Understanding WSDL in a UDDI registry, Part 2

Usage scenarios in publishing and finding WSDL service descriptions

Peter Brittenham

As explained in the first article in this series, the versatility of Web Services Description Language (WSDL) makes it more complex when using it with UDDI registries. In this second part, Peter applies the process for mapping WSDL into a UDDI environment using various WSDL usage scenarios.

As described in the first part to this series (see Resources), a complete WSDL service description consists of a service interface and a service implementation. Each of these service descriptions can reside in different physical WSDL documents.

This part of the series contains usage scenarios for publishing and finding complete WSDL service descriptions in a UDDI Registry. For each usage scenario, I will describe a method to publish a complete WSDL service definition in a UDDI registry. These methods are described based on the procedures in the UDDI best practices document, as well as usage conventions defined in both the UDDI Programmer's API Specification and UDDI Data Structure Reference (see Resources).

Scenario 1: Service interface without a service implementation

When a service interface provider publishes a WSDL service interface definition as a tModel, that tModel can be referenced from any UDDI businessService. This scenario shows how to publish a businessService with a reference to a tModel that is associated with a WSDL service interface. For this scenario, the UDDI businessService is not associated with a WSDL service implementation definition. This scenario can be implemented using the process described in the UDDI best practices document.

WSDL service interface definition

The service interface document contains all of the WSDL types, message, portType and binding elements. For this scenario, the Web service contains one operation, and the binding is specified using SOAP. This example does not include a types element, since this usage scenario does not require data type definitions for the messages.
Listing 1 contains an example of a simple service interface document. The highlighted fields will be referenced in the UDDI data entities.

Listing 1.

```xml
<?xml version="1.0"?>
<definitions name="StockQuoteService-interface"

targetNamespace="http://www.getquote.com/StockQuoteService-interface"

xmlns:tns="http://www.getquote.com/StockQuoteService-interface"

xmlns:xsd=" http://www.w3.org/2001/XMLSchema "

xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

xmlns="http://schemas.xmlsoap.org/wsdl/">

<documentation>
  Standard WSDL service interface definition for a stock quote service.
</documentation>

<message name="SingleSymbolRequest">
  <part name="symbol" type="xsd:string"/>
</message>

<message name="SingleSymbolQuoteResponse">
  <part name="quote" type="xsd:string"/>
</message>

<portType name="SingleSymbolStockQuoteService">
  <operation name="getQuote">
    <input message="tns:SingleSymbolRequest"/>
    <output message="tns:SingleSymbolQuoteResponse"/>
  </operation>
</portType>

<binding name="SingleSymbolBinding"

type="tns:SingleSymbolStockQuoteService">
  <soap:binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="getQuote">
    <input>
      <soap:body use="encoded"
        namespace="urn:single-symbol-stock-quotes"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </input>
    <output>
      <soap:body use="encoded"
        namespace="urn:single-symbol-stock-quotes"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </output>
  </operation>
</binding>
</definitions>
```

Publishing the UDDI tModel

The WSDL service interface description is published by the service interface provider. This document must be published before a service provider can publish a businessService that references the service interface. This service interface is published as a tModel, and will contain a reference to the WSDL service interface document.

Listing 2 contains the tModel that is created when the service interface listed above is published.
Listing 2.

```xml
<?xml version="1.0"?>
<tModel tModelKey="">
  <name>http://www.getquote.com/StockQuoteService-interface</name>
  
  <description xml:lang="en">
    Standard WSDL service interface definition for a stock quote service.
  </description>

  <overviewDoc>
    <description xml:lang="en">
      WSDL Service Interface Document
    </description>
    <overviewURL>
      http://www.getquote.com/services/SQS-interface.wsdl#SingleSymbolBinding
    </overviewURL>
  </overviewDoc>

  <categoryBag>
    <keyedReference tModelKey="UUID:C1ACF26D-9672-4404-9D70-39B756E62AB4" keyValue="wsdlSpec" />
    <keyedReference tModelKey="UUID:DB77450D-9FA8-45D4-A7BC-04411D14E384" keyValue="84121801" />
  </categoryBag>
</tModel>
```

Publishing the UDDI businessService

A service provider that implements this service interface can publish a businessService description that references the tModel that is associated with the service interface. The service description for the service implementation does not have to be specified using WSDL.

