



## UDDI Specifications TC

### Technical Note

#### Using WSDL in a UDDI Registry, Version 2.0

**Document identifier:**

uddi-spec-tc-tn-wsdl-20030319-wd

**Location:**

<http://www.oasis-open.org/committees/uddi-spec/doc/draft/uddi-spec-tc-tn-wsdl-20030319-wd.htm>

**Authors (alphabetically):**

John Colgrave, IBM [colgrave@uk.ibm.com](mailto:colgrave@uk.ibm.com)

Karsten Januszewski, Microsoft [karstenj@microsoft.com](mailto:karstenj@microsoft.com)

**Editors:**

Anne Thomas Manes, [anne@manes.net](mailto:anne@manes.net)

Tony Rogers, Computer Associates [tony.rogers@ca.com](mailto:tony.rogers@ca.com)

**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](mailto:uddi-spec@lists.oasis-open.org) list. Others should subscribe to and send comments to the [uddi-spec-comment@lists.oasis-open.org](mailto:uddi-spec-comment@lists.oasis-open.org) list. To subscribe, send an email message to [uddi-spec-comment-request@lists.oasis-open.org](mailto:uddi-spec-comment-request@lists.oasis-open.org) with the word "subscribe" as the body of the message.

## 26 Copyright

27 Copyright © OASIS Open March 2003. All Rights Reserved.

28 *This document and translations of it may be copied and furnished to others, and derivative*  
29 *works that comment on or otherwise explain it or assist in its implementation may be*  
30 *prepared, copied, published and distributed, in whole or in part, without restriction of any kind,*  
31 *provided that the above copyright notice and this paragraph are included on all such copies*  
32 *and derivative works. However, this document itself may not be modified in any way, such as*  
33 *by removing the copyright notice or references to OASIS, except as needed for the purpose*  
34 *of developing OASIS specifications, in which case the procedures for copyrights defined in*  
35 *the OASIS Intellectual Property Rights document must be followed, or as required to translate*  
36 *it into languages other than English.*

37 *The limited permissions granted above are perpetual and will not be revoked by OASIS or its*  
38 *successors or assigns.*

39 *This document and the information contained herein is provided on an "AS IS" basis and*  
40 *OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT*  
41 *LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL*  
42 *NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY*  
43 *OR FITNESS FOR A PARTICULAR PURPOSE.*

## Table of Contents

45	1	Introduction .....	6
46	1.1	Goals and Requirements.....	6
47	1.2	Relationship to Version 1 Best Practice .....	7
48	1.3	Terminology.....	7
49	2	Mapping Two Data Models: WSDL & UDDI .....	8
50	2.1	WSDL Data Model .....	8
51	2.1.1	portType .....	8
52	2.1.2	binding.....	8
53	2.1.3	service and port.....	9
54	2.1.4	import .....	9
55	2.2	UDDI Data Model.....	9
56	2.2.1	tModels.....	9
57	2.2.2	businessService & bindingTemplate.....	10
58	2.3	Mapping WSDL and UDDI .....	10
59	2.3.1	Mapping Overview .....	10
60	2.3.2	Comparison to Version 1 Mapping .....	11
61	2.3.3	New Canonical tModels .....	11
62	2.3.4	General Conventions .....	11
63	2.3.5	Support for Multiple UDDI API Versions .....	12
64	2.3.6	References to WSDL Components.....	12
65	2.3.7	WSDL Extensibility Elements .....	12
66	2.3.8	Support for WSDL Implementation Documents .....	12
67	2.4	Mapping WSDL 1.1 in UDDI V2.....	13
68	2.4.1	wsdl:portType → uddi:tModel.....	13
69	2.4.2	wsdl:binding → uddi:tModel .....	13
70	2.4.3	wsdl:service → uddi:businessService .....	15
71	2.4.4	wsdl:port → uddi:bindingTemplate .....	16
72	2.4.5	wsdl:port Address Extensions → uddi:bindingTemplate.....	17
73	2.5	Differences in mapping WSDL 1.1 in UDDI V3 .....	17
74	2.5.1	Mandatory Differences.....	18
75	2.5.2	Optional Extensions.....	18
76	2.5.3	Comparison to wsdlDeployment in UDDI V3 Specification.....	18
77	3	A Complete Example.....	19
78	3.1	WSDL Sample .....	19
79	3.2	UDDI V2 Model.....	20
80	3.2.1	UDDI portType tModel .....	20
81	3.2.2	UDDI binding tModel.....	20
82	3.2.3	UDDI businessService and bindingTemplate.....	21
83	3.3	Sample V2 Queries.....	21
84	3.3.1	Find tModel for portType name .....	22
85	3.3.2	Find bindings for portType .....	22
86	3.3.3	Find Implementations of portType .....	22
87	3.3.4	Find implementations of binding.....	22
88	3.3.5	Find SOAP Implementations of portType .....	22
89	3.3.6	Find SOAP/HTTP Implementations of portType .....	23

90	3.3.7 Find the portType of a binding.....	23
91	4 References.....	24
92	4.1 Normative .....	24
93	A External WSDL Implementation Documents .....	25
94	A.1 Capturing The URL .....	25
95	A.2 Obtaining the Port Address from WSDL.....	25
96	A.3 Querying Services that use a WSDL Implementation Document .....	25
97	B Canonical tModels.....	26
98	B.1 WSDL Entity Type tModel.....	26
99	B.1.1 Design Goals .....	26
100	B.1.2 Definition.....	26
101	B.1.3 Valid Values.....	26
102	B.1.4 Example of Use.....	27
103	B.2 XML Namespace tModel .....	27
104	B.2.1 Design Goals .....	27
105	B.2.2 Definition.....	27
106	B.2.3 Valid Values.....	28
107	B.2.4 Example of Use.....	28
108	B.3 XML Local Name tModel .....	28
109	B.3.1 Design Goals .....	28
110	B.3.2 Definition.....	28
111	B.3.3 Valid Values.....	29
112	B.3.4 Example of Use.....	29
113	B.4 WSDL portType Reference tModel .....	29
114	B.4.1 Design Goals .....	29
115	B.4.2 Definition.....	29
116	B.4.3 Valid Values.....	30
117	B.4.4 Example of Use.....	30
118	B.5 SOAP Protocol tModel.....	30
119	B.5.1 Design Goals .....	30
120	B.5.2 Definition.....	30
121	B.5.3 Example of Use.....	31
122	B.6 HTTP Protocol tModel .....	31
123	B.6.1 Design Goals .....	31
124	B.6.2 Definition.....	31
125	B.6.3 Example of Use.....	32
126	B.7 Protocol Categorization .....	32
127	B.7.1 Design Goals .....	32
128	B.7.2 Definition.....	32
129	B.7.3 Example of Use.....	33
130	B.8 Transport Categorization .....	33
131	B.8.1 Design Goals .....	33
132	B.8.2 Definition.....	33
133	B.8.3 Example of Use.....	34
134	B.9 WSDL Address tModel .....	35
135	B.9.1 Design Goals .....	35
136	B.9.2 Definition.....	35
137	B.9.3 Valid Values.....	35

138	B.9.4 Example of Use.....	35
139	C Using XPointer in overviewURL.....	36
140	C.1 XPointer Syntax.....	36
141	C.1.1 Example of Use.....	36
142	D Acknowledgments.....	37
143	E Revision History.....	38
144	F Notices.....	39
145		

146

## 1 Introduction

147 The Universal Description, Discovery & Integration (UDDI) specification provides a platform-  
148 independent way of describing and discovering Web services and Web service providers. The  
149 UDDI data structures provide a framework for the description of basic service information, and  
150 an extensible mechanism to specify detailed service access information using any standard  
151 description language. Many such languages exist in specific industry domains and at different  
152 levels of the protocol stack. The Web Services Description Language (WSDL) is a general  
153 purpose XML language for describing the interface, protocol bindings, and the deployment  
154 details of network services. WSDL complements the UDDI standard by providing a uniform  
155 way of describing the abstract interface and protocol bindings of arbitrary network services.  
156 The purpose of this document is to clarify the relationship between the two and to describe a  
157 recommended approach to mapping WSDL descriptions to the UDDI data structures.  
158 Consistent and thorough WSDL mappings are critical to the utility of UDDI.

