WSDL Technical Note

Document identifier:

wsdl-TN-V2.00-Draft-20020924

Location:

[http://tbd]

Editors (alphabetically):

John Colgrave, IBM colgrave@uk.ibm.com
Karsten Januszewski, Microsoft karstenj@microsoft.com

Contributors:

Abstract:

This document is an OASIS UDDI Technical Note that defines a new approach to using WSDL in a UDDI Registry.

Status:

This document is a working draft.

Committee members should send comments on this document to the *uddi-spec@lists.oasis-open.org* list. Others should subscribe to and send comments to the *uddi-spec-comment@lists.oasis-open.org* list. To subscribe, send an email message to *uddi-spec-comment-request@lists.oasis-open.org* with the word "subscribe" as the body of the message.

Copyright

Table of Contents

1 Introduction

The Universal Description Discovery and Integration (UDDI) specification provides a platform-independent way of describing and discovering Web services and Web services providers. The UDDI data structures provide a framework for the description of basic service information, and architects an extensible mechanism to provide detailed service access information using any standard description language. Many such languages exist in specific industry domains and at different levels of the protocol stack. The Web Services Description Language (WSDL) is a general purpose XML language for describing the interface, protocol bindings and the deployment details of network services. WSDL complements the UDDI standard by providing a uniform way of describing the abstract interface and protocol bindings of arbitrary network services. The purpose of this document is to clarify the relationship between the two, describe how WSDL can be used to help create UDDI business service descriptions.

The importance of mapping WSDL consistently and thoroughly is critical to the utility of UDDI.

1.1 Goals and Requirements

The primary goals of this new mapping are to represent sufficient information from the WSDL documents to allow the following types of queries without further recourse to the source WSDL documents, and to allow the appropriate WSDL documents to be retrieved once a match has been found. Given that the source WSDL documents can be distributed among the publishers using a UDDI registry, a UDDI registry provides a convenient central point where such queries can be executed.

This new best practice would satisfy the following types of queries for both design-time and run-time discovery:

1) Given the namesapce and/or local name of a wsdl:portType, find the tModel that represents that portType.
2) Given a tModel representing a portType, find all tModels representing bindings for that portType.
3) Given a tModel representing a portType, find all bindingTemplates that represent implementations of that portType.
4) Given a tModel representing a binding, find all bindingTemplates that represent implementations of that binding.
5) Find all bindingTemplates that represent implementations of an extended binding, for example all SOAP/HTTP implementations, of a portType.

Some aspects of the mapping allow information to be retrieved directly without further queries being necessary. For example, given the tModel representing a binding, it is possible to retrieve the key of the tModel representing the portType that the binding refers to. Other aspects of the mapping may require multiple queries to be issued to UDDI.

A further goal is to allow deployment WSDL to be generated. With this new mapping, it is possible to generate a wsdl:service element directly from the mapping in UDDI.

1.2 Relationship to Version 1 Best Practice

This document builds on *Using WSDL in a UDDI Registry, Version 1.07* [2], providing an expanded modeling practice which encompasses the flexibility of WSDL.

As a Technical Note, this document does not replace the existing Best Practice. If the additional flexibility is not required the existing Best Practice can continue to be used, particularly when the UDDI model is published manually.

It is anticipated that implementations of the approach described in this Technical Note will be developed and that once experience with those implementations is obtained this Technical Note will become a Best Practice.

A final goal is to be compatible with the existing Best Practice in that a model published using the approach described in this document should be usable by a client that uses the Version 1 Best Practice approach.

1.3 tModel Keys

The tModel keys presented in this draft document are fictitious. When this document is approved the real tModels will be published to the UDDI Business Registry and the tModel keys in this document will be updated to the real values.

2 Rationalizing Two Data Models: WSDL & UDDI

A brief discussion of the two respective data models, WSDL and UDDI, follows. For a complete explanation of the these specifications, see [1] and [3].

2.1 WSDL Data Model

A review of WSDL in the context of the goals and requirements will help guide a new mapping practice in UDDI.

2.1.1 portType

The central construct in WSDL is the portType. A portType is an abstract collection of operations that may be supported by one or more Web services. A WSDL portType defines these operations in terms of message definitions which usually rely on the XML Schema language to describe the representation of each message. A single WSDL file may contain multiple portType entities. Each portType is uniquely identified by the combination of its local name and the target namespace of the definitions element that contains the portType.

WSDL portTypes may be "implemented" by more than one Web service. Web services that purport to support a given portType must adhere not only to the message formats that are part of the WSDL definition, they must also adhere to the semantic agreement that is implicitly part of the portType. This allows applications to treat two web services as substitutable if and only if they implement a common portType.

2.1.2 binding

WSDL portTypes are defined independently from the transport protocol used to transmit the messages. To allow transport protocol details to be expressed in WSDL, one must define a second construct, known as a binding. A WSDL binding adds transport specifics to a particular WSDL portType. Again, a single WSDL file may contain multiple bindings. A WSDL binding specifies its portType through a QName reference. The portType implemented by a binding may or may not be in the same target namespace as the binding itself. Like a portType, a binding is uniquely identified by the combination of its local name and the target namespace of the definitions element that contains the binding.

2.1.3 service and port

Finally, WSDL defines a service as a collection of named ports, each of which is bound to a particular portType through a named binding. A service may expose multiple ports in order to make a single portType available over multiple protocols. Or, a service may expose multiple ports in order to expose more than one portType from a single logical entity. A WSDL port specifies the binding it implements through a QName reference.

2.1.4 import

The import directive in WSDL allows the separation of these different entities into multiple files. As such, a WSDL file may be composed of a single portType, multiple portTypes, a single binding that imports its portType definition, multiple bindings, a single service or multiple services, etc. The WSDL data model provides great flexibility in terms of composition and reusability of WSDL entities.

Given this, the critical components of a WSDL file in terms of composition and identity are the target namespace of the definitions element and the local names which identify each portType, binding, service and port within the target namespace.

