



Search
for:

Use + - () " "

within

[Search help](#)

[IBM home](#) | [Products & services](#) | [Support & downloads](#) | [My account](#)

[IBM developerWorks](#) > [Web services](#)

developerWorks

A new approach to UDDI and WSDL:
Introduction to the new OASIS UDDI WSDL
Technical Note



The why and what of the new approach

Level: Advanced

[John Colgrave](#) (colgrave@uk.ibm.com)

Senior Software Engineer, IBM United Kingdom Limited
August 19, 2003

This is the first article in a series of articles that relate to a new approach to using WSDL and UDDI, described in a new OASIS UDDI Technical Note. This article describes the motivation and goals of the new Technical Note, and describes the approach set out in the Technical Note.

The OASIS UDDI Specifications Technical Committee has recently published a new Technical Note that describes a much richer approach to using UDDI to model a WSDL description of a Web service. This is a significant improvement over previous methods that needs some exploration and explanation.

This article is the first in a series that examines this new approach to modeling WSDL in UDDI and shows how to write applications that support it. In this article, the background to the new Technical Note is described, followed by an explanation of the approach taken. The reader is assumed to be familiar with WSDL 1.1 and UDDI V2.

The second article will examine the types of query that can be executed against this UDDI model.

The third article will present a more complex example than the one in the Technical Note, including screen shots showing how to publish the UDDI entities and how to construct the types of query described in the Technical Note.

The final articles will describe how to write Java™ applications that can publish these models to UDDI and issue the queries, using UDDI4J.

What is a Technical Note?

A Technical Note is a document published by the OASIS UDDI Specifications Technical Committee that provides guidance on how to use UDDI registries. Once an approach described in a Technical Note has been implemented and become established, it becomes a candidate for a Best Practice.

Why is a new Technical Note needed?

Contents:

[What is a Technical Note?](#)

[Why is a new Technical Note needed?](#)

[The goals of the new Technical Note](#)

[New Canonical tModels](#)

[How to Model WSDL Descriptions in UDDI](#)

[Summary](#)

[Conclusion](#)

[Resources](#)

[About the author](#)

[Rate this article](#)

Related content:

[Understanding WSDL in a UDDI registry, Part 1](#)

[Understanding WSDL in a UDDI registry, Part 2](#)

[Understanding WSDL in a UDDI registry, Part 2](#)

[Subscribe to the developerWorks newsletter](#)

[developerWorks Toolbox subscription](#)

Also in the Web services zone:

[Tutorials](#)

[Tools and products](#)

[Articles](#)

The existing Best Practice on using WSDL and UDDI is limited in scope and focussed on the WSDL binding, whereas the programming models that are emerging for Web services such as JAX-RPC are focussed on the WSDL portType.

The existing Best Practice provides no support for queries other than the very simple query of finding tModels that relate to WSDL descriptions. No other information is captured so there is no other information to issue queries against.

The existing Best Practice did not encompass the WSDL service and port elements which meant that there was no standard way to find implementations of a given portType or binding.

The goals of the new Technical Note

The goals of the new Technical Note are as follows:

1. Support the modeling of WSDL portTypes in UDDI.
2. Support the modeling of WSDL bindings in UDDI, in a way that is compatible with the Version 1 Best Practice.
3. Support the modeling of WSDL services and ports to allow queries to find implementations of particular portTypes and/or bindings.
4. Provide a model that is consistent when manipulated through either the UDDI V2 API or the UDDI V3 API.
5. Support any logical or physical structure of WSDL description, any number of portTypes etc. and any number of separate files/resources.
6. Capture sufficient information from the WSDL description without excessive duplication of information.
7. Support precise and flexible UDDI queries against the model.

New Canonical tModels

The Technical Note introduces nine new tModels that must be provided by a UDDI registry in order to support the approach described in the Technical Note. Six of these tModels represent new Category Systems and the remaining ones are other types of tModel. These new tModels are described in the following sections.

New Category Systems

The Technical Note makes extensive use of categorization and introduces six new Category Systems:

1. WSDL Entity Type Category System
2. XML Namespace Category System
3. XML Local Name Category System
4. WSDL portType Reference Category System
5. Protocol Category System
6. Transport Category System

These new Category Systems are described in the following sections.

WSDL Entity Type Category System

This Category System is used to identify UDDI entities that correspond to WSDL entities. It is currently used with the following UDDI entities:

1. tModels
2. businessServices

As WSDL allows a portType and a binding to have the same name a keyedReference using this Category System may be the only way to differentiate two tModels, one of which relates to a portType and the other to a binding.

XML Namespace Category System

As its name suggests, this Category System is used to represent any XML namespace, not just WSDL-related namespaces. It is defined in the WSDL Technical Note because that is the first context in which it is required but it can be used with any XML-based modeling approach. Capturing the XML namespace of an entity enables precise queries to be made using full XML entity names, consisting of both a namespace name and a local name.

