IBM

Understanding WSDL in a UDDI Registry

How to Publish and Find
WSDL Service Descriptions

Version 1.0

August 20, 2001

Peter Brittenham
Francisco Curbera
David Ehnebuske
Steve Graham

Table of Contents

Figures

Introduction

The Web Services Description Language (WSDL) is an XML language for describing Web services as a set of network endpoints that operate on messages. A WSDL service description contains an abstract definition for a set of operations and messages, a concrete protocol binding for these operations and messages, and a network endpoint specification for the binding.

Universal Description Discovery and Integration (UDDI) provides a method for publishing and finding service descriptions. The UDDI data entities provide support for defining both business and service information. The service description information defined in WSDL is complementary to the information found in a UDDI registry. UDDI provides support for many different types of service descriptions. As a result, UDDI has no direct support for WSDL or any other service description mechanism.

The UDDI organization has published a best practices document titled *Using WSDL in a UDDI Registry 1.05*. This best practices document describes how to publish WSDL service descriptions in a UDDI Registry. All of the information contained in this document adheres to the procedures outlined in the best practices document.

The purpose of this document is to augment the information in the best practices document. This document focuses on how to map a complete WSDL service description into a UDDI registry. Complete WSDL service descriptions are required by existing WSDL tooling and runtime support.

This document contains three sections. The first section provides a brief overview of the UDDI data types, and the two types of WSDL documents. The second section provides a description of how to map WSDL service descriptions into a UDDI registry. The last section contains a set of usage scenarios that provide examples of how to publish and find WSDL service descriptions in a UDDI registry.

The information in this document is consistent with the WSDL V1.1 specification, and the UDDI V1.0 and V2.0 specifications. Refer to the References section on page 32 for the location of these specifications.

Overview

Before describing the process for mapping WSDL service descriptions into a UDDI registry, it is important to understand the UDDI data types and the primary WSDL document types.

UDDI Data Types

There are four primary data types in a UDDI registry: *businessEntity, businessService, bindingTemplate*, and *tModel*. Figure 1 shows the relationship between all of these data types.

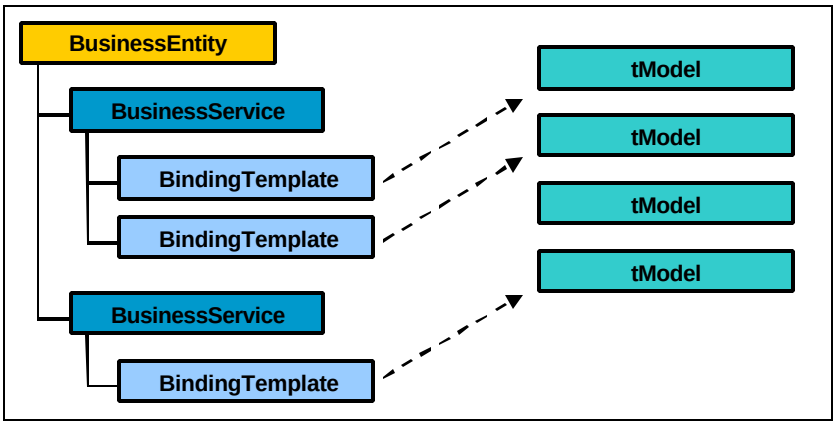

Figure 1. UDDI Data Types

The businessEntity provides information about a business, and can contain one or more businessServices. The business is the service provider. The technical and business descriptions for a Web service are defined in a businessService and its bindingTemplates. Each bindingTemplate contains a reference to one or more tModels. A tModel is used to define the technical specification for a service.

WSDL Document Types

To assist with publishing and finding WSDL service descriptions in a UDDI Registry, WSDL documents are divided into two types: *service interfaces* and *service implementations*.

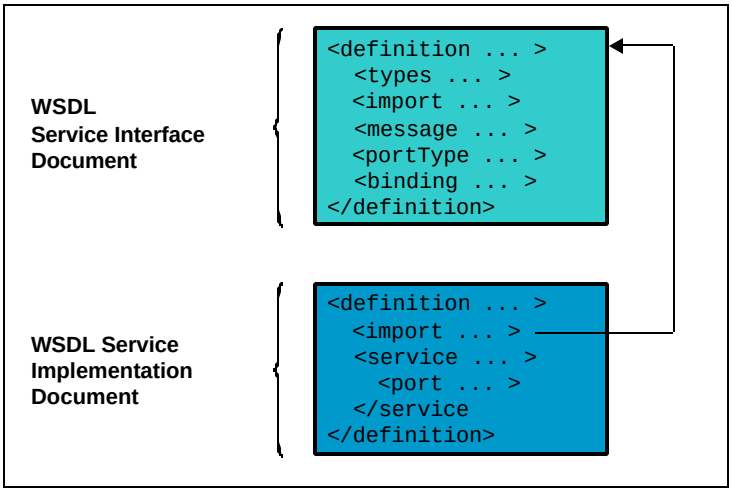

Figure 2. WSDL Document Types

A service interface is described by a WSDL document that contains the *types*, *import*, *message*, *portType*, and *binding* elements. A service interface contains the WSDL service definition that will be used to implement one or more services. It is an abstract definition of a web service, and is used to describe a specific type of service.