### 1.1 Goals and Requirements

159 The primary goals of this mapping are:

- 161 1. To enable the automatic registration of WSDL definitions in UDDI
- 162 2. To enable precise and flexible UDDI queries based on specific WSDL artifacts and  
163 metadata
- 164 3. To maintain compatibility with the mapping described in the *Using WSDL in a UDDI*  
165 *Registry, Version 1.08* [1] Best Practice document
- 166 4. To provide a consistent mapping for UDDI Version 2 and UDDI Version 3
- 167 5. To support any logical and physical structure of WSDL description

168 This mapping prescribes a consistent methodology to map WSDL 1.1 artifacts to UDDI  
169 structures. It describes an approach that represents reusable, abstract Web service artifacts,  
170 (WSDL portTypes and WSDL bindings) and Web service implementations (WSDL services  
171 and ports). Tools can use this mapping to generate UDDI registrations automatically from  
172 WSDL descriptions.

173 This mapping captures sufficient information from the WSDL documents to allow precise  
174 queries for Web services information without further recourse to the source WSDL  
175 documents, and to allow the appropriate WSDL documents to be retrieved once a match has  
176 been found. Given that the source WSDL documents can be distributed among the publishers  
177 using a UDDI registry, a UDDI registry provides a convenient central point where such  
178 queries can be executed.

179 This mapping enables the following types of queries for both design-time and run-time  
180 discovery:

- 181 • Given the namespace and/or local name of a wsdl:portType, find the tModel that  
182 represents that portType.
- 183 • Given the namespace and/or local name of a wsdl:binding, find the tModel that  
184 represents that binding.
- 185 • Given a tModel representing a portType, find all tModels representing bindings for  
186 that portType.
- 187 • Given a tModel representing a portType, find all bindingTemplates that represent  
188 implementations of that portType.
- 189 • Given a tModel representing a binding, find all bindingTemplates that represent  
190 implementations of that binding.

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

195 Although the UDDI V3 data model is slightly different from the UDDI data model, this mapping  
196 ensures that the same information is captured in both versions.

## 197 **1.2 Relationship to Version 1 Best Practice**

198 This document builds on *Using WSDL in a UDDI Registry, Version 1.08*, providing an  
199 expanded modeling practice that encompasses the flexibility of WSDL. The primary difference  
200 between this mapping and the one described in the existing Best Practice is that this mapping  
201 provides a methodology to represent individual Web services artifacts.

202 As a Technical Note, this document does not replace the Version 1 Best Practice. If the  
203 additional flexibility is not required, the existing Best Practice can still be used, particularly  
204 when the UDDI artifacts are published manually.

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

208 A final goal is to be compatible with the existing Best Practice in that a tModel representing a  
209 WSDL binding published using the approach described in this document should be usable by  
210 a client that uses the Version 1 Best Practice approach.

## 211 **1.3 Terminology**

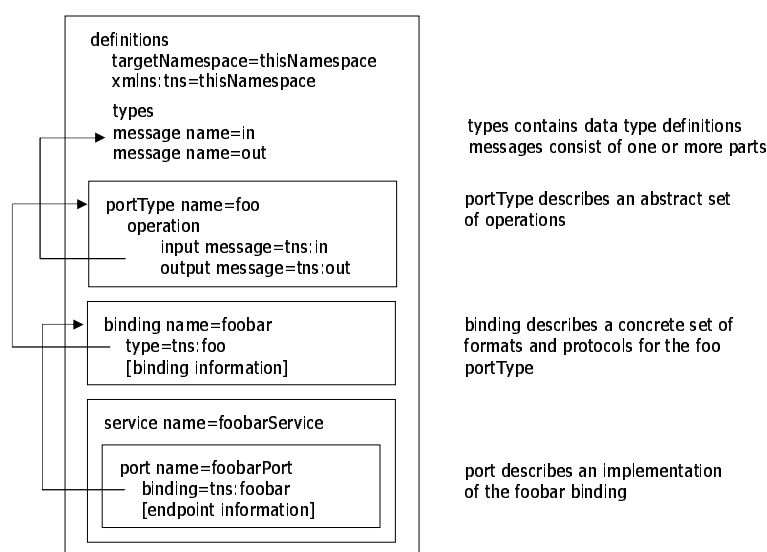
212 The key words *must*, *must not*, *required*, *shall*, *shall not*, *should*, *should not*, *recommended*,  
213 *may*, and *optional* in this document are to be interpreted as described in [RFC2119].

## 214 2 Mapping Two Data Models: WSDL & UDDI

215 A brief discussion of the two respective data models, WSDL and UDDI, follows. For a  
216 complete explanation of these specifications, see [2], [3], and [4].

### 217 2.1 WSDL Data Model

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



220

#### 221 2.1.1 portType

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

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

#### 233 2.1.2 binding

234 WSDL portTypes are abstract Web service descriptions and do not specify information about  
235 the encoding and transport protocols used to transmit the messages. To specify encoding and  
236 transport protocol details in WSDL, one must define a second construct, known as a binding.  
237 A WSDL binding specifies a specific set of encoding and transport protocols that may be used  
238 to communicate with an implementation of a particular WSDL portType. A WSDL binding  
239 specifies its portType through a QName reference. The referenced portType may or may not  
240 be in the same target namespace as the binding itself. Again, a single WSDL file may contain  
241 multiple bindings. For example, a WSDL file may describe multiple protocol bindings for a



242 single portType. Like a portType, a binding is uniquely identified by the combination of its  
243 local name and the target namespace of the definitions element that contains the binding.  
244 As with portTypes, WSDL bindings are abstract definitions and do not represent a Web  
245 service implementation. Multiple Web services may implement the same WSDL binding.

### 246 **2.1.3 service and port**

247 Finally, WSDL defines a Web service implementation as a service with a collection of named  
248 ports. Each port implements a particular portType using the protocols defined by a named  
249 binding. A service may expose multiple ports in order to make a single portType available  
250 over multiple protocols. A service may also expose multiple ports in order to expose more  
251 than one portType from a single logical entity. A WSDL port specifies the binding it  
252 implements through a QName reference. The referenced binding may or may not be in the  
253 same target namespace as the port itself. A single WSDL file may contain multiple services. A  
254 service is uniquely identified by the combination of its local name and the target namespace  
255 of the definitions element that contains the service. Likewise, a port is uniquely identified by  
256 the combination of its local name and the target namespace of the definitions element that  
257 contains the port.

### 258 **2.1.4 import**

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

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

## 267 **2.2 UDDI Data Model**

268 As an aid to understanding the sections ahead, we provide here a brief overview of two UDDI  
269 data structures that are particularly relevant to the use of WSDL in the context of a UDDI  
270 registry: the tModel and the businessService.

### 271 **2.2.1 tModels**

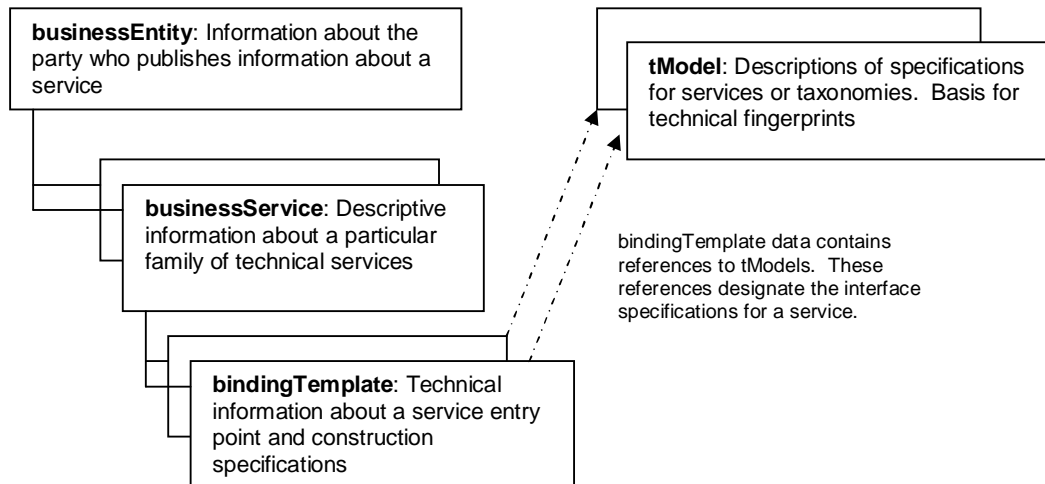
272 TModels are often referred to as service type definitions. TModels represent unique concepts  
273 or constructs. They are used to describe compliance with a specification, a concept, or a  
274 shared design. TModels have various uses in the UDDI registry. In the case of mapping  
275 WSDL-described Web services, tModels have two uses. First, tModels are used to represent  
276 technical specifications such as service types, bindings, and wire protocols. Second, tModels  
277 are used to implement taxonomies that are used to identify or categorize technical  
278 specifications and services. This Technical Note defines a set of specification and taxonomy  
279 tModels that are used when mapping WSDL entities to UDDI entities. These tModels are  
280 defined in Appendix B.