2.2 UDDI Data Model

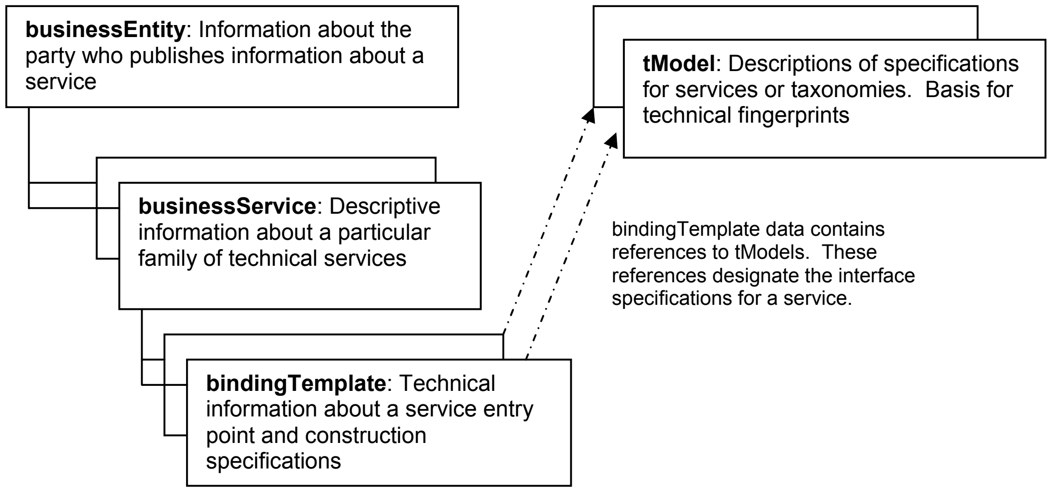
As an aid to understanding the sections ahead, we provide here a brief overview of two UDDI data structures that are particularly relevant to the use of WSDL in the context of a UDDI registry: the tModel, also known as the service type definition, and the businessService.

2.2.1 tModels

TModels represent unique concepts or constructs and provide the ability to describe compliance with a specification, a concept, or a shared design. TModels have various uses in the UDDI registry. In the case of mapping WSDL-described Web services, tModels are used to represent technical specifications like wire protocols, interchange formats and sequencing rules. When a particular specification is registered with the UDDI repository as a tModel, it is assigned a unique key, which is then used in the description of service instances to indicate compliance with the specification.

Each tModel contains an overviewURL which provides an address where the specification itself can be retrieved, for example, a WSDL file.

TModels can be decorated with searchable metadata by associating identifierBags and categoryBags with that tModel. These bags contain keyedReferences: name/value pairs associated with a unique identifier. This metadata can be used to search for these tModels through UDDI Inquiry API calls.


businessEntity: Information about the party who publishes information about a service
tModel: Descriptions of specifications for services or taxonomies. Basis for technical fingerprints
businessService: Descriptive information about a particular family of technical services
bindingTemplate data contains references to tModels. These references designate the interface specifications for a service.
bindingTemplate: Technical information about a service entry point and construction specifications

2.2.2 businessService & bindingTemplate

Services themselves are represented in UDDI by the businessService data structure, and the details of how and where the service is accessed are provided by one or more bindingTemplate structures. The businessService might be thought of as a logical container of services. The bindingTemplate structure contains the accessPoint of the service itself, as well as references to the tModels it is said to implement.

2.3 Mapping WDSL and UDDI

To represent information about a WSDL file (such as target namespace, QName relationships and resource locations) so that it is consistently described and discovered requires a mapping practice that acknowledges the variety and flexibility within the WSDL model itself. This can be accomplished by a comprehensive use of UDDI's key entity structures (tModels) and typed meta-data construct (keyedReferences).

By representing WSDL portType and binding entities as UDDI tModels, one can accurately represent the building blocks of "abstract" WSDL interface information in UDDI. Similiarly, by representing WSDL service and port as, respectively, UDDI businessService and bindingTemplate, one achieves a similar parallel. Moreover, by representing the target namespace and dependencies of a WSDL entity as keyedReferences on the UDDI entities, one can represent strongly typed metadata about the WDSL entity within UDDI.

There are two important things to note about this mapping, especially as compared to the Version 1 *Using WSDL in a UDDI Registry* Best Practice.

First, this new approach means that a single WSDL file may be mapped to multiple tModels. For example, a WSDL file located at http://www.example.com/example.wsdl which contains a single portType definition and 2 binding definitions will map to three distinct tModels in UDDI. This decision differs significantly from the Version 1 Best Practice, which always mapped the entirety of a WSDL file to a single tModel. The rationale for this new mapping decision is allows for the modularity of WSDL to be expressed through UDDI. By decomposing WSDL into multiple tModels, one can accurately model in UDDI exactly which portTypes and bindings a given Web service supports, as opposed to being constrained to asserting that a Web service always supports the entirely of the WSDL file, which may not be the case.

While there is an increased amount of data from a WSDL file modeled in UDDI, this new approach is in accord with the original Best Practice in that it does not attempt to use UDDI as a repository for *all* of the data in a WSDL file. Just as in the Version 1 Best Practice, one still must go outside of the UDDI registry to retrieve the portType and binding information necessary for software applications to work with that Web service.

2.4 References to WSDL Components

As part of mapping WSDL in UDDI it is necessary to refer to various components in the WSDL document(s). These references occur as overviewURL values. As noted above, in this mapping several tModels can refer to the same WSDL file. The particular WSDL component SHOULD be determined by using the metadata contained within the tModel's categoryBag. Alternatively, the overviewURL value MAY contain a fragment identifier which identifies the particular WSDL component, but this is not required. If the optional fragment identifier is used then the syntax described in Appendix B MUST be used.

2.5 Mapping WSDL 1.1 in UDDI V2

2.5.1 wsdl:portType → uddi:tModel

A wsdl:portType MUST be modeled as a uddi:tModel.

The uddi:name element of the tModel MUST be the value of the name attribute of the portType.

The tModel MUST contain a categoryBag and the categoryBag MUST contain at least two keyedReferences:

1. A keyedReference with a tModelKey of the WSDL Namespace tModel and a keyValue of the target namespace of the definitions element that contains the portType.[1]
2. A keyedReference with a tModelKey of the WSDL Entity Type tModel and a keyValue of "portType".

The tModel MUST contain an overviewDoc with an overviewURL containing the location of the WSDL file.

2.5.2 wsdl:binding → uddi:tModel

A wsdl:binding MUST be modeled as a uddi:tModel.

The uddi:name element of the tModel MUST be the value of the name attribute of the binding.

The tModel MUST contain a categoryBag and the categoryBag MUST contain at least three keyedReferences:

1. A keyedReference with a tModelKey of the WSDL Entity Type tModel and a keyValue of "binding".
2. A keyedReference with a tModelKey of the WSDL Namespace tModel and a keyValue of the target namespace of the definitions element that contains the binding.
3. A keyedReference with a tModelKey of the portType tModel and a keyValue of the tModelKey that models the portType to which the binding relates.

The tModel categoryBag MAY contain an additional keyedReference for the uddi.org types taxonomy of wsdlSpec for backward compatibility[2].

