base="http://crl.ibm.com/policies" wsu:id="producttype">
  <wsc:CComplexTP wsp:Optional="ture">
    <wsc:Operation>addElement</wsc:Operation>
      <wsc:Operation>changeName</wsc:Operation>
      <wsc:Operation>addAttribute</wsc:Operation>
  </wsc:CComplexTP>
</wsp:Policy>
```

The product information structure in the CRM service description is relatively simple. For better management and navigation of the products information in ERP, there is a standalone database table that manages the product's category information, so that all products in the system can be categorized accordingly. Also, there are some differences in the product index ID coding: the ID in the CRM web service is a system-generated random number, while in the ERP application the ID is codified in a meaningful manner (e.g. ProductType + ProductPartnumber + Random).

After studying customization capabilities of the CRM web service, odRSI identifies two major customizations that should be taken for the CRM web service:

- Enable the product categorization capability in the CRM web service.
- Extend product data structure of CRM web service to store the internal product code defined in ERP.

Corresponding customization operations are as below:

- Add a new "Category" data type for the project in the CRM application.
- Modify the existing "Product" data type of CRM to add a "ProductCategoryID" element that refers to the corresponding product category.
- Modify the existing "Product" data type of CRM to add an "InternalProductCode" element that stores the ERP defined product ID.

From the specified customization policy above, we can see that these customization requirements can be fulfilled by the CRM web service. With the help of developed tools in this paper, odRSI submits the three customization requests. Then the customization engine on CRM service provider side will handle these requests, validate their policy compliance, and finally accept or reject the customization requests. The following shows part of the targeted CRM web service that current CRM web service should be customized into:

```
<types>
<xs:schema …>
  <xs:element name=”product” type=”ProductType” />
  <xs:element name=”category” type=”CategoryType” />
  <xs:complexType name="ProductType">
    <xs:ComplexContent>
      <xs:sequence>
        <xs:element name="ProductID" minOccurs="0" type="xs:string" />
        <xs:element name="Name" minOccurs="0" type="xs:string" />
        <xs:element name="ProductDescription" minOccurs="0"
        type="xs:string" />
        <xs:element name="AvailableTime" minOccurs="0"
        type="xs:dateTime" />
        <xs:element name="ProductUnit" minOccurs="0" type="xs:string"/>
        <xs:element name="ListPrice" minOccurs="0" type="xs:double" />
        <xs:element name="InternalProductCode" minOccurs="0"
        type="xs:string" />
        <xs:element name="ProductCategoryID" minOccurs="0"
        type="xs:string" />
      </xs:sequence>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="CategoryType">
    <xs:ComplexContent>
      <xs:sequence>
        <xs:element name="ProductCategoryID" minOccurs="0"
        type="xs:string" />
        <element name="ParentCategoryID" minOccurs="0"
        type="xs:string"/>
        <element name="ProductCategoryName" minOccurs="0"
        type="xs:string" />
        <element name="ProductCategoryDesc" minOccurs="0"
        type="xs:string" />
      </xs:sequence>
    </xs:complexContent>
  </xs:complexType>
</xs:schema>
</types>
```

After all the customization requests have been submitted, odRSI can now continue to perform the integration development work. Meanwhile, odRSI can generate the customization directives using the customization directive generator, and then submit the customization directives to the CRM service provider.

The generated customization directives are listed as following:

```
<directives xmlns="http://crl.ibm.com/wscustom/directive/">
  <directive>
    <operation type="addElement">
      <parameters>
        <parent type="complexType" name="/Product" />
        <element name="InternalProductCode" type="xs:string"
         minOccurs="0" />
        <element name="ProductCategoryID" type="xs:string"
        minOccurs="0" />
      </parameters>
    </operation>
  </directive>
  <directive>
    <operation type="addType">
      <parameters>
        <type name="ProductCategory" />
      </parameters>
    </operation>
  </directive>
  <types>
    <complexType name="ProductCategory">
      <ComplexContent>
        <sequence>
          <element name="ProductCategoryID" minOccurs="0"
          type="xs:string" />
          <element name="ParentCategoryID" minOccurs="0"
          type="xsd:double" />
          <element name="ProductCategoryName" minOccurs="0"
          type="xs:string" />
          <element name="ProductCategoryDesc" minOccurs="0"
          type="xs:string" />
        </sequence>
      </complexContent>
    </complexType>
  </types>
</directives>
```

The customization directives will be populated to the CRM service provider and processed in two ways. In one way, the customization engine interprets the customization directives and makes the change accordingly. In the other way, the CRM service people apply the customization manually by using other utilities under the guideline described by the customization directives. In parallel, odRSI can continue to perform the integration work without waiting until the customization is made effective by the service provider.