A service interface document can reference another service interface document using an import element. For example, a service interface that contains only the message and portType elements can be referenced by another service interface that contains only bindings for the portType.

The WSDL service implementation document will contain the *import* and *service* elements. A service implementation document contains a description of a service that implements a service interface. At least one of the import elements will contain a reference to the WSDL service interface document. A service implementation document can contain references to more than one service interface document.

The import element in a WSDL service implementation document contains two attributes. The *namespace* attribute value is a URI that matches the *targetNamespace* in the service interface document. The *location* attribute is a URL that is used to reference the WSDL document that contains the complete service interface definition. The *binding* attribute on the port element contains a reference to a specific binding in the service interface document.

The service interface document is developed and published by the *Service Interface Provider*. The service implementation document is created and published by the *Service Provider*. The roles of the service interface provider and service provider are logically separate, but they can be the same business entity.

Publishing and Finding WSDL Service Descriptions

This section describes the process for publishing and finding a complete WSDL service description. A complete WSDL service description is a combination of a service interface and a service implementation document.

Since the service interface represents a reusable definition of a service, it is published in a UDDI registry as a tModel. The service implementation describes instances of a service. Each instance is defined using a WSDL service element. Each service element in a service implementation document is used to publish a UDDI businessService.

When publishing a WSDL service description, a service interface must be published as a tModel before a service implementation is published as a businessService.

Figure 3 contains an overview of the mapping from WSDL to UDDI. This mapping will be described in the following sections.

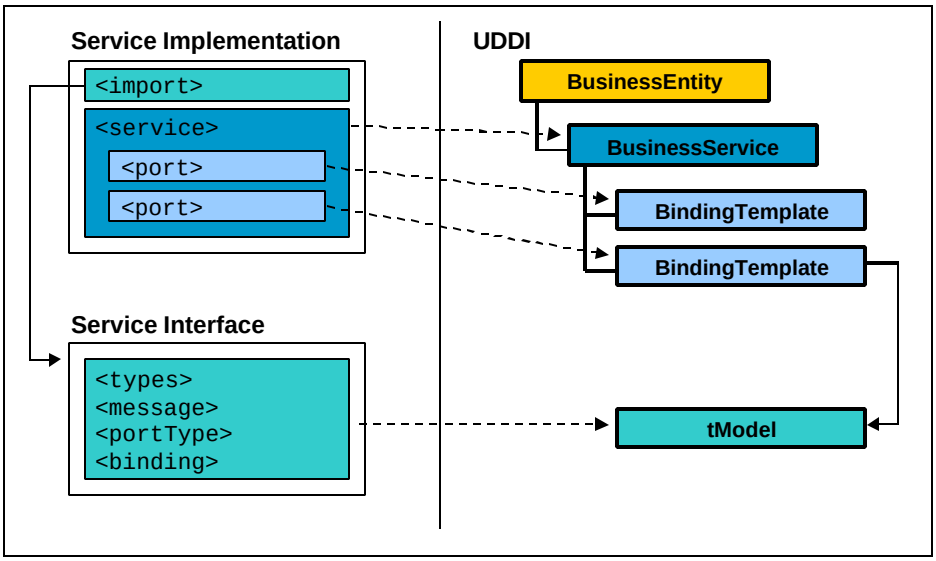

Figure 3. Overview of WSDL to UDDI Mapping

Publishing Service Interfaces

A service interface is published as a tModel in a UDDI registry. The tModel is published by the service interface provider. Some elements in the tModel are constructed using the information from the WSDL service interface description.

UDDI tModel

The following table defines the tModel creation steps. A valid tModel reference to a WSDL service interface definition should be named using the targetNamespace and must contain the overviewURL and categoryBag settings.