The tModel MUST contain an overviewDoc with an overviewURL containing the location of the WSDL file.

2.5.3 wsdl:service → uddi:businessService

A wsdl:service MUST be modeled as a uddi:businessService. An existing businessService MAY be used or a new businessService MAY be created.

If a new businessService is created, the uddi:name of this businessService MAY be the value of the name attribute of the wsdl:service[3].

[1] WSDL 1.1 does not require the usage of a targetNamespace, but such a practice is not recommended. In the event that a WSDL file without a targetNamespace is registered in UDDI, it will not have a WSDL Namespace keyedReference and queries for these tModels based solely on the tModel name could return multiple results because no namespace can be specified.

[2] By categorizing a wsdl:binding tModel according to the Version 1 UDDI/WSDL Best Practice, backward compatibility is maintained. However, wsdl:portType tModels should not be categorized with this designation, as the wsdl:portType tModel will not contain sufficient information to compose a complete WSDL binding.

The bindingTemplates of the businessService MUST include bindingTemplates that model the ports of the service, as described in the following sections.

The businessService MUST contain a categoryBag and the categoryBag MUST contain three keyedReferences:

1. A keyedReference with a tModelKey of the WSDL Entity Type tModel and a keyValue of “service”.
2. A keyedReference with a tModelKey of the WSDL Namespace tModel and a keyValue of the target namespace of the definitions element that contains the service.
3. A keyedReference with a tModelKey of the WSDL Local Name tModel and a keyValue that is the value of the name attribute of the wsdl service.

2.5.4 wsdl:port → uddi:bindingTemplate

A wsdl :port MUST be modeled as a uddi:bindingTemplate.

The bindingTemplate tModelInstanceDetails element MUST contain the following tModelInstanceInfo elements:

1. A tModelInstanceInfo with a tModelKey value of the key of the tModel that models the wsdl:binding that this port implements. The instanceParms of this tModelInstanceInfo MUST contain the wsdl:port local name.
2. A tModelInstanceInfo with a tModelKey value of the key of the tModel that models the portType[4].

If the wsdl:binding contains a soap:binding then the bindingTemplate MUST include the following additional tModelInstanceInfo elements:

1. A tModelInstanceInfo with a tModelKey value of the key of the standard tModel that represents the SOAP extension to WSDL.
2. A tModelInstanceInfo with a tModelKey value of the key of the standard tModel that represents the HTTP transport for SOAP.

Other binding extensibility elements are handled in a similar fashion.

2.5.5 soap:address → uddi:accessPoint

A soap:address MUST be modeled as a uddi:accessPoint in the bindingTemplate that models the wsdl:port that contains the soap:address.

The value of the URLType attribute of the accessPoint MUST correspond to the transport specified by the soap:binding, or “other” if no correspondence exists. In the case of the HTTP transport for example, the value of the URLType attribute MUST be “http”.

The value of the accessPoint MUST be the value of the location attribute of the soap:address.

2.6 Mapping WSDL 1.1 in UDDI V3

The differences in UDDI data structures that are relevant are:

1. A bindingTemplate can have a categoryBag.
2. An accessPoint has a useType attribute not a URLType attribute and one of the values of useType is wsdlDeployment which indicates that the address information should be retrieved from the referenced WSDL document.
3. An overviewURL now has an optional useType attribute and a standard value of “wsdlInterface” has been defined to indicate “an abstract interface document”. This mapping assumes that “wsdlInterface” is used with tModels that represent both portTypes and bindings.

[3] Because the wsdl:service name is also captured in a categoryBag, it is not required to name the businessService after the service name attribute. In fact, such a practice is problematic because businessService can contain multiple names and, thus, the wsdl:service name could never be deterministically discovered in the businessService name.

[4] While it may seem redundant to create tModelInstanceInfos for both portType and binding, doing so allows one to query for services based on known portTypes, as opposed to just known bindings.

2.6.1 wsdl:portType → uddi:tModel

A wsdl:portType MUST be modeled as a uddi:tModel.

The name of the tModel MUST be the value of the name attribute of the portType.

The tModel MUST contain a categoryBag and the categoryBag MUST contain at least the following keyedReferences:

1. A keyedReference with a tModelKey of the WSDL Namespace tModel and a keyValue of the target namespace of the definitions element that contains the portType.
2. A keyedReference with a tModelKey of the WSDL Entity Type tModel and a keyValue of “portType”.

The tModel MUST contain an overviewDoc with an overviewURL containing the location of the WSDL file. The value of the useType attribute of the overviewURL MUST be “wsdlInterface”.

2.6.2 wsdl:binding → uddi:tModel

A wsdl:binding MUST be modeled as a uddi:tModel.

The name of the tModel MUST be the value of the name attribute of the binding.

The tModel MUST contain a categoryBag and the categoryBag MUST contain at least the following keyedReferences:

1. A keyedReference with a tModelKey of the WSDL Entity Type tModel and a keyValue of “binding”.
2. A keyedReference with a tModelKey of the WSDL Namespace tModel and a keyValue of the target namespace of the definitions element that contains the binding.
3. A keyedReference with a tModelKey of the portType tModel and a keyValue of the tModel that models the portType to which the binding relates.

The tModel categoryBag MAY contain an additional keyedReference for the uddi.org types taxonomy of wsdlSpec for backward compatibility.

The tModel MUST contain an overviewDoc with an overviewURL containing the location of the WSDL file. The value of the useType attribute of the overviewURL MUST be “wsdlInterface”.

2.6.3 wsdl:service → uddi:businessService

A wsdl:service MUST be modeled as a uddi:businessService. An existing businessService MAY be used or a new businessService MAY be created.

If a new businessService is created, the uddi:name of this businessService MAY be the value of the name attribute of the wsdl:service.

The bindingTemplates of the businessService MUST include bindingTemplates that model the ports of the service, as described in the following sections.

The businessService MUST contain a categoryBag and the categoryBag MUST contain the following keyedReferences:

1. A keyedReference with a tModelKey of the WSDL Entity Type tModel and a keyValue of “service”.
2. A keyedReference with a tModelKey of the WSDL Namespace tModel and a keyValue of the target namespace.
3. A keyedReference with a tModelKey of the WSDL Local Name tModel and a keyValue that is the value of the name attribute of the wsdl service.

2.6.4 wsdl:port → uddi:bindingTemplate

A wsdl:port MUST be modeled as a uddi:bindingTemplate.

The bindingTemplate tModelInstanceDetails element MUST contain the following tModelInstanceInfo elements:

1. A tModelInstanceInfo with a tModelKey value of the key of the tModel that models the wsdl:binding that this port implements.
2. A tModelInstanceInfo with a tModelKey value of the key of the tModel that models the portType.