281 When a particular specification is registered in the UDDI registry as a tModel, it is assigned a  
282 unique key, called a tModelKey. This key is used by other UDDI entities to reference the  
283 tModel, for example to indicate compliance with the specification.

284 Each specification tModel contains an overviewURL, which provides the address of the  
285 specification itself, for example, a WSDL file.

286 Additional metadata can be associated with a specification tModel using any number of  
287 identifier and categorization taxonomies. Identifiers are grouped in a construct called an  
288 identifierBag, and categories are grouped in a construct called a categoryBag. These bags  
289 contain a set of keyedReference elements. Each keyedReference specifies the tModelKey of  
290 the taxonomy tModel and a name/value pair that specifies the metadata. For example, a  
291 keyedReference referencing the namespace taxonomy can be used to specify a WSDL

292 namespace. The metadata values specified in keyedReference elements can be used as  
293 selection criteria when searching UDDI.



## 294 2.2.2 businessService & bindingTemplate

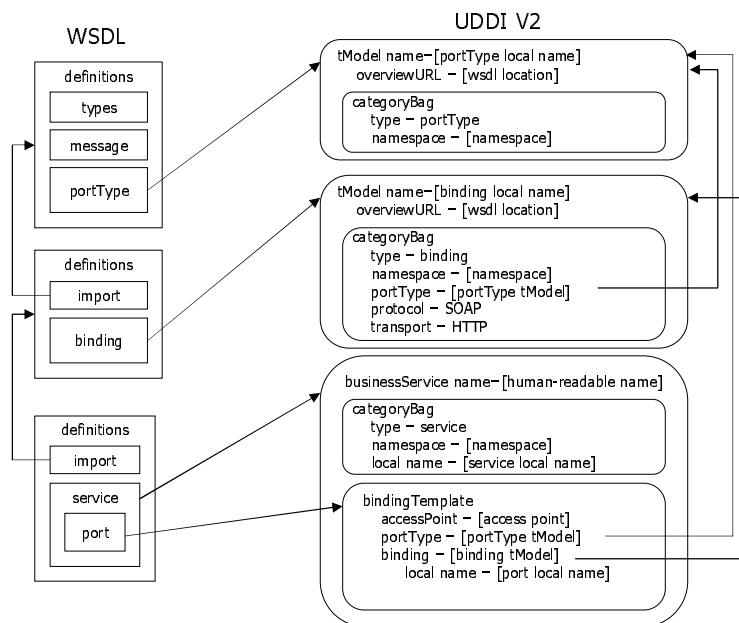
295 Services are represented in UDDI by the businessService data structure, and the details of  
296 how and where the service is accessed are provided by one or more bindingTemplate  
297 structures. The businessService might be thought of as a logical container of services. The  
298 bindingTemplate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

## 300 2.3 Mapping WSDL and UDDI

301 WSDL is designed to support modular and reusable definitions, and each definition artifact  
302 has certain relationships with other definition artifacts. As described in Section 1.1, the goals  
303 of this technical note and the mapping it defines are to enable the automatic registration of  
304 WSDL definitions in UDDI, to enable precise and flexible UDDI queries based on specific  
305 WSDL artifacts and metadata, to maintain compatibility with the Version 1 Best Practice  
306 methodology, and to ease migration from UDDI V2 to UDDI V3. The mapping itself addresses  
307 the first goal. The second goal provides the rationale for the methodology used in this  
308 mapping. In order to support queries based on specific WSDL artifacts and metadata, this  
309 mapping must be able to represent the individual WSDL artifacts and the relationships  
310 between artifacts. This goal also provides the rationale for the amount of information that  
311 must be captured in UDDI. Additional information must also be included in some cases to  
312 support the third goal. To address the fourth goal, the information captured in the two  
313 mappings is as consistent as possible.

### 314 2.3.1 Mapping Overview

315 This mapping describes a methodology for mapping WSDL 1.1 definitions to the UDDI V2 and  
316 UDDI V3 data models. The methodology maps each WSDL artifact to a separate UDDI entity,  
317 accurately representing the “building block” design of WSDL descriptions. wsdl:portType and  
318 wsdl:binding elements map to uddi:tModel entities, wsdl:service elements map to  
319 uddi:businessService entities and wsdl:port elements map to uddi:bindingTemplate entities.  
320 KeyedReferences provide a mechanism to express additional metadata and to represent a  
321 relationship between two UDDI entities.



322

### 323 2.3.2 Comparison to Version 1 Mapping

324 One important thing to note about this mapping, especially as compared to the mapping  
 325 described in the Version 1 Best Practice, is that this approach may map a single WSDL file to  
 326 multiple tModels. For example, a single WSDL file that contains one portType definition and  
 327 two binding definitions will map to three distinct tModels in UDDI. This approach differs from  
 328 the Version 1 Best Practice, which would map the entire WSDL file to a single tModel. The  
 329 rationale for this new mapping decision is to more effectively represent the modularity and  
 330 reusability of WSDL artifacts in UDDI. A Web service implementation might implement only  
 331 one of the bindings described in a WSDL file. By decomposing WSDL into multiple tModels,  
 332 one can accurately model in UDDI exactly which portTypes and bindings a given Web service  
 333 implementation supports, as opposed to being constrained to asserting that a Web service  
 334 always supports the entirety of the WSDL file.

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

### 340 2.3.3 New Canonical tModels

341 This mapping introduces a number of canonical tModels that are used to represent WSDL  
 342 metadata and relationships. These tModels, including the WSDL Entity Type tModel, the XML  
 343 Namespace tModel, the XML Local Name tModel, the WSDL portType Reference tModel, the  
 344 WSDL URL Reference tModel, the SOAP Protocol tModel, the HTTP Protocol tModel, the  
 345 Protocol Categorization tModel, the Transport Categorization tModel, the WSDL URL tModel,  
 346 and the WSDL Address tModel, are described in Appendix B. These tModels **MUST** be  
 347 registered in the UDDI registry to support this mapping. As both V1/V2 and V3 keys are given  
 348 for these tModels, their keys should be treated as evolved keys.

### 349 2.3.4 General Conventions

350 In this mapping, each WSDL artifact is mapped to its corresponding UDDI entity. A set of  
 351 keyedReference elements is added to each UDDI entity to capture additional metadata. In  
 352 order to support the requirements outlined in Section 1.1, the following metadata is captured  
 353 for each entity:

- 354 • The type of WSDL entity being defined (i.e., portType, binding, service, or port)
- 355 • The target namespace of the WSDL definitions file that defines the WSDL entity
- 356 • The local name of the WSDL entity being defined
- 357 • The location of the WSDL file that defines the WSDL entity is captured for portType,
- 358 binding and, optionally, service entities.

359 Any relationships and dependencies between entities must also be captured. For example, a  
360 tModel that represents a binding provides a reference to the tModel that represents the  
361 portType implemented by the binding.

362 To maintain compatibility with the Version 1 Best Practice mapping, certain UDDI entities are  
363 also characterized as being of type "wsdlSpec".

### 364 **2.3.5 Support for Multiple UDDI API Versions**

365 The mapping described is designed to appear the same whichever version of the UDDI API is  
366 used to access it. There are differences that are mandated by the differences in the API  
367 versions, and such differences are noted in the appropriate sections.

368 The V3 API also introduces some optional features that are not visible to the older APIs, and  
369 some guidance is given as to the usage of these optional features.

### 370 **2.3.6 References to WSDL Components**

371 A UDDI entity normally references technical specifications using the overviewURL element.  
372 As noted above, in this mapping a single WSDL file may map to multiple tModels, and each  
373 tModel refers to a particular WSDL entity within the file. The particular WSDL entity is  
374 uniquely identified by the combination of its local name and the target namespace of the  
375 definitions element that contains the WSDL entity. This identity information SHOULD be  
376 determined from the metadata contained within the entity's categoryBag. Alternatively, the  
377 overviewURL value MAY contain a fragment identifier that identifies the particular WSDL  
378 entity. If the optional fragment identifier is used, then the value of the overviewURL MUST  
379 conform to the syntax described in Appendix C.

### 380 **2.3.7 WSDL Extensibility Elements**

381 WSDL uses extensibility elements to describe technology-specific information within a WSDL  
382 definition. Extensibility elements may be included under many of the WSDL elements. The  
383 only extensibility elements that are relevant to this mapping are binding and port extensions,  
384 specifically the extensibility elements that can be added to the wsdl:binding and wsdl:port  
385 elements. The first of these is used to declare particular protocols and message formats; the  
386 second is to provide address information.