	UDDI tModel	WSDL Service Interface	Description	Required
1	name	targetNamespace attribute for definitions element	The tModel name is set using the target namespace for the service interface document. A consistent name is needed to ensure that the tModel can be located using just the information from a service implementation document.	Yes
2	description	documentation element within the definitions element	The tModel description element is restricted to 256 characters. The English value for this element can be set from the first 256 characters of the documentation element that is associated with the definitions element in the service interface document. If the documentation element does not exist, then the name attribute from the definitions element should be used.	No
3	overviewURL	[Service interface document URL and binding specification]	The location for the service interface document must be set in the overviewURL element. If there is more than one binding in the service interface document, then the binding must be encoded in the URL.	Yes
4	categoryBag	[Not applicable]	The categoryBag for the tModel must contain at least one keyed reference. This keyed reference must contain a reference to the **uddi-org:types** tModel and the key name must be **wsdlSpec**. This entry identifies this tModel as a WSDL service interface definition.	Yes

Example of Service Interface to tModel Mapping

Figure 4 contains an example of a WSDL service interface document. The values that are mapped into a UDDI tModel are highlighted.

- **Definitions**
  1. The targetNamespace will be used as the name for the tModel.
  2. The contents of the documentation element will be used for the description for the tModel.

- **Binding**
  1. The binding name will be used to qualify the overviewURL.

```
<?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">
      <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>
```

Figure 4. Example WSDL Service Interface

Figure 5 contains the UDDI tModel that is created when the WSDL service interface definition is published.

- **tModel**
  1. The tModel name is set from the targetNamespace.
  2. The description is set from the documentation element that is associated with the definitions element.
  3. The overviewURL is set to the network accessible location for the WSDL service interface document which is http://www.getquote.com/services/SQS-interface.wsdl. It also contains a direct reference to the binding named SingleSymbolBinding in the service interface document. Since this is the only binding in the service interface document, this reference to the binding is not required.
  4. The categoryBag contains the wsdlSpec entry, as well as any other keyedReferences which indicate the intended business use for this service interface description.

```
<?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-9672-4404-9D70-39B756E62AB4"
                    keyName="uddi-org:types" keyValue="wsdlSpec"/>
    <keyedReference tModelKey="UUID:DB77450D-9FA8-45D4-A7BC-04411D14E384"
                    keyName="Stock market trading services"
                    keyValue="84121801"/>
  </categoryBag>
</tModel>
```

Figure 5. UDDI tModel Created From WSDL Service Interface

Publishing Service Implementations

A service implementation is published in a UDDI registry as a businessService with one or more bindingTemplates. The businessService is published by the service provider.

UDDI businessService

A new businessService will be created for each service element that is defined in the service implementation document. The following table contains the list of businessService elements that can be created based on the contents of the WSDL service implementation document.

	UDDI businessService	Description	Required
1	name	The name element for the businessService is set from the name attribute from the service element in the service implementation document	Yes
2	description	The description element is set from the contents of the documentation element within a service element. The English value for the description element is set from the first 256 characters in the documentation element that is associated with the service element. If a documentation element doesn't exist, then the description element for the businessService is not set.	No

UDDI bindingTemplate

A new bindingTemplate element is created within a businessService for each port element that is defined within a service element.

	UDDI bindingTemplate	Description	Required
1	description	If the port element contains a documentation element, then one description element is set from the first 256 characters of the documentation element.	No
2	accessPoint	For a SOAP or HTTP binding, the accessPoint is set from the location attribute on the extension element that is associated with the port element. The element will contain the URL, and the URLType attribute is set based on the protocol in this URL. For protocol bindings that don't use a URL specification, the URLType attribute should be used to indicate that type of protocol binding and the accessPoint element should contain a value that could be used to locate the Web service using the specified protocol.	Yes
3	tModelInstanceInfo	The bindingTemplate will contain one tModelInstanceInfo element for each tModel that it references. There will be at least one tModelInstanceInfo element which will contain a direct reference to the tModel that represents the service interface document.	Yes
4	overviewURL	The overviewURL element may contain a direct reference to the service implementation document. The reference to this document is used only to provide access to human readable documentation. All of the other information within this document should be accessible through a UDDI data entity. By maintaining a direct reference to the original WSDL document, you are assured the document that is published is the same one that it is returned during a find operation. If this document contains more than one port, then this element should contain a direct reference to the port name. It is not sufficient to use the binding reference in the tModel, since there can be more than one port that references the same binding. The port name is specified as a fragment identifier on the overviewURL. The fragment identifier is an extension to the URL using a '#' character as a separator.	No

Example of Service Implementation to businessService Mapping

Figure 6 contains an example of a WSDL service implementation document. The highlighted values are required when mapping the WSDL information to a UDDI businessService and bindingTemplates.