If the wsdl:binding contains a soap:binding then the bindingTemplate MUST include the following additional tModelInstanceInfo elements:

1. A tModelInstanceInfo with a tModelKey value of the key of the standard tModel that represents the SOAP binding.
2. A tModelInstanceInfo with a tModelKey value of the key of the standard tModel that represents the HTTP transport.

Other binding extensibility elements are handled in a similar fashion.

The bindingTemplate MUST contain a categoryBag and the categoryBag MUST contain the following keyedReferences:

1. A keyedReference with a tModelKey of the WSDL Entity Type tModel and a keyValue of “port”.
2. A keyedReference with a tModelKey of the WSDL Namespace tModel and a keyValue of the target namespace of the definitions element that contains the port.
3. A keyedReference with a tModelKey of the WSDL Local Name tModel and a keyValue of the local name of the port.[5]

2.6.5 soap:address → uddi:accessPoint

A soap:address MUST be modeled as a uddi:accessPoint in the bindingTemplate that models the wsdl:port that contains the soap:address.

The useType of the accessPoint SHOULD be set to endPoint and the content of the accessPoint SHOULD be the value of the location attribute of the soap:address. Alternatively, the useType of the accessPoint MAY be set to wsdlDeployment and the content of the accessPoint SHOULD be a reference to the WSDL file containing the address information.

3 A Complete Example

Consider the following sample based on the WSDL file presented in the WSDL 1.1 specification. This sample shows how a single WSDL file is decomposed into two tModels (one for the portType and one for the binding) and one businessService with one bindingTemplate. It then shows the kinds of UDDI API queries that can be used for the purposes of discovery.

3.1 WSDL Sample

```
<?xml version="1.0" encoding="utf-8" ?>
<definitions
        name="StockQuote"
        targetNamespace="http://example.com/stockquote/"
        xmlns:tns="http://example.com/stockquote/"
        xmlns:xsd1="http://example.com/stockquote/schema/"
        xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
        xmlns="http://schemas.xmlsoap.org/wsdl/">
   <import
        namespace="http://example.com/stockquote/schema/"
        location="http://location/schema.xsd" />
   <message name="GetLastTradePriceInput">
        <part name="body" element="xsd1:TradePriceRequest" />
   </message>
   <message name="GetLastTradePriceOutput">
        <part name="body" element="xsd1:TradePrice" />
   </message>
   <portType name="StockQuotePortType">
        <operation name="GetLastTradePrice">
            <input message="tns:GetLastTradePriceInput" />
            <output message="tns:GetLastTradePriceOutput" />
        </operation>
   </portType>
   <binding name="StockQuoteSoapBinding" type="tns:StockQuotePortType">
        <soap:binding style="document"
            transport="http://schemas.xmlsoap.org/soap/http" />
        <operation name="GetLastTradePrice">
            <soap:operation soapAction="http://example.com/GetLastTradePrice" />
            <input>
                <soap:body use="literal" />
```

[5] Note that in the v3 modeling, the local name of the port is captured in a keyedReference, whereas in v2, it was captured in the tModelInstanceInfo instanceParm.

```
                </input>
                <output>
                    <soap:body use="literal" />
                </output>
            </operation>
        </binding>

        <service name="StockQuoteService">
            <documentation>My first service</documentation>
            <port name="StockQuotePort" binding="tns:StockQuoteBinding">
                <soap:address location="http://location/sample"/>
            </port>
        </service>

    </definitions>
```

Note that this wsdl file has one portType, one binding, one service and one port. As such, this sample represents the simplest wsdl file. Also note that the location of this WSDL is at http://location/sample.wsdl.

3.2 UDDI V2 Model

3.2.1 UDDI portType tModel

```
<tModel tModelKey="uuid:e8cf1163-8234-4b35-865f-94a7322e40c3" >
    <name>
        StockQuotePortType
    </name>
    <categoryBag>
        <keyedReference
            tModelKey="uuid:c0ec5422-44e9-4774-b2f1-5a89eca389f3"
            keyValue="http://example.com/stockquote/"
        />
        <keyedReference
            tModelKey="uuid:7950ce8d-259a-4a9a-8f13-e51632fe12d5"
            keyValue="portType"
        />
    </categoryBag>
    <overviewDoc>
        <overviewURL>
            http://location/sample.wsdl
        <overviewURL>
    <overviewDoc>
</tModel>
```

3.2.2 UDDI binding tModel

```
<tModel tModelKey="uuid:49662926-f4a5-4ba5-b8d0-32ab388dadda">
    <name>
        StockQuoteSoapBinding
    </name>
    <categoryBag>
        <keyedReference
            tModelKey="uuid:c0ec5422-44e9-4774-b2f1-5a89eca389f3"
            keyValue="http://example.com/stockquote/"
        />
        <keyedReference
            tModelKey="uuid:7950ce8d-259a-4a9a-8f13-e51632fe12d5"
            keyValue="binding"
        />
        <keyedReference
            tModelKey="uuid:acc0db62-2866-4081-8986-d62a7bf75ab3"
            keyValue="uuid:e8cf1163-8234-4b35-865f-94a7322e40c3"
        />
        <keyedReference
            tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4"
            keyValue="wsdlSpec"
        />
    </categoryBag>
    <overviewDoc>
        <overviewURL>
            http://location/sample.wsdl
        <overviewURL>
    <overviewDoc>
</tModel>
```

Note how this tModel creates a link to the portType tModel through placing the tModelKey of the portType tModel above as its keyValue.