387 Information from these extensibility elements is mapped to the tModel for a wsdl:binding and  
388 a bindingTemplate. The mappings defined in this document include details on the SOAP 1.1  
389 and HTTP GET/POST bindings defined in the WSDL 1.1 W3C Note. The mappings also  
390 describe how other bindings should be incorporated into the UDDI mapping.

### 391 **2.3.8 Support for WSDL Implementation Documents**

392 In the context of this Technical Note, a WSDL Implementation Document is a WSDL  
393 document that contains at least one wsdl:service element and its associated wsdl:port  
394 elements. There are two options for how this implementation information is described in  
395 UDDI:

- 396 1. The information in the UDDI model is the authoritative information and there is no  
397 reference to a WSDL Implementation Document.
- 398 2. A reference to an external WSDL Implementation Document can be stored in UDDI  
399 and the remaining information in UDDI is used to describe the appropriate element in  
400 the external WSDL resource.

401 The mapping described in the body of this document corresponds to the first option  
402 above, and that is assumed to be the default mapping. The second option is described in  
403 Appendix A.

## 404 **2.4 Mapping WSDL 1.1 in UDDI V2**

405 This section describes a detailed mapping of WSDL 1.1 artifacts to the UDDI V2 data model.

### 406 **2.4.1 wsdl:portType → uddi:tModel**

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

408 The minimum information that must be captured about a portType is its entity type, its local  
409 name, its namespace, and the location of the WSDL document that defines the portType.  
410 Capturing the entity type enables users to search for tModels that represent portType  
411 artifacts. Capturing the local name, namespace, and WSDL location enables users to locate  
412 the definition of the specified portType artifact.

413 The wsdl:portType information is captured as follows:

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

416 The tModel MUST contain a categoryBag, and the categoryBag MUST contain at least the  
417 following keyedReference elements:

- 418 1. A keyedReference with a tModelKey of the WSDL Entity Type taxonomy and a  
419 keyValue of "portType".
- 420 2. A keyedReference with a tModelKey of the XML Namespace taxonomy and a  
421 keyValue of the target namespace of the wsdl:definitions element that contains the  
422 wsdl:portType.<sup>1</sup>

423 The tModel MUST contain an overviewDoc with an overviewURL containing the location of  
424 the WSDL file that describes the wsdl:portType.

#### 425 **2.4.1.1 Summary of Mapping of wsdl:portType**

WSDL	UDDI
portType	tModel (categorized as portType)
Namespace of portType	keyedReference in categoryBag
Local name of portType	tModel name
Location of WSDL file	overviewURL

426

### 427 **2.4.2 wsdl:binding → uddi:tModel**

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

429 The minimum information that must be captured about a binding is its entity type, its local  
430 name, its namespace, the location of the WSDL document that defines the binding, the  
431 portType that it implements, its protocol, and, optionally, the transport information. Capturing  
432 the entity type enables users to search for tModels that represent binding artifacts. Capturing  
433 the local name, namespace, and WSDL location enables users to locate the definition of the  
434 specified binding artifact. The link to the portType enables users to search for bindings that  
435 implement a particular portType.

---

<sup>1</sup> 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 an XML Namespace keyedReference, and queries for these tModels based solely on the tModel name could return multiple results because no namespace can be specified.

436 A wsdl:binding corresponds to a WSDL service interface definition as defined by the mapping  
437 in the Version 1 Best Practice. To maintain compatibility with the previous mapping, the  
438 binding must also be characterized as type "wsdlSpec".

439 The wsdl:binding information is captured as follows:

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

442 The tModel MUST contain a categoryBag, and the categoryBag MUST contain at least the  
443 following keyedReference elements:

- 444 1. A keyedReference with a tModelKey of the WSDL Entity Type taxonomy and a  
445 keyValue of "binding".
- 446 2. A keyedReference with a tModelKey of the XML Namespace taxonomy and a  
447 keyValue of the target namespace of the wsdl:definitions element that contains the  
448 wsdl:binding.
- 449 3. A keyedReference with a tModelKey of the WSDL portType Reference taxonomy and  
450 a keyValue of the tModelKey that models the wsdl:portType to which the wsdl:binding  
451 relates.
- 452 4. A keyedReference with a tModelKey of the UDDI Types taxonomy and a keyValue of  
453 "wsdlSpec" for backward compatibility<sup>2</sup>.
- 454 5. One or two keyedReferences as required to capture the protocol and optionally the  
455 transport information – refer to the next section.

456 The tModel MUST contain an overviewDoc with an overviewURL containing the location of  
457 the WSDL file that describes the wsdl:binding.

#### 458 **2.4.2.1 wsdl:binding Extensions**

459 Information about the protocol and transport, if applicable, specified in an extension to the  
460 wsdl:binding is used to categorize the binding tModel as described in the following sections.  
461 This information is specified using two of the taxonomies defined in this Technical Note:

- 462 1. Protocol Categorization
- 463 2. Transport Categorization

464 The valid values for the Protocol Categorization taxonomy are tModelKeys of tModels that are  
465 categorized as protocol tModels. Similarly, the valid values for the Transport Categorization  
466 taxonomy are tModelKeys of tModels that are categorized as transport tModels.

467 The reason for having these two categorization schemes that take tModel keys as values is to  
468 allow other standard or proprietary protocols and transports to be defined and used in the  
469 same way as the standard SOAP and HTTP protocols and transport.

#### 470 **2.4.2.1.1 soap:binding**

471 If the wsdl:binding contains a soap:binding extensibility element from the  
472 <http://schemas.xmlsoap.org/wsdl/soap/> namespace then the categoryBag MUST include a  
473 keyedReference with a tModelKey of the Protocol Categorization taxonomy and a keyValue of  
474 the tModelKey of the SOAP Protocol tModel.

475 If the value of the transport attribute of the soap:binding element is  
476 <http://schemas.xmlsoap.org/soap/http> then the categoryBag MUST include a keyedReference  
477 with a tModelKey of the Transport Categorization taxonomy and a keyValue of the tModelKey  
478 of the HTTP Transport tModel.

479 If the value of the transport attribute is anything else, then the bindingTemplate MUST include  
480 an additional keyedReference with a tModelKey of the Transport Categorization taxonomy  
481 and a keyValue of the tModelKey of an appropriate transport tModel.

---

<sup>2</sup> 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.

482 **2.4.2.1.2 http:binding**

483 If the wsdl:binding contains an http:binding extensibility element from the  
484 http://schemas.xmlsoap.org/wsdl/http/ namespace then the categoryBag MUST include a  
485 keyedReference with a tModelKey of the Protocol Categorization taxonomy and a keyValue of  
486 the tModelKey of the HTTP Protocol tModel.

487 Note that this is a different tModel from the HTTP Transport tModel, and in this case there is  
488 no separate transport tModel, and therefore no keyedReference in the categoryBag from the  
489 Transport Categorization taxonomy.

490 **2.4.2.1.3 Other wsdl:binding Extensions**

491 Other wsdl:binding extensibility elements are handled in a similar fashion. It is assumed that  
492 vendors who provide other bindings will provide the appropriate protocol and transport  
493 tModels.

494 **2.4.2.2 Summary of Mapping of wsdl:binding**

WSDL	UDDI
binding	tModel (categorized as binding and wsdlSpec)
Namespace of binding	keyedReference in categoryBag
Local name of binding	tModel name
Location of WSDL file	overviewURL
portType binding relates to	keyedReference in categoryBag
Protocol from binding extension	keyedReference in categoryBag
Transport from binding extension (if there is one)	keyedReference in categoryBag

495

496 **2.4.3 wsdl:service → uddi:businessService**

497 A wsdl:service MUST be modeled as a uddi:businessService. An existing businessService  
498 MAY be used or a new businessService MAY be created<sup>3</sup>. Only one wsdl:service can be  
499 modeled by an individual uddi:businessService.

500 The minimum information that must be captured about a service is its entity type, its local  
501 name, its namespace, and the list of ports that it supports. Capturing the entity type enables  
502 users to search for services that are described by a WSDL definition. The list of ports  
503 provides access to the technical information required to consume the service.

504 The wsdl:service information is captured as follows:

505 If a new businessService is created, the uddi:name of this businessService SHOULD be a  
506 human readable name, although if no human readable name is specified, it MUST be the  
507 value of the name attribute of the wsdl:service<sup>4</sup>.

---

<sup>3</sup> WSDL permits any arbitrary group of ports to be collected into a single service, therefore a wsdl:service may not directly correspond to a uddi:businessService. As a best practice for this mapping, a wsdl:service SHOULD contain a collection of associated ports that relate to a single logical business service, for example, a collection of ports that implement alternate bindings for a particular portType. A wsdl:service SHOULD NOT contain multiple ports that do not relate to a single logical business service.