- **Service**
  1. The name attribute for the service element is used as the name for the businessService.
  2. The documentation element within the service element is used for the description of the businessService.

- **Import**
  1. The port name is appended to the overviewURL, which contains the reference to the service implementation document.
  2. The location of the service is used to set the accessPoint in the bindingTemplate.

```
<?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/">

  <documentation>
    This service provides an implementation of a standard stock quote service.
    The Web service uses the live stock quote service provided by XMLtoday.com.
    The XMLtoday.com stock quote service uses an HTTP GET interface to request
    a quote, and returns an XML string as a response.

    For additional information on how this service obtains stock quotes, go to
    the XMLtoday.com web site: http://www.xmltoday.com/examples/soap/stock.psp.
  </documentation>

  <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="SingleSymbolServicePort"
        binding="interface:SingleSymbolBinding">
      <documentation>Single Symbol Stock Quote Service</documentation>
      <soap:address location="http://www.getquote.com/stockquoteservice"/>
    </port>
  </service>
</definitions>
```

Figure 6. Example WSDL Service Implementation

Figure 7 contains the UDDI businessService definition that is created from the service implementation document. The categoryBag should contain one or more keyedReferences, which are used to categorize the intended business use for the service.

- **BusinessService**
  1. The businessService name is set from the service name in the WSDL service implementation document.
  2. The description is set from the documentation element within the service element.

- **BindingTemplate**
  1. The description is set from the documentation element within the port element.
  2. The accessPoint is set from the soap:address extension element.
  3. The tModelKey is set to the UUID for the tModel that is associated with the service interface document. This tModel can be located using the namespace attribute on the import element.
  4. The overviewURL is the location of the service implementation document. It does not contain a reference to the SingleServiceQuote port, since it is the only one in this document.

```
<businessService businessKey="..." serviceKey="...">
  <name>StockQuoteService</name>

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

  <bindingTemplates>
    <bindingTemplate bindingKey="..." serviceKey="...">
      <description>
        Single Symbol Stock Quote Service
      </description>
      <accesssPoint URLType="http">
        http://www.getquote.com/singlestockquote
      </accessPoint>
      <tModelInstanceDetails>
        <tModelInstanceInfo tModelKey="[tModel Key for Service Interface]">
          <instanceDetails>
            <overviewURL>
              http://www.getquote.com/services/SQS.wsdl
            </overviewURL>
          </instanceDetails>
        </tModelInstanceInfo>
      </tModelInstanceDetails>
    </bindingTemplate>
  </bindingTemplates>

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

Figure 7. UDDI Business Service Created From WSDL Service Implementation

Finding WSDL Service Interface Descriptions

All WSDL service interfaces are published in a UDDI registry as a tModel. Each of these tModels are categorized to identify them as a WSDL service description. The UDDI **find_tModel** message can be used to find tModels that have been categorized. Using the UDDI V1.0 API, the find_tModel message listed in Figure 8 can be used to locate all WSDL service interface descriptions.

```
<?xml version="1.0"?>
<find_tModel generic="1.0" xmlns="urn:uddi-org:api">
  <categoryBag>
    <keyedReference tModelKey="UUID:C1ACF26D-9672-4404-9D70-39B756E62AB4"
                    keyName="uddi-org:types" keyValue="wsdlSpec"/>
  </categoryBag>
</find_tModel>
```

Figure 8. Finding WSDL Service Interface Descriptions

The find_tModel message will return a list of tModel keys. A specific service interface description is retrieved using the **get_tModelDetail** message. The get_tModelDetail message will return a tModel such as the one listed in Figure 5 on page 7.

After a tModel has been retrieved, the overviewURL can be used to retrieve the contents of the WSDL service interface document.

Additional keyedReferences can be added to the categoryBag to limit the set of tModels that are returned in the response to this find request. The find_tModel message in Figure 9 can be used to locate all of the stock quote services that are defined using WSDL.

```
<?xml version="1.0"?>
<find_tModel generic="1.0" xmlns="urn:uddi-org:api">
  <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>
</find_tModel>
```

Figure 9. Finding WSDL Service Interfaces for an Intended Business Use

Finding WSDL Service Implementation Descriptions

A WSDL service implementation is published in a UDDI registry as a businessService. The businessService will contain one or more bindingTemplates. The businessService is classified to identify it as a WSDL based service description. Using the UDDI V1.0 API, a businessEntity or set of businessEntities must be found before the find API for a businessService can be used. Similarly, a businessService or set of businessServices must be found before using the find API to locate a bindingTemplate.

Finding a UDDI businessService

There are two basic methods that can be used to find a service description. The UDDI **find_service** message can be used to find service descriptions with a specific classification, or it can be used to find service descriptions that implement a specific service interface.

Using the UDDI V1.0 API, the message in Figure 10 can be issued for a specific businessEntity to locate the businessServices that are implementations of a stock quote service. Additional keyedReferences can be added to the categoryBag to narrow the scope of the service descriptions that are returned in the response to this message.