3.2.3 UDDI businessService and bindingTemplate

```
    <businessService
            serviceKey="102b114a-52e0-4af4-a292-02700da543d4"
            businessKey="1e65ea29-4e0f-4807-8098-d352d7b10368">
        <name>StockQuoteService</name>
        <bindingTemplates>
            <bindingTemplate
                    bindingKey="f793c521-0daf-434c-8700-0e32da232e74"
                    serviceKey="102b114a-52e0-4af4-a292-02700da543d4">
                <accessPoint URLType="http">
                    http://location/sample
                </accessPoint>
                <tModelInstanceDetails>
          <tModelInstanceInfo
            tModelKey="uuid:454ef824-994f-4ce3-a892-3d97ad952707">
            <description xml:lang="en">
              This tModel represents the SOAP binding
                            extensibility element.
            </description>
                    <tModelInstanceInfo
                        tModelKey="uuid:26de470c-0dab-4471-89f9-57450a524005">
                        <description xml:lang="en">
                            This tModel represents the HTTP transport using SOAP.
                        </description>
                    </tModelInstanceInfo>
                    <tModelInstanceInfo
                        tModelKey="uuid:49662926-f4a5-4ba5-b8d0-32ab388dadda">
                        <description xml:lang="en">
                            This tModel key represents the wsdl:binding that this
                            wsdl:port implements.
                        </description>
                        <instanceDetails>
                                <instanceParms>StockQuotePort</instanceParms>
                        </instanceDetails>
                    </tModelInstanceInfo>
                    <tModelInstanceInfo
                        tModelKey="uuid:e8cf1163-8234-4b35-865f-94a7322e40c3">
                        <description xml:lang="en">
                            This tModel represents the wsdl:portType that
                            this wsdl:port implements.
                        </description>
                    </tModelInstanceInfo>
                </tModelInstanceDetails>
            </bindingTemplate>
        </bindingTemplates>
        <categoryBag>
         <keyedReference
                tModelKey="uuid:7950cc8d-259a-4a9a-8f13-e51632fe12d5"
                keyValue="service"
            />
         <keyedReference
                tModelKey="uuid:c0ec5422-44e9-4774-b2f1-5a89eca389f3"
                keyValue="http://example.com/stockquote/"
            />
         <keyedReference
                tModelKey="uuid:248ccb56-6365-446d-847a-91171ae79740"
                keyValue="StockQuoteService"
            />

      </categoryBag>
    </businessService>
```

3.3 Sample V2 Queries

This section shows how to perform various UDDI V2 queries given the model of the example.

3.3.1 Find tModel for the portType StockQuotePortType in the namespace http://example.com/stockquote/

```
<find_tModel generic=”2.0” xmlns=”urn:uddi-org:api_v2”>
  <name>StockQuotePortType</name>
  <categoryBag>
    <keyedReference tModelKey=”uuid:7950ce8d-259a-4a9a-8f13-e51632fe12d5” keyValue=”portType”/>
    <keyedReference tModelKey=”uuid:c0ec5422-44e9-4774-b2f1-5a89eca389f3” keyValue=”http://example.com/stockquote/”/>
  </categoryBag>
</find_tModel>
```

This should return the tModelKey uuid:e8cf1163-8234-4b35-865f-94a7322e40c3.

3.3.2 Find bindings for portType

```
<find_tModel generic=”2.0” xmlns=”urn:uddi-org:api_v2”>
  <categoryBag>
    <keyedReference tModelKey=”uuid:7950ce8d-259a-4a9a-8f13-e51632fe12d5” keyValue=”binding”/>
    <keyedReference tModelKey=”uuid:acc0db62-2866-4081-8986-d62a7bf75ab3” keyValue=”uuid:e8cf1163-8234-4b35-865f-94a7322e40c3”/>
  </categoryBag>
</find_tModel>
```

This should return the tModelKey uuid:49662926-f4a5-4ba5-b8d0-32ab388dadda.

3.3.3 Find Implementations of portType

```
<find_binding generic=”2.0” xmlns=”urn:uddi-org:api_v2”>
  <tModelBag>
    <tModelKey>uuid:e8cf1163-8234-4b35-865f-94a7322e40c3</tModelKey>
  </tModelBag>
</find_binding>
```

This should return the bindingKey f793c521-0daf-434c-8700-0e32da232e74.

3.3.4 Find implementations of binding

```
<find_binding generic=”2.0” xmlns=”urn:uddi-org:api_v2”>
  <tModelBag>
    <tModelKey>uuid:49662926-f4a5-4ba5-b8d0-32ab388dadda</tModelKey>
  </tModelBag>
</find_binding>
```

This should return the bindingKey f793c521-0daf-434c-8700-0e32da232e74.

3.3.5 Find SOAP Implementations of portType

```
<find_binding generic=”2.0” xmlns=”urn:uddi-org:api_v2”>
  <tModelBag>
    <tModelKey>uuid:e8cf1163-8234-4b35-865f-94a7322e40c3</tModelKey>
    <tModelKey>uuid:454ef824-994f-4ce3-a892-3d97ad952707</tModelKey>
  </tModelBag>
</find_binding>
```

This should return the bindingKey f793c521-0daf-434c-8700-0e32da232e74.

3.3.6 Find SOAP/HTTP Implementations of portType

```
<find_binding generic=”2.0” xmlns=”urn:uddi-org:api_v2”>
  <tModelBag>
    <tModelKey>uuid:e8cf1163-8234-4b35-865f-94a7322e40c3</tModelKey>
    <tModelKey>uuid:454ef824-994f-4ce3-a892-3d97ad952707</tModelKey>
    <tModelKey>uuid:26de470c-0dab-4471-89f9-57450a524005</tModelKey>
  </tModelBag>
</find_binding>
```

This should return the bindingKey f793c521-0daf-434c-8700-0e32da232e74.

3.3.7 Find the portType of a binding

This is simply the keyValue of the keyedReference in the categoryBag with tModelKey=”uuid:acc0db62-2866-4081-8986-d62a7bf75ab3” and therefore no query is required once the tModel of the binding is obtained.

3.4 UDDI V3 Model

3.4.1 UDDI portType tModel

```
<tModel tModelKey="uddi:e8cf1163-8234-4b35-865f-94a7322e40c3" >
    <name>
        StockQuotePortType
    </name>
    <categoryBag>
        <keyedReference
            tModelKey="uddi:c0ec5422-44e9-4774-b2f1-5a89eca389f3"
            keyValue="http://example.com/stockquote/"
        />
        <keyedReference
            tModelKey="uddi:7950ce8d-259a-4a9a-8f13-e51632fe12d5"
            keyValue="portType"
        />
    </categoryBag>
    <overviewDoc>
        <overviewURL useType="wsdlInterface">
            http://location/sample.wsdl
        <overviewURL>
    <overviewDoc>
</tModel>
```

3.4.2 UDDI binding tModel

```
<tModel tModelKey="uddi:49662926-f4a5-4ba5-b8d0-32ab388dadda">
    <name>
        StockQuoteBinding
    </name>
    <categoryBag>
        <keyedReference
            tModelKey="uddi:c0ec5422-44e9-4774-b2f1-5a89eca389f3"
            keyValue="http://example.com/stockquote/"
        />
        <keyedReference
            tModelKey="uddi:7950ce8d-259a-4a9a-8f13-e51632fe12d5"
            keyValue="binding"
        />
        <keyedReference
            tModelKey="uddi:acc0db62-2866-4081-8986-d62a7bf75ab3"
            keyValue="uddi:e8cf1163-8234-4b35-865f-94a7322e40c3"
        />
        <keyedReference
            tModelKey="uddi:c1acf26d-9672-4404-9d70-39b756e62ab4"
            keyValue="wsdlSpec"
        />
    </categoryBag>
    <overviewDoc>
        <overviewURL useType="wsdlInterface">
            http://location/sample.wsdl
        <overviewURL>
    <overviewDoc>
</tModel>
```

Note how this tModel creates a link to the portType tModel through placing the tModelKey of the portType tModel above as its keyValue.