508 The businessService MUST contain a categoryBag, and the categoryBag MUST contain at  
509 least the following keyedReference elements:

- 510 1. A keyedReference with a tModelKey of the WSDL Entity Type taxonomy and a  
511 keyValue of "service".
- 512 2. A keyedReference with a tModelKey of the XML Namespace taxonomy and a  
513 keyValue of the target namespace of the wsdl:definitions element that contains the  
514 wsdl:service.
- 515 3. A keyedReference with a tModelKey of the XML Local Name taxonomy and a  
516 keyValue that is the value of the name attribute of the wsdl:service.

517 The bindingTemplates element of the businessService MUST include bindingTemplate  
518 elements that model the ports of the service, as described in the following sections.

### 519 2.4.3.1 Summary of Mapping

WSDL	UDDI
Service	businessService (categorized as service)
Namespace of Service	keyedReference in categoryBag
Local Name of Service	keyedReference in categoryBag; optionally also the name of the service

### 520 2.4.4 wsdl:port → uddi:bindingTemplate

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

522 The minimum information that must be captured about a port is the binding that it implements,  
523 the portType that it implements, and its local name<sup>5</sup>.

524 By capturing the binding, users can search for services that implement a specific binding. By  
525 capturing the portType, users can search for services that implement a particular portType  
526 without necessarily knowing the specific binding implemented by the service.

527 The wsdl:port information is captured as follows:

528 The bindingTemplate tModelInstanceDetails element MUST contain at least the following  
529 tModelInstanceInfo elements:

- 530 1. A tModelInstanceInfo with a tModelKey of the tModel that models the wsdl:binding  
531 that this port implements. The instanceParms of this tModelInstanceInfo MUST  
532 contain the wsdl:port local name.
- 533 2. A tModelInstanceInfo with a tModelKey of the tModel that models the wsdl:portType.

### 534 2.4.4.1 Summary of Mapping

WSDL	UDDI
port	bindingTemplate
Namespace	Captured in keyedReference of the containing businessService
Local Name of port	instanceParms of the tModelInstanceInfo relating to the tModel for the binding

---

<sup>4</sup> Users searching for a wsdl:service MUST NOT assume that the businessService name is the same as the wsdl:service local name. Because an existing businessService could be used, the wsdl:service local name MUST be specified as a keyedReference in the categoryBag.

<sup>5</sup> The namespace is captured in the businessService element.



Binding implemented by port	tModelInstanceInfo with tModelKey of the tModel corresponding to the binding
portType implemented by port	tModelInstanceInfo with tModelKey of the tModel corresponding to the portType

535

## 536 **2.4.5 wsdl:port Address Extensions → uddi:bindingTemplate**

537 The uddi:bindingTemplate MUST contain address information for the Web service. This  
538 information comes from the wsdl:port address extensibility element.

### 539 **2.4.5.1 soap:address → uddi:accessPoint**

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

542 The soap:address information is captured as follows:

- 543 • The accessPoint value MUST be the value of the location attribute of the  
544 soap:address element.
- 545 • The URLType attribute of the accessPoint MUST correspond to the transport  
546 specified by the soap:binding, or “other” if no correspondence exists. In the case of  
547 the HTTP transport, for example, the URLType attribute MUST be “http”.

548 If “other” is used then a tModelInstanceInfo element referencing the appropriate vendor-  
549 defined transport tModel MUST be added to the bindingTemplate.

### 550 **2.4.5.2 http:address → uddi:accessPoint**

551 An http:address MUST be modeled as a uddi:accessPoint in the uddi:bindingTemplate that  
552 models the wsdl:port that contains the http:address.

553 The http:address information is captured as follows:

- 554 • The accessPoint value MUST be the value of the location attribute of the http:address  
555 element.
- 556 • The URLType attribute of the accessPoint MUST be “http”.

### 557 **2.4.5.3 Other wsdl:port Address Extensions**

558 Any other address extensibility element MUST be modeled as a uddi:accessPoint in the  
559 uddi:bindingTemplate that models the wsdl:port that contains the address extensibility  
560 element.

561 The address information is captured as follows:

- 562 • The accessPoint value MUST be the value of the location attribute of the address  
563 extensibility element. If the value of the location attribute cannot be mapped to the  
564 accessPoint value then the WSDL Implementation Document approach must be  
565 used. See Appendix A for further information.
- 566 • The URLType attribute of the accessPoint MUST correspond to the transport protocol  
567 associated with the URL, or “other” if none of the defined values of the attribute are  
568 appropriate.

## 569 **2.5 Differences in mapping WSDL 1.1 in UDDI V3**

570 This section describes the differences in the UDDI V3 view of the model that are a  
571 consequence of mandatory items in the UDDI V3 Specification and some optional extensions  
572 that can only be used with UDDI V3.

### 573 **2.5.1 Mandatory Differences**

574 The mandatory differences are:

- 575 1. Entities will have V3 keys rather than V2 keys.
- 576 2. An accessPoint has a useType attribute rather than a URLType attribute.

### 577 **2.5.2 Optional Extensions**

578 The optional extensions are:

- 579 1. Entities can have publisher-assigned keys.
- 580 2. A bindingTemplate can have a categoryBag. If a categoryBag is used, it MUST  
581 contain at least the following keyedReferences:
  - 582 a. A keyedReference with a tModelKey of the WSDL Entity Type taxonomy and  
583 a keyValue of "port".
  - 584 b. A keyedReference with a tModelKey of the XML Namespace taxonomy and a  
585 keyValue of the target namespace of the wsdl:definitions element that  
586 contains the wsdl:port.
  - 587 c. A keyedReference with a tModelKey of the XML Local Name taxonomy and a  
588 keyValue of the local name of the wsdl:port.
- 589 3. An overviewURL can have an optional useType attribute, and a standard value of  
590 "wsdlInterface" has been defined to indicate "an abstract interface document". This  
591 mapping assumes that "wsdlInterface" can be used with tModels that represent both  
592 portTypes and bindings.

### 593 **2.5.3 Comparison to wsdlDeployment in UDDI V3 Specification**

594 The UDDI V3 specification includes support for wsdlDeployment, which appears as both a  
595 value for the useType attribute of an accessPoint and as a categorization of a  
596 bindingTemplate. Use of wsdlDeployment is not compatible with this Technical Note as it  
597 assumes that no modeling of the WSDL is performed, nothing is known about the WSDL  
598 other than its URL.

599

## 3 A Complete Example

600 Consider the following WSDL sample based on the WSDL file presented in the WSDL 1.1  
601 specification.<sup>6</sup> This example shows how this one WSDL file is decomposed into two tModels  
602 (one for the portType and one for the binding) and one businessService with one  
603 bindingTemplate. It then shows the kinds of UDDI API queries that can be used for the  
604 purpose of discovery.

605

### 3.1 WSDL Sample

```
606 <?xml version="1.0" encoding="utf-8" ?>
607 <definitions
608     name="StockQuote"
609     targetNamespace="http://example.com/stockquote/"
610     xmlns:tns="http://example.com/stockquote/"
611     xmlns:xsd="http://example.com/stockquote/schema/"
612     xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
613     xmlns="http://schemas.xmlsoap.org/wsdl/"
614 <import
615     namespace="http://example.com/stockquote/schema/"
616     location="http://location/schema.xsd" />
617 <message name="GetLastTradePriceInput">
618     <part name="body" element="xsd:TradePriceRequest" />
619 </message>
620 <message name="GetLastTradePriceOutput">
621     <part name="body" element="xsd:TradePrice" />
622 </message>
623 <portType name="StockQuotePortType">
624     <operation name="GetLastTradePrice">
625         <input message="tns:GetLastTradePriceInput" />
626         <output message="tns:GetLastTradePriceOutput" />
627     </operation>
628 </portType>
629 <binding name="StockQuoteSoapBinding" type="tns:StockQuotePortType">
630     <soap:binding style="document"
631         transport="http://schemas.xmlsoap.org/soap/http" />
632     <operation name="GetLastTradePrice">
633         <soap:operation soapAction="http://example.com/GetLastTradePrice"/>
634         <input>
635             <soap:body use="literal" />
636         </input>
637         <output>
638             <soap:body use="literal" />
639         </output>
640     </operation>
641 </binding>
642
643 <service name="StockQuoteService">
644     <documentation>My first service</documentation>
645     <port name="StockQuotePort" binding="tns:StockQuoteSoapBinding">
646         <soap:address location="http://location/sample"/>
647     </port>
648 </service>
649
650 </definitions>
```

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

---

<sup>6</sup> The WSDL sample in the WSDL 1.1 spec has an error (the port references the wrong binding QName). This WSDL sample has been corrected.