```
<?xml version="1.0"?>
<find_service businessKey="..." generic="1.0" xmlns="urn:uddi-org:api">
  <categoryBag>
    <keyedReference tModelKey="UUID:DB77450D-9FA8-45D4-A7BC-04411D14E384"
                    keyName="Stock market trading services"
                    keyValue="84121801"/>
  </categoryBag>
</find_service>
```

Figure 10. Finding WSDL Service Implementation Descriptions

The find_service message can also be used to locate businessServices that are implementations of a specific service interface. The message in Figure 11 contains an example of a find_service message that will find all businessServices, within a businessEntity, that implement the service interface for a stock quote service. Since a service interface is represented by a tModel, a tModelBag is used to specify the tModel key for the WSDL service interface associate with the stock quote service.

```
<?xml version="1.0"?>
<find_service businessKey="..." generic="1.0" xmlns="urn:uddi-org:api">
  <tModelBag>
    <tModelKey>[tModel key for WSDL service interface]</tModelKey>
  </tModelBag>
</find_service>
```

Figure 11. Finding WSDL Service Implementation Descriptions using a tModelBag

The find_service message will return a list of service keys. The businessService description can be retrieved using the **get_serviceDetail** message. The get_serviceDetail message will return a businessService such as the one listed in Figure 7 on page 10.

After a businessService has been retrieved, a specific bindingTemplate can be selected to invoke the Web service. The accessPoint within the bindingTemplate is the endpoint for the service. The overviewURL can be used to retrieve the contents of the WSDL service implementation document, which may contain additional details about the implemented service.

Finding a UDDI BindingTemplate

If the businessService contains more than one bindingTemplate it might be difficult to determine which bindingTemplate to use. The **find_binding** message can be used to locate the bindingTemplate that should be used.

Figure 12 contains a find_binding message that can be used to retrieve the bindingTemplates that reference a specific tModel. This tModel can be associated with a specific binding within a WSDL service interface description.

```
<?xml version="1.0"?>
<find_binding serviceKey="..." generic="1.0" xmlns="urn:uddi-org:api">
  <tModelBag>
    <tModelKey>[tModelKey for WSDL service interface]<tModelKey/>
  </tModelBag>
</find_service>
```

Figure 12. Find UDDI bindingTemplate

This message will return one or more bindingTemplates, such as the one in Figure 13. After accessing the bindingTemplate, the endpoint for the Web service is listed in the accessPoint. If the bindingTemplate was created from an existing WSDL service implementation document, then the overviewURL may contain a reference to this document. This document can be accessed to obtain additional, human readable information about the Web service that can not be found in the UDDI registry.