Listing 3.

```xml
<?xml version="1.0"?>
<businessService businessKey="" serviceKey="">
  <name>StockQuoteService</name>

  <description xml:lang="en">
    Stock Quote Service
  </description>

  <bindingTemplates>
    <bindingTemplate bindingKey="" serviceKey="">
      <description>
        Single Symbol Service
      </description>
      <accessPoint URLType="http">
        http://www.getquote.com/stockquoteservice
      </accessPoint>

      <tModelInstanceDetails>
        <tModelInstanceInfo tModelKey="[tModel Key for Service Interface]"></tModelInstanceInfo>
        <tModelInstanceDetails>
      </tModelInstanceDetails>
    </bindingTemplate>
  </bindingTemplates>

  <categoryBag>
    <keyedReference tModelKey="UUID:DB77450D-9FA8-45D4-A7BC-04411D14E384" keyValue="84121801" />
  </categoryBag>
</businessService>
```
Building a WSDL service implementation document

Since the UDDI businessService was not created from a WSDL service implementation document, a tool or runtime component that requires WSDL would have to build a WSDL document based on the contents of the businessService and the tModels that are referenced by its bindingTemplates.

Listing 4 contains a WSDL service implementation document that was created based on the contents of the UDDI businessService for this scenario. The values are shown in the diagram key.

Listing 4.

```xml
<?xml version="1.0"?>
<definitions name="StockQuoteService"
  xmlns:interface1="http://www.getquote.com/StockQuoteService-interface"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
  xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
  xmlns="http://schemas.xmlsoap.org/wsdl/">
  <import namespace="http://www.getquote.com/StockQuoteService-interface"
    location="http://www.getquote.com/wsdl/SQS-interface.wsdl"/>
  <service name="StockQuoteService">
    <documentation>Stock Quote Service</documentation>
    <port name="StockQuoteServicePort"
      binding="interface1:SingleSymbolBinding">
      <documentation>Single Symbol Service</documentation>
      <soap:address location="http://www.getquote.com/stockquoteservice"/>
    </port>
  </service>
</definitions>
```

Key:

- **Definitions** - The name attribute for the definitions element is set from the businessService name. The XML namespace references for XML Schema, WSDL, SOAP binding, HTTP binding, and MIME binding are added to the definitions element.
- **Import** - One import element is created for each tModel that references a WSDL service interface document. The tModel key is used to get the tModel details. The tModel contains the overviewURL which references the WSDL service interface document. The namespace attribute is set using the targetNamespace in the WSDL service interface document. The location attribute is set using the value from the overviewURL within the tModel that is associated with the service interface definition. For each import element, an XML namespace reference is added to the definitions element using the targetNamespace from the service interface document.
- **Service** - The name attribute for the service element is set from the businessService name. Since the service names is a NCName, the businessService name may have to be modified to get a proper WSDL service name. The documentation element within the service element is set from the businessService description.
• **Port** - A port element is created for each bindingTemplate in the businessService that references a service that is described using WSDL. The name attribute for the port element is set to the service name with the string "Port" appended. To set the binding attribute, the binding reference is obtained from the overviewURL in the tModel. If the binding is not specified on the overviewURL, then the first binding in the WSDL service interface is used to set the binding name. Since the binding attribute is a QName, the binding name is prefixed with the XML namespace specification for the service interface reference. If the bindingTemplate contains a description, then it is used to set the documentation element within the port element. The extension element within the port element is created based on the binding type. For a SOAP binding, a soap:address element is used. The location attribute for this extension element is set from the accessPoint within the bindingTemplate.

**Scenario 2: Service implementation with one service interface document**

A service interface can be developed by one person, and then implemented and referenced by a different person. For the set of tools and runtime components that operate only on WSDL service descriptions, the WSDL service implementation document will contain a reference to the WSDL service interface document. In this scenario, the service implementation document references only one WSDL service interface document.

**WSDL service interface definition**

The service interface definition for this scenario is consistent with the one described in the UDDI best practices document. The service interface document contains all of the WSDL types, message, portType and binding elements.

Listing 5 contains an example of a simple service interface document. The service that is described in this document contains one operation, and the binding is specified using SOAP. The highlighted fields will be referenced in the UDDI data entities.