6 Conclusions and future work

To acquire widespread adoption, a web service implementation, especially complex web service such as SaS service, should satisfy customized requirements. Although a lot of literatures are focusing on how to enable web service customization at the service provider side such as implementation of the customization engine, this paper focuses on the customization policy which could help streamline the customization collaboration between service consumer and provider. Before the web service can be consumed, its customization policies should be published by the service provider. A service consumer can identify and submit its customization requests according to the policy definition. This paper proposes web service customization policy, its usage process and a framework to support the collaboration between service consumer and provider, so that the web service customization process can be conducted in a programatical way. The customization policy framework proposed in this paper is implemented with tools and proved with real case. Nevertheless, the customization capability in current framework focuses on the message structure customization, which is only part of web service customization category. We will further study a whole web service customization framework, including web service operation customization, and user interface (interaction) customization.

References

[UD] UDDI, http://www.uddi.org/

[AM] Amazon E-Commerce Service, http://www.amazon.com/gp/browse.html/104-3921307-0838325?%5Fencoding=UTF8&node=3435361

[EB] eBay SOAP Developer Center, http://developer.ebay.com/soap/

[GO] Google, http://www.google.com/apis/

[SF] Sforce on-demand integration, http://www.sforce.com

[SI] Siebel CRL on-demand, http://www.crmondemand.com/index.jsp

[IT] Intacct ERP on demand, http://us.intacct.com/

[PC] Personalization Consortium, http://www.personalization.org/index.html

[KT01] Kantor, T. Adaptive Personal Mobile Communication – Service Architecture and Protocols (PhD thesis). *Stockholm University/KTH*, Dec. 2001.

[KS03] Harumi Kuno, Akhil Sahai. "My Agent Wants to Talk to Your Service: Personalizing Web Services through Agents", *Agentcities: Challenges in Open Agent Environments*, Springer-Verlag, pp. 25-31, 2003.

[JR03] Rykowski J. Agent Technology for Secure Personalized Web Services. *24th International Scientific School ISAT 2003*. Szklarska Porba (Poland), September 2003, pp. 185-193.

[BM01] M. Bonett. Personalization of Web Services: Opportunities and Challenges, June 2001, Ariadne Issue 28. http://www.ariadne.ac.uk/issue28/personalization /intro.html

[JR04] Rykowski, J. "Virtual Web Services - Application of Software Agents to Personalization of Web Services". *6th International Conference on Electronic Commerce ICEC 2004*: Engineering the New Landscape, Delft (The Netherlands), October 2004. ACM Publishers, pp. 419-428.

[CE04] A. Corallo, G. Elia, A. Caforio, and G. Solazzo. "Service Customization supporting an Adaptive Information System". *KES International (Knowledge-Based Intelligent Information & Engineering Systems)*, 2004.

[AG04] L. Ardissono, A. Goy, G. Petrone, M. Segnan. Interaction with Web Services in the Adaptive Web. *Adaptive Hypermedia and Adaptive Web-Based Systems*. 2004, pp. 14-23.

[AA03] S. Abiteboul, B. Aman, J. Schema-driven Customization of Web Services. *VLDB*. 2003.

[WSE] Web Service Experience Language, http://www-128.ibm.com/developerworks /library/specification/ws-wsxl/

[OA] OASIS Web Services Interactive Applications, http://www.oasis-open.org /committees/wsia/

[WP] Web Services Policy Framework (WS-Policy), http://www.ibm.com /developerworks/webservices/library/specification/ws-polfram/

[WSP] WS-SecurityPolicy, ftp://www6.software.ibm.com/software/developer/library /ws-secpol.pdf

[WSU] Using WSDL in a UDDI Registry, Version 2.0.2. http://www.oasis-open.org /committees/uddi-spec/doc/tn/uddi-spec-tc-tn-wsdl-v202-20040631.pdf