```
<?xml version="1.0"?>
<bindingTemplate bindingKey="" serviceKey="">
  <accesssPoint URLType="http">
    http://www.getquote.com/singlestockquote
  </accessPoint>
  <tModelInstanceDetails>
    <tModelInstanceInfo tModelKey="[tModel Key for Service Interface]">
      <instanceDetails>
        <overviewURL>
          http://www.getquote.com/services/SQS.wsdl
        </overviewURL>
      </instanceDetails>
    </tModelInstanceInfo>
  </tModelInstanceDetails>
</bindingTemplate>
```

Figure 13. UDDI bindingTemplate

Summary

This section has provided an overview of the process for publishing and finding WSDL service descriptions in a UDDI registry. Figure 14 provides a summary of the mapping from WSDL service interface and service implementation documents to UDDI registry entries. The remaining sections in this document provide usage scenarios that show how to apply the process for mapping WSDL into a UDDI registry.

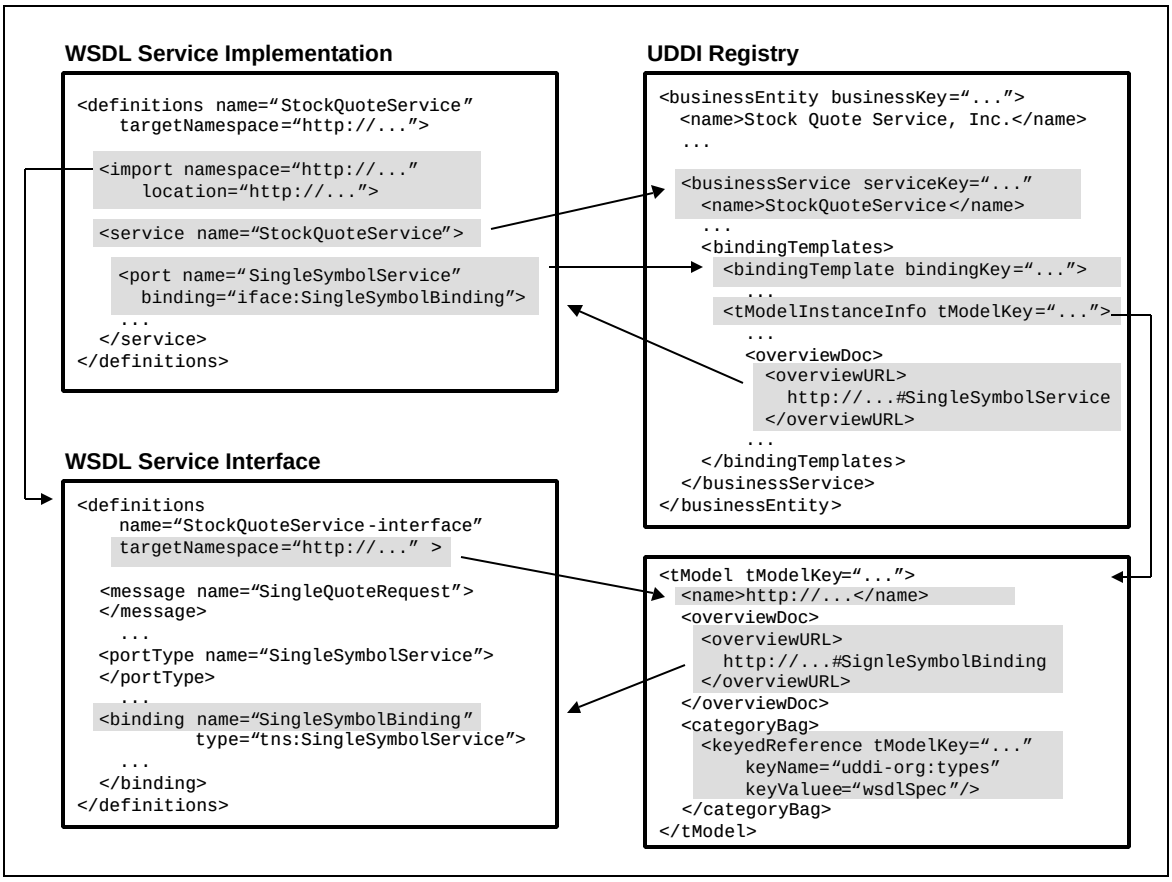

Figure 14. Summary of WSDL to UDDI Mapping

Usage Scenarios

As described in the first section, 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 section contains usage scenarios for publishing and finding complete WSDL service descriptions in a UDDI Registry. For each usage scenario, a method is described 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.

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.

Figure 15 contains an example of a simple service interface document. The highlighted fields will be referenced in the UDDI data entities.

```
<?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">
      <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>
```

Figure 15. Scenario 1: WSDL Service Interface Definition

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.

Figure 16 contains the tModel that is created when the service interface listed above is published.

```
<?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"
                    keyName="uddi-org:types" keyValue="wsdlSpec"/>
    <keyedReference tModelKey="UUID:DB77450D-9FA8-45D4-A7BC-04411D14E384"
                    keyName="Stock market trading services"
                    keyValue="84121801"/>
  </categoryBag>
</tModel>
```

Figure 16. Scenario 1: UDDI 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.

```
<?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>
      <accesssPoint URLType="http">
        http://www.getquote.com/stockquoteservice
      </accessPoint>
      <tModelInstanceDetails>
        <tModelInstanceInfo tModelKey="[tModel Key for Service Interface]">
        </tModelInstanceInfo>
      </tModelInstanceDetails>
    </bindingTemplate>
  </bindingTemplates>

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

Figure 17. Scenario 1: UDDI 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.

- **Definitions**
  1. The name attribute for the definitions element is set from the businessService name.
  2. 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.
  1. The namespace attribute is set using the targetNamespace from the WSDL service interface document.
  2. The location attribute is set using the value from the overviewURL within the tModel that is associated with the service interface definition.
  3. For each import element, an XML namespace reference is added to the definitions element using the targetNamespace from the service interface document.

- **Service**
  1. 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.
  2. 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.
  1. The name attribute for the port element is set to the service name concatenated with the binding name. This will ensure a unique name for each port element.
  2. To set the binding attribute, the binding name is obtained from the reference on the overviewURL in the tModel. If the binding name 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.
  3. If the bindingTemplate contains a description, then it is used to set the documentation element within the port element.
  4. 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.