**Listing 5.**

```xml
<?xml version="1.0"?>
definitions name="StockQuoteService-interface"
    targetNamespace="http://www.getquote.com/StockQuoteService-interface"
    xmlns:tns="http://www.getquote.com/StockQuoteService-interface"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
    xmlns="http://schemas.xmlsoap.org/wsdl/">
    
    <documentation>
        Standard service interface definition for a stock quote service.
    </documentation>

    <message name="SingleSymbolRequest">
        <part name="symbol" type="xsd:string"/>
    </message>

    <message name="SingleSymbolQuoteResponse">
        <part name="quote" type="xsd:string"/>
    </message>
```
<portType name="SingleSymbolStockQuoteService">
  <operation name="getQuote">
    <input message="tns:SingleSymbolRequest"/>
    <output message="tns:SingleSymbolQuoteResponse"/>
  </operation>
</portType>

<binding name="SingleSymbolBinding" type="tns:SingleSymbolStockQuoteService">
  <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="getQuote">
    <soap:operation soapAction="http://www.getquote.com/GetQuote"/>
    <input>
      <soap:body use="encoded"
        namespace="urn:single-symbol-stock-quotes"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </input>
    <output>
      <soap:body use="encoded"
        namespace="urn:single-symbol-stock-quotes"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </output>
  </operation>
</binding>
</definitions>

Publishing the UDDI tModel

The service interface provider must publish the service interface description before the service implementation description can be published. This service interface is published as a tModel. The tModel will contain a reference to the WSDL service interface document.

Listing 6 contains the tModel that is created when the service interface listed above is published.

Listing 6.

```xml
<?xml version="1.0"?>
<tModel tModelKey="">
  <name>http://www.getquote.com/StockQuoteService-interface</name>
  <description xml:lang="en">
    Standard service interface definition for a stock quote service.
  </description>

  <overviewDoc>
    <description xml:lang="en">
      WSDL Service Interface Document
    </description>
    <overviewURL>
      http://www.getquote.com/services/SQS-interface.wsdl#SingleSymbolBinding
    </overviewURL>
  </overviewDoc>

  <categoryBag>
    <keyedReference tModelKey="UUID:C1ACF26D-9672-4404-9D70-398756E62AB4"
      keyName="uddi-org:types" keyValue="wsdlSpec"/>
    <keyedReference tModelKey="UUID:DB77456D-9FA8-45D4-A7BC-04411D14E384"
      keyName="Stock market trading services" keyValue="84121801"/>
  </categoryBag>
</tModel>
```
WSDL service implementation definition

The service implementation document contains one import element that references the service interface document, as well as a service element that contains a reference to the location of the Web service. Listing 7 contains an example of this type of WSDL document.

Listing 7.

```xml
<?xml version="1.0"?>
<definitions name="StockQuoteService"
    targetNamespace="http://www.getquote.com/StockQuoteService"
    xmlns:interface="http://www.getquote.com/StockQuoteService-interface"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
    xmlns="http://schemas.xmlsoap.org/wsdl/">
    <import namespace="http://www.getquote.com/StockQuoteService-interface"
        location="http://www.getquote.com/wsdl/SQS-interface.wsdl"/>
    <service name="StockQuoteService">
        <documentation>Stock Quote Service</documentation>
        <port name="SingleSymbolService"
            binding="interface:SingleSymbolBinding">
            <soap:address location="http://www.getquote.com/stockquoteservice"/>
        </port>
    </service>
</definitions>
```

Publishing the UDDI businessService

The WSDL service implementation description is published by the service provider. The service implementation is published as a businessService. The tModel associated with the service interface is referenced by the bindingTemplate within the businessService.

This method for publishing a service implementation is not defined in the UDDI best practices document. "Publishing Service Implementations" on page 7 of the best practices document contains a description of how the businessService is created from a WSDL service implementation document.

Listing 8 contains an example of a businessService that is created from the service implementation listed above.