XML Local Name Category System

This is another general XML Category System which is used to hold a local name when the name of the corresponding UDDI entity cannot be the same as the XML element's local name.

WSDL portType Reference Category System

This Category System is used to represent the link between a binding and a portType by allowing one UDDI entity to refer to another UDDI entity by storing its key as the keyValue attribute value of a keyedReference relating to this Category System.

Protocol Category System

This Category System allows user-defined protocol tModels to be incorporated into the approach described in the Technical Note.

Transport Category System

This Category System allows user-defined transport tModels to be incorporated into the approach described in the Technical Note.

Other New tModels

The Technical Note also introduces several other canonical tModels and these are described in the following sections.

SOAP Protocol tModel

This tModel is used to represent the use of the WSDL SOAP binding.

HTTP Protocol tModel

This tModel is used to represent the use of the WSDL HTTP binding. Note that this is a different tModel to the HTTP transport tModel which is used with the SOAP protocol tModel to represent the use of a SOAP/HTTP binding.

WSDL Address tModel

This tModel is used when a WSDL Implementation Document is used.

How to Model WSDL Descriptions in UDDI

This section describes the approach defined in the Technical Note. The model for each of the WSDL entities is described. The description is in terms of the UDDI V2 API and Data Structures specifications. There is very little difference in the UDDI V3 model. For details see the Technical Note.

WSDL portType

A WSDL portType is represented by a UDDI tModel. This tModel is categorized, using the WSDL Entity Type Category System described above, as a WSDL portType tModel to distinguish it from any other type of tModel.

The name of the portType tModel is the name of the portType. This goes against the advice in the UDDI Data Structure Specification that a tModel name should be formatted as a URI. The options considered for the tModel name were:

1. Use the portType name. This was the option chosen.
2. Use a combination of the portType name and the namespace name.
3. Use some other URI and put both the local name and the namespace name in the tModel categoryBag.

Option 2 would make it more difficult to search on either the namespace name or the portType name.

Option 3 would make it more difficult to search on the portType name and there would be no information

represented in the tModel name. Options 2 and 3 could be combined but that would introduce redundant data as both the namespace name and the portType name would be stored twice, once in the categoryBag and once as part of the tModel name.

The namespace of the portType is represented by a keyedReference in the categoryBag of the portType tModel. This keyedReference relates to the XML Namespace Category System.

As the detailed information in the portType relating to messages, operations etc., is not duplicated in UDDI the portType tModel must refer to the WSDL document in which the portType is defined so that tools that need this information such as application development tools that wish to generate a programming language interface from the portType, or sophisticated systems that wish to validate requests against the definition of the portType can retrieve the WSDL document. The tModel overviewURL is used to hold the URL for the WSDL document.

WSDL binding

A WSDL binding is represented by a UDDI tModel. This tModel is categorized as a WSDL binding tModel to distinguish it from any other type of tModel. This categorization uses the same Category System as the portType tModel, but with a different keyValue value to differentiate a binding tModel from a portType tModel.

The name of the binding tModel is the name of the binding. The options above relating to the name of a portType tModel were also considered for binding tModels, and the same approach was chosen.

The namespace of the binding is represented by a keyedReference in the categoryBag of the binding tModel. This keyedReference relates to another new Category System defined in the Technical Note for the purpose of holding an XML namespace name.

As the detailed information in the binding relating to document/literal etc. is not duplicated in UDDI the binding tModel must refer to the WSDL document in which the binding is defined so that tools that need this information can retrieve the WSDL document. The tModel overviewURL is used to hold the URL for the WSDL document.

The link between the binding and the portType it relates to is represented by another keyedReference in the categoryBag of the binding tModel. The tModelKey value of this keyedReference is the key of a new canonical tModel representing a Category System that has valid values of the keys of other tModels, specifically portType tModels. The keyValue value of the keyedReference is the key of the portType tModel that represents the portType that the binding relates to.

The protocol that the binding represents is represented by another keyedReference in the categoryBag of the binding tModel. The tModelKey value of this keyedReference is the key of a new canonical tModel representing a Category System that has valid values of the keys of other tModels, specifically protocol tModels. The keyValue value of the keyedReference is the key of the appropriate protocol tModel. In the common case of SOAP/HTTP, a new SOAP protocol tModel has been defined by the Technical Note and the key of this tModel would be used. Other protocol tModels can be published and they would automatically be valid for this new Category System.

If the binding represents a transport in addition to the protocol, then a similar approach is used to represent the transport information. There is another new canonical tModel defined in the Technical Note that has valid values of the keys of transport tModels, and a keyedReference relating to this Category System is added to the binding tModel. In the common case of SOAP/HTTP, the standard HTTP transport tModel is used. Other transport tModels can be defined to UDDI and they automatically become valid for this Category System.

For compatibility with the Version 1 Best Practice a binding tModel should be categorized using wsdlSpec.

WSDL service

A WSDL service is represented by a UDDI businessService. If the WSDL service represents a Web

service interface for an existing service then there may be an existing UDDI businessService that is relevant, in which case the WSDL information can be added to that existing service. If there is no suitable existing service then a new UDDI businessService can be created.