## 654 3.2 UDDI V2 Model

### 655 3.2.1 UDDI portType tModel

656 The WSDL portType entity maps to a tModel. The tModel name is the same as the WSDL  
657 portType local name. The tModel contains a categoryBag that specifies the WSDL  
658 namespace, and it indicates that the tModel is of type "portType". The overviewDoc provides  
659 a pointer to WSDL file.

```
660 <tModel tModelKey="uuid:e8cf1163-8234-4b35-865f-94a7322e40c3" >  
661 <name>  
662     StockQuotePortType  
663 </name>  
664 <categoryBag>  
665     <keyedReference  
666         tModelKey="uuid:fb5fb934-9a3d-39dc-9871-271f64780496"  
667         keyName="portType namespace"  
668         keyValue="http://example.com/stockquote/" />  
669     <keyedReference  
670         tModelKey="uuid:5b67c4b8-fbb8-3681-9c63-bf6b0c838dd0"  
671         keyName="WSDL type"  
672         keyValue="portType" />  
673 </categoryBag>  
674 <overviewDoc>  
675     <overviewURL>  
676         http://location/sample.wsdl  
677     </overviewURL>  
678 </overviewDoc>  
679 </tModel>
```

### 680 3.2.2 UDDI binding tModel

681 The WSDL binding entity maps to a tModel. The tModel name is the same as the WSDL  
682 binding local name. The tModel contains a categoryBag that specifies the WSDL namespace,  
683 it indicates that the tModel is of type "binding", it supplies a pointer to the portType tModel,  
684 and it indicates what protocols are supported by the binding. The wsdlSpec keyedReference  
685 ensures that users can find the tModel using the conventions defined in the Version 1 Best  
686 Practice. The overviewDoc provides a pointer to the WSDL file.

```
687 <tModel tModelKey="uuid:49662926-f4a5-4ba5-b8d0-32ab388dadda">  
688 <name>  
689     StockQuoteSoapBinding  
690 </name>  
691 <categoryBag>  
692     <keyedReference  
693         tModelKey="uuid:fb5fb934-9a3d-39dc-9871-271f64780496"  
694         keyName="binding namespace"  
695         keyValue="http://example.com/stockquote/" />  
696     <keyedReference  
697         tModelKey="uuid:5b67c4b8-fbb8-3681-9c63-bf6b0c838dd0"  
698         keyName="WSDL type"  
699         keyValue="binding" />  
700     <keyedReference  
701         tModelKey="uuid:d3e8ef29-877e-3486-b9e2-46af338d6c85"  
702         keyName="portType reference"  
703         keyValue="uuid:e8cf1163-8234-4b35-865f-94a7322e40c3" />  
704     <keyedReference  
705         tModelKey="uuid:ee733f78-b289-3637-8ff5-1623ea4672dd"  
706         keyName="SOAP protocol"  
707         keyValue="uuid:057916d3-6ec1-3755-b847-013f0f514586" />  
708     <keyedReference  
709         tModelKey="uuid:4eecd58-d3b0-3a6f-a466-9cce01cb1273"  
710         keyName="HTTP transport"  
711         keyValue="uuid:68DE9E80-AD09-469D-8A37-088422BFBC36" />  
712     <keyedReference  
713         tModelKey="uuid:clacf26d-9672-4404-9d70-39b756e62ab4"  
714         keyName="uddi-org:types"  
715         keyValue="wsdlSpec" />  
716 </categoryBag>  
717 <overviewDoc>  
718     <overviewURL>  
719         http://location/sample.wsdl
```

720  
721  
722

```
</overviewURL>  
</overviewDoc>  
</tModel>
```

### 723 3.2.3 UDDI businessService and bindingTemplate

724 The WSDL service entity maps to a businessService, and the WSDL port entity maps to a  
725 bindingTemplate. Any information from the WSDL binding extensibility elements is also  
726 captured in the bindingTemplate. The businessService name should be a human-readable  
727 name. The businessService contains a categoryBag that indicates that this service represents  
728 a WSDL service, and it specifies the WSDL namespace and WSDL service local name. The  
729 bindingTemplate specifies the endpoint of the service, and it contains a set of  
730 tModelInstanceDetails. The first tModelInstanceInfo indicates that the service implements the  
731 StockQuoteSoapBinding and provides the WSDL port local name. The second  
732 tModelInstanceInfo indicates that the service implements the StockQuotePortType.

733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778

```
<businessService  
  serviceKey="102b114a-52e0-4af4-a292-02700da543d4"  
  businessKey="1e65ea29-4e0f-4807-8098-d352d7b10368">  
  <name>Stock Quote Service</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:49662926-f4a5-4ba5-b8d0-32ab388dadda">  
          <description xml:lang="en">  
            The wsdl:binding that this wsdl:port implements.  
            The instanceParms specifies the port local name.  
          </description>  
          <instanceDetails>  
            <instanceParms>StockQuotePort</instanceParms>  
          </instanceDetails>  
        </tModelInstanceInfo>  
        <tModelInstanceInfo  
          tModelKey="uuid:e8cfl163-8234-4b35-865f-94a7322e40c3">  
          <description xml:lang="en">  
            The wsdl:portType that this wsdl:port implements.  
          </description>  
        </tModelInstanceInfo>  
      </tModelInstanceDetails>  
    </bindingTemplate>  
  </bindingTemplates>  
  <categoryBag>  
    <keyedReference  
      tModelKey=" uuid:5b67c4b8-fbb8-3681-9c63-bf6b0c838dd0"  
      keyName="WSDL type"  
      keyValue="service" />  
    <keyedReference  
      tModelKey="uuid:fb5fb934-9a3d-39dc-9871-271f64780496"  
      keyName="service namespace"  
      keyValue="http://example.com/stockquote/" />  
    <keyedReference  
      tModelKey=" uuid:451515ac-db54-3785-8937-114029f1d37b"  
      keyName="service local name"  
      keyValue="StockQuoteService" />  
  </categoryBag>  
</businessService>
```

### 779 3.3 Sample V2 Queries

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

781 **3.3.1 Find tModel for portType name**

782 Find the portType tModel for StockQuotePortType in the namespace  
783 <http://example.com/stockquote/>.

```
784 <find_tModel generic="2.0" xmlns="urn:uddi-org:api_v2">  
785 <name>StockQuotePortType</name>  
786 <categoryBag>  
787 <keyedReference  
788 tModelKey="uuid:5b67c4b8-fbb8-3681-9c63-bf6b0c838dd0"  
789 keyName="WSDL type"  
790 keyValue="portType" />  
791 <keyedReference  
792 tModelKey="uuid:fb5fb934-9a3d-39dc-9871-271f64780496"  
793 keyName="portType namespace"  
794 keyValue="http://example.com/stockquote/" />  
795 </categoryBag>  
796 </find_tModel>
```

797 This should return the tModelKey [uuid:5b67c4b8-fbb8-3681-9c63-bf6b0c838dd0](http://www.uuidgenerator.net/uuid/5b67c4b8-fbb8-3681-9c63-bf6b0c838dd0).

798 **3.3.2 Find bindings for portType**

799 Find all bindings for StockQuotePortType.

```
800 <find_tModel generic="2.0" xmlns="urn:uddi-org:api_v2">  
801 <categoryBag>  
802 <keyedReference  
803 tModelKey=" uuid:5b67c4b8-fbb8-3681-9c63-bf6b0c838dd0"  
804 keyName="WSDL type"  
805 keyValue="binding" />  
806 <keyedReference  
807 tModelKey="uuid:d3e8ef29-877e-3486-b9e2-46af338d6c85"  
808 keyName="portType reference"  
809 keyValue="uuid:e8cf1163-8234-4b35-865f-94a7322e40c3" />  
810 </categoryBag>  
811 </find_tModel>
```

812 This should return the tModelKey [uuid:d3e8ef29-877e-3486-b9e2-46af338d6c85](http://www.uuidgenerator.net/uuid/d3e8ef29-877e-3486-b9e2-46af338d6c85).

813 **3.3.3 Find Implementations of portType**

814 Find all implementations of StockQuotePortType.

```
815 <find_binding generic="2.0" xmlns="urn:uddi-org:api_v2">  
816 <tModelBag>  
817 <tModelKey>uuid:e8cf1163-8234-4b35-865f-94a7322e40c3</tModelKey>  
818 </tModelBag>  
819 </find_binding>
```

820 This should return the bindingKey [f793c521-0daf-434c-8700-0e32da232e74](http://www.uuidgenerator.net/uuid/e8cf1163-8234-4b35-865f-94a7322e40c3).