Listing 8.

```xml
<?xml version="1.0"?>
<businessService businessKey="" serviceKey="">
    <name>StockQuoteService</name>
    <description xml:lang="en">
        Stock Quote Service
    </description>
    <bindingTemplates>
        <bindingTemplate bindingKey="" serviceKey="">
            <accessPoint URLType="http">
                http://www.getquote.com/stockquoteservice
            </accessPoint>
        </bindingTemplate>
    </bindingTemplates>
</businessService>
```
Scenario 3: Service implementation with multiple service interface documents

A service provider may decide to develop a Web service that implements more than one service interface definition. The WSDL service implementation document for this Web service will contain references to all of the service interface documents. Each reference to a service interface document is represented by an import element in the service implementation document.

WSDL service interface definition

For this scenario, there are two different service interfaces. These service interfaces could be published by the same service interface provider, or by different service interface providers. Each service interface document contains a different target namespace, port type, and binding. Also, each SOAP binding references a different SOAP service.

The following figure contains the service interface definition for a service that requires a single stock symbol as an input message, and a single stock quote as a response.

Listing 9.

```xml
<?xml version="1.0"?>
<definitions name="StockQuoteService-SingleSymbol-interface"

targetNamespace="http://www.getquote.com/StockQuoteService-SingleSymbol-interface"
xmns:xsd="http://www.w3.org/2001/XMLSchema"
xmns:soap="http://schemas.xmlsoap.org/wsd1/soap/"
xmns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmns="http://schemas.xmlsoap.org/wsd1/"/>

<documentation>
Standard service interface definition for a stock quote service that
has only one symbol as an input parameter.
</documentation>

<message name="SingleSymbolRequest">
<part name="symbol" type="xsd:string"/>
</message>
```
Listing 10 contains a service interface definition that can have multiple stock symbols as an input message and multiple stock quotes as a response. This service interface contains its own message, portType and binding specifications.

Listing 10.

```xml
<?xml version="1.0"?>
<definitions name="StockQuoteService-MultSymbol-interface"
    targetNamespace="http://www.getquote.com/StockQuoteService-MultSymbol-interface"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsd1="http://www.getquote.com/StockQuoteService/schema"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/
    xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/
    xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd11="http://www.getquote.com/StockQuoteService/schema">
    <documentation>
        Standard service interface definition for a stock quote service that can receive a request that contains multiple symbols.
    </documentation>
    <types>
        <schema targetNamespace="http://www.getquote.com/StockQuoteService/schema"
            xmlns="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/
            xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/
            xmlns:xsn="http://www.w3.org/2001/XMLSchema-instance"
            xmlns:xsd1="http://www.getquote.com/StockQuoteService/schema">
            <complexType name="ArrayOfString">
                <complexContent>
                    <restriction base="soapenc:Array">
                        <attribute ref="soapenc:arrayType"
                            name="soapenc:ArrayType"
                            namespace="http://www.getquote.com/StockQuoteService/schema">
                            <enumeration value="xsd:string"/>
                        </attribute>
                    </restriction>
                </complexContent>
            </complexType>
        </schema>
    </types>
</definitions>
```
Publishing the UDDI tModel

Both WSDL service interfaces for this scenario will be published in a UDDI registry as separate tModels. Since each service interface contains only one binding, the overviewURL does not need to reference a specific binding. Listing 11 contains the tModel that describes the first service interface definition.

Listing 11.

```xml
<?xml version="1.0"?
<tModel tModelKey="">
  <name>http://www.getquote.com/StockQuoteService-SingleSymbol-interface</name>

  <description xml:lang="en">
    Standard service interface definition for a stock quote service that has only one symbol as an input parameter.
  </description>

  <overviewDoc>
    <description xml:lang="en">
      WSDL Service Interface Document
    </description>
    <overviewURL>
      http://www.getquote.com/services/SQS-SingleSymbol-interface.wsdl
    </overviewURL>
  </overviewDoc>
</tModel>
```
Listing 12 contains the tModel for the second service interface definition

Listing 12.

```xml
<definitions name="StockQuoteService"

targetNamespace="http://www.getquote.com/StockQuoteService"


xmlns:xsd="http://www.w3.org/2001/XMLSchema"

xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

xmlns="http://schemas.xmlsoap.org/wsdl/"/>

<import

namespace="http://www.getquote.com/StockQuoteService-SingleSymbol-interface"


<import

namespace="http://www.getquote.com/StockQuoteService-MultSymbol-interface"

```
Publishing the UDDI businessService

For this scenario, the service provider develops a service that implements both service interfaces. The service implementation description is published by the service provider as a UDDI businessService. Since a service implementation references multiple service interfaces using separate import elements, each bindingTemplate within the businessService will reference a different tModel.