3.4.3 UDDI businessService and bindingTemplate

```
<businessService
        serviceKey="uddi:102b114a-52e0-4af4-a292-02700da543d4"
        businessKey="uddi:1e65ea29-4e0f-4807-8098-d352d7b10368">
    <name>StockQuoteService</name>
    <bindingTemplates>
        <bindingTemplate
                bindingKey="uddi:f793c521-0daf-434c-8700-0e32da232e74"
                serviceKey="uddi:102b114a-52e0-4af4-a292-02700da543d4">
            <accessPoint useType="endPoint">
                http://location/sample
```

```
                              </accessPoint>
                              <tModelInstanceDetails>
                        <tModelInstanceInfo
                           tModelKey=”uddi:454ef824-994f-4ce3-a892-3d97ad952707”>
                           <description xml:lang=”en”>
                              This tModel represents the SOAP binding
                                                extensibility element.
                           </description>
                                    <tModelInstanceInfo
                                         tModelKey="uddi:26de470c-0dab-4471-89f9-57450a524005">
                                         <description xml:lang="en">
                                                This tModel represents the HTTP transport using SOAP.
                                         </description>
                                    </tModelInstanceInfo>
                                    <tModelInstanceInfo
                                         tModelKey="uddi:49662926-f4a5-4ba5-b8d0-32ab388dadda">
                                         <description xml:lang="en">
                                                This tModel key represents the wsdl:binding that this
                                                wsdl:port implements.
                                         </description>
                                    </tModelInstanceInfo>
                                    <tModelInstanceInfo
                                         tModelKey="uddi:e8cf1163-8234-4b35-865f-94a7322e40c3">
                                         <description xml:lang="en">
                                                This tModel represents the wsdl:portType that
                                                this wsdl:port implements.
                                         </description>
                                    </tModelInstanceInfo>
                              </tModelInstanceDetails>
                        <categoryBag>
                        <keyedReference
                                   tModelKey=”uddi:7950cc8d-259a-4a9a-8f13-e51632fe12d5”
                                   keyValue=”port”
                              />
                        <keyedReference
                                   tModelKey="uddi:c0ec5422-44e9-4774-b2f1-5a89eca389f3"
                                   keyValue=”http://example.com/stockquote/”
                              />
                        <keyedReference
                                   tModelKey="uddi:248ccb56-6365-446d-847a-91171ae79740"
                                   keyValue=”StockQuotePort”
                              />
                      </categoryBag>
                           </bindingTemplate>
                   </bindingTemplates>
                   <categoryBag>
                    <keyedReference
                              tModelKey=”uddi:7950cc8d-259a-4a9a-8f13-e51632fe12d5”
                              keyValue=”service”
                         />
                    <keyedReference
                              tModelKey="uddi:c0ec5422-44e9-4774-b2f1-5a89eca389f3"
                              keyValue=”http://example.com/stockquote/”
                         />
                    <keyedReference
                            tModelKey="uddi:248ccb56-6365-446d-847a-91171ae79740"
                            keyValue=”StockQuoteService”
                         />
                 </categoryBag>
               </businessService>
```

3.5 Sample V3 Queries

This section shows how to perform various UDDI V3 queries given the model of the example.

3.5.1 Find tModel for the portType StockQuotePortType in the namespace http://example.com/stockquote/

```
<find_tModel xmlns=”urn:uddi-org:api_v3”>
   <name>StockQuotePortType</name>
   <categoryBag>
      <keyedReference tModelKey=”uddi:7950ce8d-259a-4a9a-8f13-e51632fe12d5” keyValue=”portType”/>
      <keyedReference tModelKey=”uddi:5165f13f-139c-4649-adf8-89f66e437f55” keyValue=”http://example.com/stockquote/”/>
   </categoryBag>
</find_tModel>
```

This should return the tModelKey uddi:e8cf1163-8234-4b35-865f-94a7322e40c3.

3.5.2 Find bindings for portType

```
<find_tModel xmlns=”urn:uddi-org:api_v3”>
  <categoryBag>
    <keyedReference tModelKey=”uddi:7950ce8d-259a-4a9a-8f13-e51632fe12d5” keyValue=”binding”/>
    <keyedReference tModelKey=”uddi:acc0db62-2866-4081-8986-d62a7bf75ab3” keyValue=”uddi:e8cf1163-8234-4b35-865f-94a7322e40c3”/>
  </categoryBag>
</find_tModel>
```

This should return the tModelKey uddi:49662926-f4a5-4ba5-b8d0-32ab388dadda.

3.5.3 Find Implementations of portType

```
<find_binding xmlns=”urn:uddi-org:api_v3”>
  <tModelBag>
    <tModelKey>uddi:e8cf1163-8234-4b35-865f-94a7322e40c3</tModelKey>
  </tModelBag>
</find_binding>
```

This should return the bindingKey f793c521-0daf-434c-8700-0e32da232e74.

3.5.4 Find Implementations of binding

```
<find_binding xmlns=”urn:uddi-org:api_v3”>
  <tModelBag>
    <tModelKey>uddi:49662926-f4a5-4ba5-b8d0-32ab388dadda</tModelKey>
  </tModelBag>
</find_binding>
```

This should return the bindingKey f793c521-0daf-434c-8700-0e32da232e74.

3.5.5 Find SOAP Implementations of portType

```
<find_binding xmlns=”urn:uddi-org:api_v3”>
  <tModelBag>
    <tModelKey>uddi:e8cf1163-8234-4b35-865f-94a7322e40c3</tModelKey>
    <tModelKey>(key of tModel representing SOAP extensibility element)</tModelKey>
  </tModelBag>
</find_binding>
```

This should return the bindingKey f793c521-0daf-434c-8700-0e32da232e74.

3.5.6 Find SOAP/HTTP Implementations of portType

```
<find_binding xmlns=”urn:uddi-org:api_v3”>
  <tModelBag>
    <tModelKey>uddi:e8cf1163-8234-4b35-865f-94a7322e40c3</tModelKey>
    <tModelKey>(key of tModel representing SOAP extensibility element)</tModelKey>
    <tModelKey>uddi:1be8abb8-86e3-4ee5-99ac-b7fffbed8110</tModelKey>
  </tModelBag>
</find_binding>
```

This should return the bindingKey uddi:f793c521-0daf-434c-8700-0e32da232e74.