821 **3.3.4 Find implementations of binding**

822 Find all implementations of StockQuoteSoapBinding.

```
823 <find_binding generic="2.0" xmlns="urn:uddi-org:api_v2">  
824 <tModelBag>  
825 <tModelKey>uuid:49662926-f4a5-4ba5-b8d0-32ab388dadda</tModelKey>  
826 </tModelBag>  
827 </find_binding>
```

828 This should return the bindingKey [f793c521-0daf-434c-8700-0e32da232e74](http://www.uuidgenerator.net/uuid/49662926-f4a5-4ba5-b8d0-32ab388dadda).

829 **3.3.5 Find SOAP Implementations of portType**

830 Find all implementations of StockQuotePortType that support SOAP.

831 At least two queries are needed. The first query returns all the binding tModels that are  
832 categorized with SOAP. The second phase depends on whether or not other criteria are also  
833 required in the query.

834 **3.3.5.1 No Other Criteria**

835 In this case, only one other query is required. This query must specify “orAllKeys” and a  
836 tModelBag must be supplied which contains all the binding tModel keys returned by the first  
837 query.

838 **3.3.5.2 Other Criteria**

839 In this case, a query per binding tModel key is required and the default of “andAllKeys” must  
840 be used.

841 **3.3.6 Find SOAP/HTTP Implementations of portType**

842 This is similar to the previous case except that the first query must also include a category for  
843 the HTTP transport in addition to the SOAP protocol.

844 **3.3.7 Find the portType of a binding**

845 The portType of a binding is contained in the categoryBag of the binding tModel. No query is  
846 required once the tModel of the binding has been obtained. The keyValue of the  
847 keyedReference with tModelKey=”uuid:d3e8ef29-877e-3486-b9e2-46af338d6c85” contains  
848 the portType tModelKey.

---

849 **4 References**

850 **4.1 Normative**

- 851 **[RFC2119]** S. Bradner, *Key words for use in RFCs to Indicate Requirement*  
852 *Levels*, IETF RFC 2119, March 1997. Available at  
853 <http://www.ietf.org/rfc/rfc2119.txt>.
- 854 **[1]** Using WSDL in a UDDI Registry 1.08. Available at [http://www.oasis-](http://www.oasis-open.org/committees/uddi-spec/doc/bp/uddi-spec-tc-bp-using-wsdl-v108-20021110.pdf)  
855 [open.org/committees/uddi-spec/doc/bp/uddi-spec-tc-bp-using-wsdl-](http://www.oasis-open.org/committees/uddi-spec/doc/bp/uddi-spec-tc-bp-using-wsdl-v108-20021110.pdf)  
856 [v108-20021110.pdf](http://www.oasis-open.org/committees/uddi-spec/doc/bp/uddi-spec-tc-bp-using-wsdl-v108-20021110.pdf)
- 857 **[2]** Web Services Description Language (WSDL) 1.1, March 15, 2000.  
858 Available at <http://www.w3.org/TR/wsdl>
- 859 **[3]** UDDI Version 2.03 Data Structure Reference, July 7, 2002. Available  
860 at <http://uddi.org/pubs/DataStructure-V2.03-Published-20020719.pdf>.
- 861 **[4]** UDDI Version 3.0 Published Specification, 19 July 2002. Available at  
862 <http://www.uddi.org/pubs/uddi-v3.00-published-20020719.pdf>.
- 863 **[5]** XPointer xpointer() Scheme, W3C Working Draft, 10 July 2002.  
864 Available at <http://www.w3.org/TR/2002/WD-xptr-xpointer-20020710/>  
865



866

---

## A External WSDL Implementation Documents

867 There are multiple reasons why it may be desirable to support an external WSDL  
868 Implementation Document, among which are the following:

- 869 1. There are extensibility elements defined for the `wsdl:service`.
- 870 2. There is a `wsdl:documentation` element for a `wsdl:port`.
- 871 3. The address of a port may not be representable as a `uddi:accessPoint` value.
- 872 4. The authoritative source of the address is desired to be the WSDL document rather  
873 than UDDI.

874 The approach described here assumes that if any one of these reasons leads to the use of an  
875 external WSDL Deployment Document then the entire mapping described in this section is  
876 used.

877 There are two additional necessary pieces of information that must be captured to use  
878 external WSDL Implementation Documents:

- 879 1. The URL of the WSDL Implementation Document.
- 880 2. An indication that the port address must be obtained from the WSDL Implementation  
881 Document.

### 882 A.1 Capturing The URL

883 If an external WSDL Implementation Document is being used then the URL of this document  
884 must be used as the `accessPoint` value of each and every port of each and every service.

### 885 A.2 Obtaining the Port Address from WSDL

886 If a WSDL Implementation Document is being used then the `bindingTemplate` MUST contain  
887 sufficient information to identify the port address in the WSDL Implementation Document.  
888 The mapping described here MUST be used instead of the mapping defined in section 2.4.5.

889 In all cases where a WSDL Implementation Document is used, the `URLType` attribute of the  
890 `accessPoint` corresponding to each port MUST be "other", and the value of the `accessPoint`  
891 MUST be the URL of the WSDL Implementation Document.

892 The `bindingTemplate` MUST contain a `tModelInstanceInfo` element with a `tModelKey` of the  
893 WSDL Address `tModel`. This `tModelInstanceInfo` element, in combination with the protocol  
894 and transport information from the binding `tModel`, provides the necessary information to  
895 locate and interpret the endpoint address.

### 896 A.3 Querying Services that use a WSDL Implementation 897 Document

898 It is possible to query the services that have a WSDL Implementation Document by querying  
899 specifying the `tModelKey` of the WSDL Address `tModel`.

---

## 900 B Canonical tModels

901 This mapping introduces a number of canonical tModels that are used to represent WSDL  
902 metadata and relationships. These tModels are defined here.

### 903 B.1 WSDL Entity Type tModel

#### 904 B.1.1 Design Goals

905 This mapping uses a number of UDDI entities to represent the various entities within a WSDL  
906 file. A mechanism is required to indicate what type of WSDL entity is being described by each  
907 UDDI entity. The WSDL Entity Type tModel provides a typing system for this purpose. This  
908 taxonomy is used to indicate that a UDDI entity represents a particular type of WSDL entity.

#### 909 B.1.2 Definition

910 **Name:** uddi.org:wSDL:types  
911 **Description:** WSDL Type Category System  
912 **V3 format key:** uddi:uddi.org:wSDL:types  
913 **V1,V2 format key:** uuid:5b67c4b8-fbb8-3681-9c63-bf6b0c838dd0  
914 **Categorization:** categorization  
915 **Checked:** no

##### 916 B.1.2.1 V2 tModel Structure

```
917 <tModel tModelKey="uuid:5b67c4b8-fbb8-3681-9c63-bf6b0c838dd0" >  
918 <name>uddi.org:wSDL:types</name>  
919 <overviewDoc>  
920 <overviewURL>  
921 http://www.oasis-open.org/committees/uddi-spec/doc/tn/uddi-spec-tc-tn-  
922 wsdl-20030319-wd.htm#wSDLTypes  
923 </overviewURL>  
924 </overviewDoc>  
925 <categoryBag>  
926 <keyedReference  
927 tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4"  
928 keyValue="unchecked"  
929 />  
930 <keyedReference  
931 tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4"  
932 keyValue="categorization"  
933 />  
934 </categoryBag>  
935 </tModel>
```

#### 936 B.1.3 Valid Values

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

939

keyValue	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)

## 940 B.1.4 Example of Use

941 A V2 tModel representing a portType tModel would have a categoryBag representing its type:

```
942 <categoryBag>
943   <keyedReference
944     tModelKey="uuid:5b67c4b8-fbb8-3681-9c63-bf6b0c838dd0"
945     keyName="WSDL Entity type"
946     keyValue="portType"
947   />
948   ...
949 </categoryBag>
```

## 950 B.2 XML Namespace tModel

### 951 B.2.1 Design Goals

952 A namespace provides necessary qualifying information about a technical concept or model.  
 953 The XML Namespace tModel provides a mechanism to associate a namespace with a UDDI  
 954 entity. This taxonomy describes a UDDI entity by specifying the target namespace of the  
 955 description file (i.e., a WSDL file or XML Schema file) that describes the entity. *More than one*  
 956 *tModel might be categorized with the same namespace* – in fact, this mapping would be quite  
 957 common, as many WSDL files use a common target namespace for <wsdl:portType>,  
 958 <wsdl:binding>, and <wsdl:service> elements.