This method for publishing a service implementation is not defined in the UDDI best practices document. "Publishing Service Implementations" on page 7 of the best practices document contains a description of how the businessService is created from a WSDL service implementation document.

Listing 14 contains an example of a businessService that is created from the service implementation listed above

Listing 14.

```xml
<?xml version="1.0"?>
<businessService businessKey="" serviceKey=""
  name="StockQuoteService"/>

<description xml:lang="en">
  Stock Quote Service
</description>

<bindingTemplates>
  <bindingTemplate bindingKey="" serviceKey=""
    accessPoint URLType="http">
    http://www.getquote.com/stockquoteservice
  </accessPoint>
  <tModelInstanceDetails>
    <tModelInstanceInfo tModelKey="[tModel Key for Service Interface 1]">
      <instanceDetails>
        <overviewURL>
          http://www.getquote.com/services/SQS.wsdl#SingleSymbolService
        </overviewURL>
      </instanceDetails>
    </tModelInstanceInfo>
  </tModelInstanceDetails>
</bindingTemplate>
```
Scenario 4: A single WSDL document

If the service interface provider and the service provider is the same person, then one WSDL document can be used to define a complete WSDL service description. The purpose of this scenario is to show that a UDDI businessService and tModel can reference the same document from their respective overviewURL elements.

The single WSDL document is published as both a UDDI tModel and businessService. Both of these data entities will contain a reference to the same WSDL document.

Single WSDL service definition

Listing 15 contains an example of a single WSDL document that contains a complete service definition. This document contains all of the WSDL elements that are required to fully describe a web service. This is accomplished by combining the service interface definition and the service implementation definition into one WSDL document.

Listing 15..

```xml
<?xml version="1.0"?>
<definitions name="StockQuoteService"
    targetNamespace="http://www.getquote.com/StockQuoteService"
    xmlns:tns="http://www.getquote.com/StockQuoteService"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:soap="http://schemas.xmlsoap.org/soap/"/>
    xmlns="http://schemas.xmlsoap.org/soap/">

  <documentation>
      Service definition for a complete stock quote service.
  </documentation>

  <message name="SingleSymbolRequest">
      <part name="symbol" type="xsd:string"/>
  </message>
```
Publishing the UDDI tModel

When the UDDI tModel is published, it will contain a reference to the WSDL document that contains the complete service description. The tModel is created using the same process that is defined for a WSDL service interface definition. Although the overviewURL will contain a reference to a complete WSDL document, the tModel still refers only to the bindings in this document. This means that the tModel does not refer to the service and port elements.

Listing 16.

```xml
<?xml version="1.0"?>
<definitions>

<message name="SingleSymbolQuoteResponse">
  <part name="quote" type="xsd:string"/>
</message>

<portType name="SingleSymbolStockQuoteService">
  <operation name="getQuote">
    <input message="tns:SingleSymbolRequest"/>
    <output message="tns:SingleSymbolQuoteResponse"/>
  </operation>
</portType>

<binding name="SingleSymbolBinding">
  <soap:binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="getQuote">
    <soap:operation soapAction="http://www.getquote.com/GetQuote"/>
    <input>
      <soap:body use="encoded"
        namespace="urn:single-stock-quotes"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </input>
    <output>
      <soap:body use="encoded"
        namespace="urn:single-stock-quotes"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </output>
  </operation>
</binding>

<service name="StockQuoteService">
  <documentation>Stock Quote Service</documentation>
  <port name="SingleSymbolService" binding="tns:SingleSymbolBinding">
    <soap:address location="http://www.getquote.com/stockquoteservice"/>
  </port>
</service>
</definitions>
```
Publishing the UDDI businessService

The UDDI businessService is created from the complete WSDL service description using the same process that is used for a WSDL service implementation document.