3.5.7 Find the portType of a binding

This is simply the keyValue of the keyedReference in the categoryBag with tModelKey=”uddi:acc0db62-2866-4081-8986-d62a7bf75ab3” and therefore no query is required once the tModel of the binding is obtained.

4 References

1. UDDI Version 2.0 Data Structure Reference, July 7, 2002. Available at http://uddi.org/pubs/DataStructure-V2.03-Published-20020719.pdf.

2. Using WSDL in a UDDI Registry, May 21, 2002. Available at http://uddi.org/pubs/wsdlbestpractices.pdf

3. Web Services Description Language (WSDL) 1.1, March 15, 2000. Available at http://www.w3.org/TR/wsdl

4. XML Pointer Language (XPointer) Version 1.0, January 8, 2001. Available at http://www.w3.org/TR/WD-xptr

Appendix A: Canonical tModels

A.1 WSDL Entity Type tModel

A.1.1 Design Goals

This tModel provides a typing system based on the WSDL entity which the UDDI entity represents.

A.1.2 Definition

Name: uddi.org:wsdl:types

Description: WSDL Type Category System

V3 format key: uddi:7950ce8d-259a-4a9a-8f13-e51632fe12d5

V1,V2 format key: uuid:7950ce8d-259a-4a9a-8f13-e51632fe12d5

Categorization: categorization

Checked: no

A.1.2.1 tModel Structure

```
<tModel tModelKey="uuid:7950ce8d-259a-4a9a-8f13-e51632fe12d5" >
    <name>uddi.org:wsdl:types</name>
        <overviewDoc>
        <overviewURL>
            http://uddi.org/pubs/uddi_wsdl_technical_note_v2.htm#wsdlTypes
        </overviewURL>
        </overviewDoc>
    <categoryBag>
     <keyedReference tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4" keyValue="unchecked" />
     <keyedReference tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4" keyValue="categorization" />
    </categoryBag>
</tModel>
```

A.1.3 Values

While this is an unchecked taxonomy, there are only four values that should be used with this taxonomy:

ID	Description	UDDI Entity
portType	Represents a UDDI entity categorized as a wsdl:portType	tModel
binding	Represents a UDDI entity categorized as a wsdl:binding	tModel
service	Represents a UDDI entity categorized as a wsdl:service	businessService
port	Represents a UDDI entity categorized as a wsdl:port	bindingTemplate (v3 only)

A.1.4 Example of Use

A tModel representing a portType tModel would have a categoryBag representing its type:

```
<categoryBag>
    <keyedReference tModelKey="uuid:7950ce8d-259a-4a9a-8f13-e51632fe12d5" keyName="WSDL Entity type" keyValue="portType" />
...
</categoryBag>
```

A.2 WSDL Namespace tModel

A.2.1 Design Goals

Because each WSDL entity has an associated target namespace, there needs to be a way to capture that target namespace. The WSDL Namespace tModel serves this purpose. *More than one tModel might be categorized with the same namespace tModel* – in fact, this mapping would be quite common, as many WSDL files use a common target namespace for <wsdl:portType>, <wsdl:binding> and <wsdl:service> elements.

A.2.2 Definition

Name: uddi.org:wsdl:namespace

Description: A tModel used to represent WSDL namespaces

V3 format key: uddi:c0ec5422-44e9-4774-b2f1-5a89eca389f3

V1,V2 format key: uuid:c0ec5422-44e9-4774-b2f1-5a89eca389f3

Categorization: categorization

Checked: No

A.2.2.1 tModel Structure

```
<tModel tModelKey="uuid:c0ec5422-44e9-4774-b2f1-5a89eca389f3" >
    <name>uddi.org:wsdl:namespace</name>
        <overviewDoc>
        <overviewURL>
            http://uddi.org/pubs/uddi_wsdl_technical_note_v2.htm#wsdlNamespace
        </overviewURL>
        </overviewDoc>
    <categoryBag>
     <keyedReference tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4" keyValue="unchecked" />
     <keyedReference tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4" keyValue="categorization" />
    </categoryBag>
</tModel>
```

A.2.3 Values

The values used in this taxonomy come directly from the different target namespaces being represented by instances of this tModel.

A.2.4 Example of Use

A namespace keyedReference would be as follows:

```
<categoryBag>
     <keyedReference tModelKey=" uuid:c0ec5422-44e9-4774-b2f1-5a89eca389f3" keyName="namespace" keyValue="urn:foo" />
...
</categoryBag>
```

A.3 WSDL Local Name tModel

A.3.1 Design Goals

Because each WSDL entity has a name attribute which is key to identifying that WSDL entity, there needs to be a way to capture that attribute in the mapping. In the case of portType and binding, this is mapped to the tModel name element. However, in the case of the wsdl:service entity, the name element of the businessService is not appropriate and, in the case of wsdl:port, the bindingTemplate entity does not have a name element. The WSDL Local Name tModel serves this purpose of capturing that information in these two constructs.

A.3.2 Definition

Name: uddi.org:wsdl:localname

Description: A tModel used to represent WSDL local names

V3 format key: uddi:248ccb56-6365-446d-847a-91171ae79740

V1,V2 format key: uuid:248ccb56-6365-446d-847a-91171ae79740

Categorization: categorization

Checked: No

A.3.2.1 tModel Structure

```
<tModel tModelKey="uuid:248ccb56-6365-446d-847a-91171ae79740" >
        <name>uddi.org:wsdl:localname</name>
        <overviewDoc>
        <overviewURL>
            http://uddi.org/pubs/uddi_wsdl_technical_note_v2.htm#wsdlLocalName
        </overviewURL>
        </overviewDoc>
    <categoryBag>
     <keyedReference tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4" keyValue="unchecked" />
     <keyedReference tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4" keyValue="categorization" />
    </categoryBag>
</tModel>
```

A.3.3 Values

The values used in this taxonomy come directly from the different target namespaces being signified by this tModel.

A.3.4 Example of Use

A namespace keyedReference would be as follows:

```
<categoryBag>
      <keyedReference tModelKey="uuid:248ccb56-6365-446d-847a-91171ae79740" keyValue="StockQuoteService" />
...
</categoryBag>
```

A.4 WSDL portType Reference tModel

A.4.1 Design Goals

There is also a need to be able to relate a wsdl:binding tModel to the wsdl:portType that it implements. In order to accomplish this goal, a tModel needs to be established that allow arcs to be created between tModels. This is accomplished by placing the tModel key of a given tModel as the keyValue of a keyedReference. In order to create these arcs, a tModel is needed that represents the wsdl:portType.