Figure 18 contains a WSDL service implementation document that was created based on the contents of the UDDI businessService for this scenario.

```
<?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="StockQuoteService_SingleSymbolBinding"
        binding="interface1:SingleSymbolBinding">
      <documentation>Single Symbol Service</documentation>
      <soap:address location="http://www.getquote.com/stockquoteservice"/>
    </port>
  </service>
</definitions>
```

Figure 18. Scenario 1: WSDL Service Implementation Created From UDDI businessService

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.

Figure 19 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.

```
<?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>
```

Figure 19. Scenario 2: WSDL Service Interface Document

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.

Figure 20 contains the tModel that is created when the service interface listed above is published.

```
<?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-9672-4404-9D70-39B756E62AB4"
                    keyName="uddi-org:types" keyValue="wsdlSpec"/>
    <keyedReference tModelKey="UUID:DB77450D-9FA8-45D4-A7BC-04411D14E384"
                    keyName="Stock market trading services"
                    keyValue="84121801"/>
  </categoryBag>
</tModel>
```

Figure 20. Scenario 2: UDDI 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. Figure 21 contains an example of this type of WSDL document.

```
<?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>
```

Figure 21. Scenario 2: WSDL Service Implementation Definition

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 contains a description of how the businessService is created from a WSDL service implementation document.

Figure 22 contains an example of a businessService that is created from the service implementation listed above.

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

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

  <bindingTemplates>
    <bindingTemplate bindingKey="" serviceKey="">
      <accesssPoint 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:DB77450D-9FA8-45D4-A7BC-04411D14E384"
                    keyName="Stock market trading services"
                    keyValue="84121801"/>
  </categoryBag>
</businessService>
```

Figure 22. Scenario 2: UDDI 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.

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

  <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>

  <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-symobl-stock-quotes"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
      </output>
    </operation>
  </binding>
</definitions>
```

Figure 23. Scenario 3: First WSDL Service Interface

Figure 24 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.

```
<?xml version="1.0"?>
<definitions name="StockQuoteService-MultSymbol-interface"
  targetNamespace="http://www.getquote.com/StockQuoteService-MultSymbol-
interface"
  xmlns:tns="http://www.getquote.com/StockQuoteService-MultSymbol-interface"
  xmlns:xsd=" http://www.w3.org/2001/XMLSchema "
  xmlns:xsd1="http://http://www.getquote.com/StockQuoteService/schema"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns="http://schemas.xmlsoap.org/wsdl/">

  <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:wsdl="http://schemas.xmlsoap.org/wsdl/">
      <complexType name="ArrayOfString">
        <complexContent>
          <restriction base="soapenc:Array">
            <attribute ref=soapenc:arrayType
                       wsdl:arrayType="xsd:string[]"/>
          </restriction>
        </complexContent>
      </compextType>
    </schemas>
  </types>

  <message name="MultSymbolRequest">
    <part name="symbols" type="xsd1:ArrayOfString"/>
  </message>
  <message name="MultSymbolQuoteResponse">
    <part name="quotes" type="xsd1:ArrayOfString"/>
  </message>

  <portType name="MultSymbolStockQuoteService">
    <operation name="getQuotes">
      <input message="tns:MultSymbolRequest"/>
      <output message="tns:MultSymbolQuoteResponse"/>
    </operation>
  </portType>

  <binding name="MultSymbolBinding"
           type="tns:MultSymbolStockQuoteService">
    <soap:binding style="rpc"
                  transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="getQuotes">
      <soap:operation soapAction="http://www.getquote.com/GetQuotes"/>
      <input>
        <soap:body use="encoded"
            namespace="urn:mult-symbol-stock-quotes"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
      </input>
      <output>
        <soap:body use="encoded"
            namespace="urn:mult-symbol-stock-quotes"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
      </output>
    </operation>
  </binding>
</definitions>
```

Figure 24. Scenario 3: Second WSDL Service Interface

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. Figure 25 contains the tModel that describes the first service interface definition.

```
<?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>

  <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>
</tModel>
```

Figure 25. Scenario 3: First UDDI tModel

Figure 26 contains the tModel for the second service interface definition.

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

  <description xml:lang="en">
    Standard service interface definition for a stock quote service that
    can receive a request that contains multiple symbols.
  </description>

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

  <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>
</tModel>
```

Figure 26. Scenario 3: Second UDDI tModel

WSDL Service Implementation Definition

The WSDL service implementation document for this scenario will contain two import elements. Each import element will reference one of the service interface documents. This document also contains at least two port elements. Each port element references a binding from within one of the service interface documents.