Listing 17.

```xml
<?xml version="1.0"?>
<businessService businessKey="" serviceKey="">
  <name>StockQuoteService</name>
  <description xml:lang="en">
    Stock Quote Service
  </description>
  <bindingTemplates>
    <bindingTemplate bindingKey="" serviceKey="">
      <accessPoint URLType="http">
        http://www.getquote.com/stockquoteservice
      </accessPoint>
      <tModelInstanceDetails>
        <tModelInstanceInfo tModelKey="[tModel Key for Service Interface]">
          <instanceDetails>
            <overviewURL>
              http://www.getquote.com/services/SQS.wsdl#SingleSymbolService
            </overviewURL>
          </instanceDetails>
        </tModelInstanceInfo>
      </tModelInstanceDetails>
    </bindingTemplate>
  </bindingTemplates>
  <categoryBag>
    <keyedReference tModelKey="UUID:C1ACF26D-9672-4404-9D70-39B756E62AB4" keyName="uddi-org:types" keyValue="wsdlSpec"/>
    <keyedReference tModelKey="UUID:DB77450D-9FA8-45D4-A7BC-04411D14E384" keyName="Stock market trading services" keyValue="84121801"/>
  </categoryBag>
</businessService>
```

Scenario 5: A service interface that references another service interface

A service interface provider can develop a service interface that contains only the types, message and portType elements. This service interface can then be used by other service interface providers to specify specific bindings for the service interface. Only the service interface that
contains the bindings can be published as a UDDI tModel. The WSDL service interface that contains only the types, message and portType elements, can not be published as a UDDI tModel. A UDDI tModel can only refer to a WSDL service interface that contains at least one binding element.

**WSDL service interface definition with messages and portTypes**

The service interface provider develops the service interface that contains only the messages and portTypes. A service interface does not have to contain a binding specification.

**Listing 18.**

```xml
<?xml version='1.0'?>
<definitions name='StockQuoteService-interface'
    targetNamespace='http://www.getquote.com/StockQuoteService-interface'
    xmlns:tns='http://www.getquote.com/StockQuoteService-interface'
    xmlns:xsd='http://www.w3.org/2001/XMLSchema'
    xmlns:soap='http://schemas.xmlsoap.org/wsdl/soap/
    xmlns='http://schemas.xmlsoap.org/wsdl/'>

    <documentation>
    Standard service interface definition for a stock quote service without any bindings.
    </documentation>

    <message name='SingleSymbolRequest'>
        <part name='symbol' type='xsd:string'/>
    </message>

    <message name='SingleSymbolQuoteResponse'>
        <part name='quote' type='xsd:string'/>
    </message>

    <portType name='SingleSymbolStockQuoteService'>
        <operation name='getQuote'>
            <input message='tns:SingleSymbolRequest'/>
            <output message='tns:SingleSymbolQuoteResponse'/>
        </operation>
    </portType>

</definitions>
```

**WSDL service interface definition with bindings**

A service provider can implement an existing service interface definition that contains only the message and portType elements. The implementation will require a specific binding, which is specified by the service provider. This binding specification is put into a service interface document, which will also contain a reference to the original service interface document.

**Listing 19.**

```xml
<?xml version='1.0'?>
<definitions name='StockQuoteService-binding'
    targetNamespace='http://www.getquote.com/StockQuoteService-binding'
    xmlns:interface='http://www.getquote.com/StockQuoteService-interface'
    xmlns:xsd='http://www.w3.org/2001/XMLSchema'
    xmlns:soap='http://schemas.xmlsoap.org/wsdl/soap/
    xmlns='http://schemas.xmlsoap.org/wsdl/'>

```

Understanding WSDL in a UDDI registry, Part 2
<documentation>
Service interface binding definition for a stock quote service.
</documentation>

<import namespace="http://www.getquote.com/StockQuoteService-interface/
location="http://www.getquote.com/SQS-interface.wsdl">

<binding name="SingleSymbolBinding"

type="interface:SingleSymbolStockQuoteService">

<soap:binding style="rpc"
transport="http://schemas.xmlsoap.org/soap/http"/>

<operation name="getQuote">

<soap:operation soapAction="http://www.getquote.com/GetQuote"/>

<input>

<soap:body use="encoded"
namespace="urn:single-symbol-stock-quotes"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>

</input>

<output>

<soap:body use="encoded"
namespace="urn:single-symbol-stock-quotes"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>

</output>

</operation>

</binding>

</definitions>

Publish the UDDI tModel for service interface with bindings

The tModel for the service interface with the bindings can be published as a tModel. This tModel will reference only the WSDL service interface document that contains the binding definitions. This document must be retrieved to determine the document that contains the remainder of the service interface definition. The complete service interface definition is determined by following the import statements in the original service interface document.