There are two approaches to representing a WSDL service and its ports:

1. If there are no extensibility elements used in the WSDL service, and the address in the extensibility element on each port is of a form that can be represented as a UDDI accessPoint then the normal approach is to represent all of the information from the service and its ports in UDDI and not to refer to the WSDL file containing the service.
2. If this is not possible then an alternative approach is defined in which a reference to the WSDL containing the service is kept and the WSDL document must be retrieve to obtain the information.

WSDL port

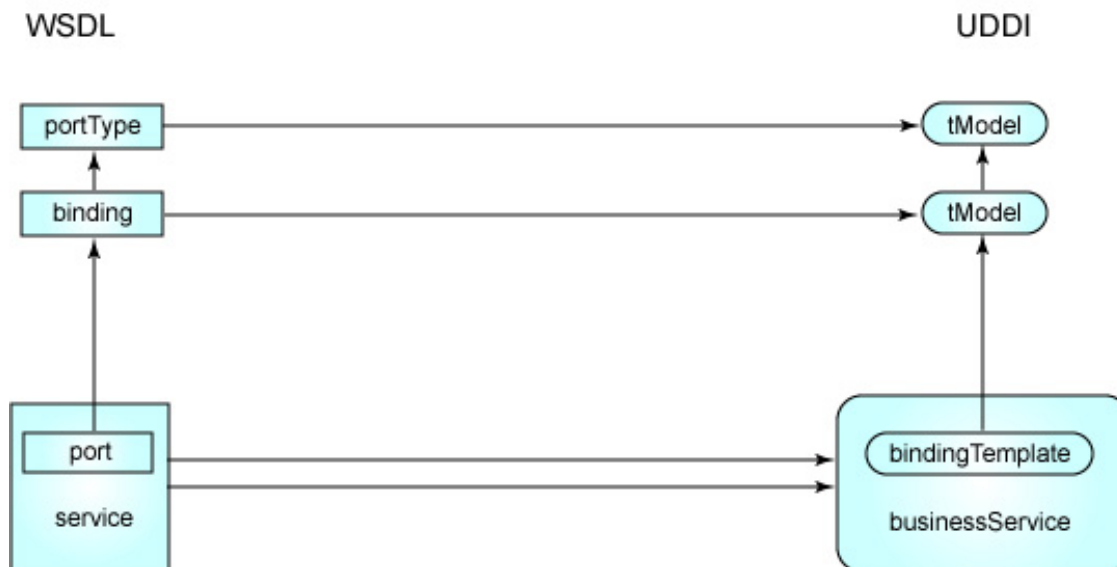
A WSDL port is represented by a UDDI bindingTemplate. The containment relationship between a WSDL service and its ports is exactly mirrored by the containment relationship between a UDDI businessService and its bindingTemplates.

Summary

The approach to mapping WSDL to UDDI described in the Technical Note is summarised in [Figure 1](#).

Figure 1. Summary of UDDI-WSDL Mapping

Mapping from WSDL to UDDI



Conclusion

This article has given an introduction to the background to the new Technical Note on using UDDI and WSDL, and described the main points of the approach described in the Technical Note.

Subsequent articles in this series will describe the types of queries that this new approach enables, present a more complex example, and show how to write Java applications to publish to UDDI in accordance with the Technical Note and how to issue typical queries against the new UDDI model of WSDL.

Resources

- "[Understanding WSDL in a UDDI registry, Part 1](#)" (*developerWorks*, September 2001) describes the previous Best Practice for using WSDL and UDDI together.
- "[Understanding WSDL in a UDDI registry, Part 2](#)" (*developerWorks*, September 2001) describes various scenarios relating to the previous Best Practice for using WSDL and UDDI together.

- "[Understanding WSDL in a UDDI registry, Part 3](#)" (*developerWorks*, November 2001) describes Java applications that implement the previous Best Practice for using WSDL and UDDI together.
- The [OASIS UDDI Specifications Technical Committee](#) is now responsible for UDDI standards.
- This article is based on the recent Technical Note "[Using WSDL in a UDDI Registry, Version 2.0](#)" which has recently been published as an official Technical Note by the OASIS UDDI Specifications Technical Committee.
- The previous Best Practice "[Using WSDL in a UDDI Registry, Version 1.08](#)" has been updated by the OASIS UDDI Specifications Technical Committee since the previous *developerWorks* articles were written.

About the author

John Colgrave is the architect of the IBM WebSphere UDDI Registry and a member of the OASIS UDDI Specifications Technical Committee. He is one of the authors of the new OASIS UDDI Technical Note on using UDDI and WSDL.



What do you think of this document?

Killer! (5) Good stuff (4) So-so; not bad (3) Needs work (2) Lame! (1)

Comments?

[IBM developerWorks](#) > [Web services](#)

developerWorks

[About IBM](#) | [Privacy](#) | [Legal](#) | [Contact](#)