```
<definitions name="StockQuoteService"
  targetNamespace="http://www.getquote.com/StockQuoteService"
  xmlns:single="http://www.getquote.com/StockQuoteService-SingleSymbol-
interface"
  xmlns:mult="http://www.getquote.com/StockQuoteService-MultSymbol-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-SingleSymbol-interface"
   location="http://www.getquote.com/wsdl/SQS-SingleSymbol-interface.wsdl"/>

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

  <service name="StockQuoteService">
    <documentation>Stock Quote Service</documentation>

    <port name="SingleSymbolService"
        binding="single:SingleSymbolBinding">
      <soap:address location="http://www.getquote.com/stockquoteservice"/>
    </port>

    <port name="MultSymbolService"
        binding="mult:MultSymbolBinding">
      <soap:address location="http://www.getquote.com/stockquoteservice"/>
    </port>
  </service>
</definitions>
```

Figure 27. Scenario 3: WSDL Service Implementation

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 contains a description of how the businessService is created from a WSDL service implementation document.

Figure 28 contains an example of a businessService that is created from the service implementation listed above.

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

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

  <bindingTemplates>
    <bindingTemplate bindingKey="" serviceKey="">
      <accesssPoint 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>

    <bindingTemplate bindingKey="" serviceKey="">
      <accesssPoint URLType="http">
        http://www.getquote.com/stockquoteservice
      </accessPoint>
      <tModelInstanceDetails>
        <tModelInstanceInfo tModelKey="[tModel Key for Service Interface 2]">
          <instanceDetails>
            <overviewURL>
              http://www.getquote.com/services/SQS.wsdl#MultSymbolService
            </overviewURL>
          </instanceDetails>
        </tModelInstanceInfo>
      </tModelInstanceDetails>
    </bindingTemplate>
  </bindingTemplates>

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

Figure 28. Scenario 3: UDDI businessService

Scenario 4: 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

Figure 29 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.

```
<?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/wsdl/soap/"
  xmlns="http://schemas.xmlsoap.org/wsdl/">

  <documentation>
    Service definition for a complete 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-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>
```

Figure 29. Scenario 4: Complete WSDL Service Definition in a Single Document

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.

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

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

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

  <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>
</tModel>
```

Figure 30. Scenario 4: UDDI tModel

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.

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

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

  <bindingTemplates>
    <bindingTemplate bindingKey="" serviceKey="">
      <accesssPoint 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:DB77450D-9FA8-45D4-A7BC-04411D14E384"
                    keyName="Stock market trading services"
                    keyValue="84121801"/>
  </categoryBag>
</businessService>
```

Figure 31. Scenario 4: UDDI businessService

Scenario 5: 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.

```
<?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>
```

Figure 32. Scenario 5: WSDL Service Interface Definition with Messages and PortType

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

```
<?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/">

  <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>
```

Figure 33. Scenario 5: WSDL Service Interface Definition with Bindings

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.

Figure 34 contains the UDDI tModel that is created from the WSDL service interface definition that only contains the binding defintion.

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

Figure 34. Scenario 5: UDDI tModel

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.

```
<?xml version="1.0"?>
<definitions name="StockQuoteService"
  targetNamespace="http://www.getquote.com/StockQuoteService"
  xmlns:binding="http://www.getquote.com/StockQuoteService-binding"
  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-binding"
     location="http://www.getquote.com/wsdl/SQS-binding.wsdl"/>

  <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>
```

Figure 35. WSDL Service Implementation Definition

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.

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

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

  <bindingTemplates>
    <bindingTemplate bindingKey="" serviceKey="">
      <accesssPoint 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>
```

Figure 36. UDDI businessService

Appendix

References

This section contains a list of references for the information contained in this document.

1. Web Services Description Language (WSDL) 1.1, March 2001, http://www.w3c.org/TR/wsdl.

2. UDDI Programmer's API Specification Version 1.0, September 30, 2000, http://www.uddi.org/specification.html.

3. UDDI Data Structure Reference Version 1.0, June 8, 2001, http://www.uddi.org/specification.html.

4. UDDI Programmer's API Specification Version 2.0, June 8, 2001, http://www.uddi.org/specification.html.

5. Using WSDL in a UDDI Registry 1.05, June 25, 2001, http://www.uddi.org/bestpractices.html

IBM®

International Business Machines Corporation
Software Communications Department
Route 100, Building 1
Somers, NY 10589
U.S.A.

08-01