Listing 20 contains the UDDI tModel that is created from the WSDL service interface definition that only contains the binding definition.

Listing 20.

<?xml version="1.0"?>
<tModel tModelKey="">

<name>http://www.getquote.com/StockQuoteService-binding</name>

<description xml:lang="en">
Service interface binding definition for a stock quote service.
</description>

<overviewDoc>
<description xml:lang="en">
WSDL Service Interface Document
</description>
<overviewURL>
http://www.getquote.com/services/SQS-binding.wsdl
</overviewURL>
</overviewDoc>

<categoryBag>

<keyedReference tModelKey="UUID:C1ACF26D-9672-4404-9D70-39B756E62AB4"
keyName="uddi-org:types" keyValue="wsdlSpec"/>

<keyedReference tModelKey="UUID:DB77458D-9FA8-45D4-A7BC-84411D14E384"
WSDL service implementation definition

The WSDL service implementation document contains an import reference to just the WSDL service interface document that contains the binding definitions. WSDL tools or runtime environments that operate on this service definition must be able to follow the complete import chain. This document will import the WSDL document that contains the binding elements, which then imports the WSDL document that contains the types, message and portType elements. All three of these documents together provide a complete WSDL service definition.

Listing 21.

```xml
<?xml version="1.0"?>
  <service name="StockQuoteService">
    <documentation>Stock Quote Service</documentation>
    <port name="SingleSymbolService" binding="binding:SingleSymbolBinding">
      <soap:address location="http://www.getquote.com/stockquoteservice"/>
    </port>
  </service>
</definitions>
```

Publish the UDDI businessService

The WSDL service implementation definition is published as a UDDI businessService. Each of the tModels will be referenced using a tModelInstanceInfo element. Both tModels are reference so that search operations can be performed using either tModel.

The import chain must be traversed to determine the number of tModelInstanceInfo elements that are required. For this scenario, there are only two service interface documents in the import chain, so only two tModelInstanceInfo elements will be specified in the bindingTemplate.

The first tModelInstanceInfo will contain a reference to the service interface document that contains the binding definitions. This tModelInstanceInfo will also contain an overviewURL that will reference the WSDL service implementation document.

The second tModelInstanceInfo element will reference the tModel associated with the service interface that contains the message and portType elements. This tModelInstanceInfo will not contain an overviewURL element.
Listing 22.

```xml
<?xml version="1.0"?>
<businessService businessKey="" serviceKey="">
   <name>StockQuoteService</name>
   
   <description xml:lang="en">
      Stock Quote Service
   </description>

   <bindingTemplates>
      <bindingTemplate bindingKey="" serviceKey="">
         <accessPoint URLType="http">
            http://www.getquote.com/stockquoteservice
         </accessPoint>
         
         <tModelInstanceDetails>
            <tModelInstanceInfo tModelKey="[tModel Key for binding Service Interface]">
               <instanceDetails>
                  <overviewURL>
                     http://www.getquote.com/services/SQS.wsdl#SingleSymbolService
                  </overviewURL>
               </instanceDetails>
            </tModelInstanceInfo>
         </tModelInstanceDetails>
      </bindingTemplate>
   </bindingTemplates>

   <categoryBag>
      <keyedReference tModelKey="UUID:DB77450D-9FA8-45D4-A7BC-04411D14E384"
                      keyName="Stock market trading services"
                      keyValue="84121801"/>
   </categoryBag>
</businessService>
```

Summary

In this article I reviewed several WSDL usage scenarios that showed how to publish different types of WSDL service descriptions in a UDDI registry. These usage scenarios provided an XML based mapping from WSDL into the appropriate UDDI data entities. The next part of this series will show how to use the UDDI for Java API (UDDI4J) and the Web Services Description Language for Java Toolkit (WSDL4J) to programmatically publish a WSDL service description in a UDDI registry. Both of these packages are open source projects that can be accessed from the Resources section.

© Copyright IBM Corporation 2001
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)