A.4.2 Definition

Name: uddi.org:wsdl:portTypeReference

Description:

V3 format key: uddi:acc0db62-2866-4081-8986-d62a7bf75ab3

V1,V2 format key: uuid:acc0db62-2866-4081-8986-d62a7bf75ab3

Categorization:

A.4.2.1 tModel Structure

```
<tModel tModelKey="uuid:acc0db62-2866-4081-8986-d62a7bf75ab3" >
     <name>uddi.org:wsdl:portTypeReference</name>
     <description xml:lang="en">This tModel is a namespace tModel that can be used to identify other tModels as portType tModels.</description>
          <overviewDoc>
          <overviewURL>
               http://uddi.org/pubs/uddi_wsdl_technical_note_v2.htm#portTypeReference
          </overviewURL>
          </overviewDoc>
     <categoryBag>
      <keyedReference tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4" keyValue="categorization" />
      <keyedReference tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4" keyValue="unchecked" />
     </categoryBag>
</tModel>
```

A.4.3 Values

The values used in this taxonomy come directly from the tModelKeys that represent wsdl:portType tModels.

A.4.4 Example of Use

One would add the following keyed reference to signify wsdl:binding that implemented a known portType:

```
<categoryBag>
    <keyedReference
          tModelKey="uuid:acc0db62-2866-4081-8986-d62a7bf75ab3"
          keyName="wsdl:portType Reference"
          keyValue="uuid:e8cf1163-8234-4b35-865f-94a7322e40c3"
     />
…
</categoryBag>
```

Note that the keyValue is a GUID, which, if queried for using get_tModelDetail, would return a tModel that represented a given portType.

A.5 SOAP Protocol tModel

A.5.1 Design Goals

The WSDL specification outlines usage of the SOAP extension to WSDL. This tModel is used to signify that a given bindingTemplate uses the SOAP extension. Definition

Name: http://schemas.xmlsoap.org/wsdl/soap/

Description: Used to represent the WSDL SOAP extension

V3 format key: uddi:454ef824-994f-4ce3-a892-3d97ad952707

V1,V2 format key: uuid:454ef824-994f-4ce3-a892-3d97ad952707

Categorization: protocol

A.5.1.1 tModel Structure

```
<tModel tModelKey="uuid:454ef824-994f-4ce3-a892-3d97ad952707">
    <name>http://schemas.xmlsoap.org/wsdl/soap/</name>
         <overviewDoc>
         <overviewURL>
              http://uddi.org/pubs/uddi_wsdl_technical_note_v2.htm#soap
         </overviewURL>
         </overviewDoc>
    <categoryBag>
     <keyedReference
              tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4"
              keyName="Protocol"
              keyValue="protocol"
        />
</categoryBag>
</tModel>
```

A.5.2 Example of Use

In the case when a uddi:bindingTemplate implements a wsdl:binding, it is important to model the protocol used by that binding. In such a way, a query can be run: show all bindings that support a given protocol.

```
<bindingTemplates>
     <bindingTemplate
               bindingKey="f793c521-0daf-434c-8700-0e32da232e74"
               serviceKey="102b114a-52e0-4af4-a292-02700da543d4">
          <accessPoint URLType="http">
               http://location/sample
          </accessPoint>
          <tModelInstanceDetails>
               <tModelInstanceInfo
                    tModelKey="uuid:454ef824-994f-4ce3-a892-3d97ad952707">
                    <description xml:lang="en">
                         This tModel key represents the soap protocol
                    </description>
               </tModelInstanceInfo>
               … … …
          </tModelInstanceDetails>
     </bindingTemplate>
</bindingTemplates>
```

A.6 SOAP over HTTP Transport tModel

A.6.1 Design Goals

Other transport tModels may be defined in the future.

A.6.2 Definition

Name: http://schemas.xmlsoap.org/soap/http

Description: Used to represent soap over http

V3 format key: uddi:26de470c-0dab-4471-89f9-57450a524005

V1,V2 format key: uuid:26de470c-0dab-4471-89f9-57450a524005

Categorization: transport

A.6.2.1 tModel Structure

```
<tModel tModelKey="uuid:26de470c-0dab-4471-89f9-57450a524005" >
    <name>http://schemas.xmlsoap.org/soap/http</name>
         <overviewDoc>
         <overviewURL>
              http://uddi.org/pubs/uddi_wsdl_technical_note_v2.htm#http
```

```
                </overviewURL>
                </overviewDoc>
            <categoryBag>
              <keyedReference tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4"
                   keyName="Wire/transport protocol"
                   keyValue="transport" />
        </categoryBag>
        </tModel>
```

A.6.3 Example of Use

In the case when a uddi:bindingTemplate implements a wsdl:binding, it is important to model the transport used by that binding. In such a way, a query can be run: show all bindings that support a given transport.

```
<bindingTemplates>
        <bindingTemplate
                bindingKey="f793c521-0daf-434c-8700-0e32da232e74"
                serviceKey="102b114a-52e0-4af4-a292-02700da543d4">
            <accessPoint URLType="http">
                http://location/sample
            </accessPoint>
            <tModelInstanceDetails>
                <tModelInstanceInfo
                    tModelKey="uuid:26de470c-0dab-4471-89f9-57450a524005">
                    <description xml:lang="en">
                        This tModel key represents the wsdl:binding transport
                    </description>
                </tModelInstanceInfo>
            … … …
            </tModelInstanceDetails>
        </bindingTemplate>
</bindingTemplates>
```

Appendix B: Using XPointer in overviewURL

B.1 XPointer Syntax

As mentioned earlier, it is optional to have a fragment identifier in an overviewURL that identifies a WSDL document.

As the WSDL 1.1 schema does not allow for id attributes on WSDL elements, we cannot simply use a fragment identifier of the form #foo.

If the optional fragment identifier is used, the syntax defined by Xpointer [4] MUST be used for the fragment identifier.

Considering Example 2 in the WSDL 1.1 W3C Note, the form of the reference to the portType named StockQuotePortType would be http://example.com/stockquote/stockquote.wsdl#xpointer(//definitions/portType[@name="StockQuotePortType"]) whereas the full Xpointer syntax for the same reference, including the namespace, would be http://example.com/stockquote/stockquote.wsdl#xmlns(x=http://example.com/stockquote/definitions) xpointer(//x:definitions/portType[@name="StockQuotePortType"]).