### 959 B.2.2 Definition

960 **Name:** uddi.org:xml:namespace  
 961 **Description:** A category system used to indicate namespaces  
 962 **V3 format key:** uddi:uddi.org:xml:namespace  
 963 **V1, V2 format key:** uuid:fb5fb934-9a3d-39dc-9871-271f64780496  
 964 **Categorization:** categorization  
 965 **Checked:** no

#### 969 B.2.2.1 V2 tModel Structure

```
970 <tModel tModelKey="uuid:fb5fb934-9a3d-39dc-9871-271f64780496">
971   <name>uddi.org:xml:namespace</name>
972   <overviewDoc>
973     <overviewURL>
974       http://www.oasis-open.org/committees/uddi-spec/doc/tn/uddi-spec-tc-
975       tn-wsdl-20030319-wd.htm #xmlNamespace
976     </overviewURL>
977   </overviewDoc>
978   <categoryBag>
979     <keyedReference
980       tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4"
981       keyValue="unchecked"
982     />
983     <keyedReference
984       tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4"
985       keyValue="categorization"
```

986  
987  
988

```
    />  
  </categoryBag>  
</tModel>
```

### 989 **B.2.3 Valid Values**

990 The values used in this taxonomy are namespaces of type "anyURI". The content of keyValue  
991 in a keyedReference that refers to this tModel is the target namespace of the WSDL file that  
992 describes the WSDL entity described by the UDDI entity.

### 993 **B.2.4 Example of Use**

994 A namespace keyedReference would be as follows:

995  
996  
997  
998  
999  
1000  
1001  
1002

```
<categoryBag>  
  <keyedReference  
    tModelKey="uuid:fb5fb934-9a3d-39dc-9871-271f64780496"  
    keyName="namespace"  
    keyValue="urn:foo"  
  />  
  ...  
</categoryBag>
```

## 1003 **B.3 XML Local Name tModel**

### 1004 **B.3.1 Design Goals**

1005 Each WSDL entity is identified by its name attribute, and this identification information needs  
1006 to be captured in the mapped UDDI entities. In the case of wsdl:portType and wsdl:binding,  
1007 the name attribute is mapped to the tModel name element. However, it isn't appropriate to  
1008 map the wsdl:service name attribute to the name element of the businessService, and, in the  
1009 case of wsdl:port, the bindingTemplate entity does not have a name element. The XML Local  
1010 Name tModel provides a mechanism to indicate the name attribute for these two constructs.

### 1011 **B.3.2 Definition**

1012 **Name:** uddi.org:xml:localName  
1013 **Description:** A category system used to indicate XML local names  
1014 **V3 format key:** uddi:uddi.org:xml:localName  
1015 **V1,V2 format key:** uuid:451515ac-db54-3785-8937-114029f1d37b  
1016 **Categorization:** categorization  
1017 **Checked:** no

1018  
1019  
1020

#### 1021 **B.3.2.1 V2 tModel Structure**

1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1030  
1031  
1032  
1033  
1034  
1035

```
<tModel tModelKey="uuid:451515ac-db54-3785-8937-114029f1d37b" >  
  <name>uddi.org:xml:localName</name>  
  <overviewDoc>  
    <overviewURL>  
      http://www.oasis-open.org/committees/uddi-spec/doc/tn/uddi-spec-tc-  
tn-wsdl-20030319-wd.htm#xmlLocalName  
    </overviewURL>  
  </overviewDoc>  
  <categoryBag>  
    <keyedReference  
      tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4"  
      keyValue="unchecked"  
    />  
  </keyedReference>
```

1036  
1037  
1038  
1039  
1040

```
        tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4"  
        keyValue="categorization"  
    />  
</categoryBag>  
</tModel>
```

### 1041 B.3.3 Valid Values

1042 The values used in this taxonomy are XML local names. The content of keyValue in a  
1043 keyedReference that refers to this tModel is equal to the name attribute of the WSDL entity  
1044 described by the UDDI entity.

### 1045 B.3.4 Example of Use

1046 A local name keyedReference would be as follows:

1047  
1048  
1049  
1050  
1051  
1052  
1053  
1054

```
<categoryBag>  
  <keyedReference  
    tModelKey="uuid:451515ac-db54-3785-8937-114029f1d37b"  
    keyName="Local service name"  
    keyValue="StockQuoteService"  
  />  
  ...  
</categoryBag>
```

## 1055 B.4 WSDL portType Reference tModel

### 1056 B.4.1 Design Goals

1057 WSDL entities exhibit many relationships. Specifically, a wsdl:port describes an  
1058 implementation of a wsdl:binding, and a wsdl:binding describes a binding of a particular  
1059 wsdl:portType. These same relationships must be expressed in the UDDI mapping. UDDI  
1060 provides a built-in mechanism, via the tModelInstanceInfo structure, to associate a  
1061 bindingTemplate with a tModel. But UDDI does not provide a built-in mechanism to describe a  
1062 relationship between two tModels. The WSDL portType Reference category system provides  
1063 a mechanism to indicate that a wsdl:binding tModel is a binding of a specific wsdl:portType  
1064 tModel.

### 1065 B.4.2 Definition

1066 **Name:** uddi.org:wsdl:portTypeReference  
1067 **Description:** A category system used to reference a wsdl:portType tModel  
1068 **V3 format key:** uddi:uddi.org:wsdl:portTypeReference  
1069 **V1,V2 format key:** uuid:d3e8ef29-877e-3486-b9e2-46af338d6c85  
1070 **Categorization:** categorization  
1071 **Checked:** no

#### 1072 B.4.2.1 V2 tModel Structure

1073  
1074  
1075  
1076  
1077  
1078  
1079  
1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087

```
<tModel tModelKey="uuid:d3e8ef29-877e-3486-b9e2-46af338d6c85" >  
  <name>uddi.org:wsdl:portTypeReference</name>  
  <description xml:lang="en">  
This tModel is a taxonomy tModel that can be used to identify a relationship  
to a portType tModel.  
</description>  
</overviewDoc>  
<overviewURL>  
  http://www.oasis-open.org/committees/uddi-spec/doc/tn/uddi-spec-tc-tn-  
wsdl-20030319-wd.htm#portTypeReference  
</overviewURL>  
</overviewDoc>  
<categoryBag>  
<keyedReference  
  tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4"
```

1088  
1089  
1090  
1091  
1092  
1093  
1094  
1095

```
        keyValue="categorization"  
    />  
    <keyedReference  
        tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4"  
        keyValue="unchecked"  
    />  
    </categoryBag>  
</tModel>
```

### 1096 B.4.3 Valid Values

1097 Valid values for this taxonomy are tModelKeys. The content of keyValue in a keyedReference  
1098 that refers to this tModel is the tModelKey of the wsdl:portType tModel being referenced.

### 1099 B.4.4 Example of Use

1100 One would add the following keyedReference to signify that a wsdl:binding implements a  
1101 specific portType:

1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109

```
<categoryBag>  
    <keyedReference  
        tModelKey="uuid:d3e8ef29-877e-3486-b9e2-46af338d6c85"  
        keyName="wsdl:portType Reference"  
        keyValue="uuid:e8cf1163-8234-4b35-865f-94a7322e40c3"  
    />  
    ...  
</categoryBag>
```

1110 Note that the keyValue is a tModelKey, which, if queried for using get\_tModelDetail, would  
1111 return the tModel that represents the portType.

## 1112 B.5 SOAP Protocol tModel

### 1113 B.5.1 Design Goals

1114 Web services can support a wide variety of protocols. Users looking for Web services may  
1115 want to search for Web services that support a specific protocol. The SOAP Protocol tModel  
1116 can be used to indicate that a Web service supports the SOAP 1.1 protocol. This tModel  
1117 correlates to the <http://schemas.xmlsoap.org/wsdl/soap/> namespace identified in the WSDL  
1118 Specification.

### 1119 B.5.2 Definition

1120 **Name:** uddi.org:protocol:soap  
1121 **Description:** A tModel that represents the SOAP 1.1 protocol  
1122 **V3 format key:** uddi:uddi.org:protocol:soap  
1123 **V1,V2 format key:** uuid:057916d3-6ec1-3755-b847-013f0f514586  
1124 **Categorization:** protocol

#### 1125 B.5.2.1 tModel Structure

1126  
1127  
1128  
1129  
1130  
1131  
1132  
1133  
1134  
1135  
1136  
1137  
1138  
1139

```
<tModel tModelKey="uuid:057916d3-6ec1-3755-b847-013f0f514586">  
    <name>uddi.org:protocol:soap</name>  
    <overviewDoc>  
        <overviewURL>  
            http://www.oasis-open.org/committees/uddi-spec/doc/tn/uddi-spec-tc-  
            tn-wsdl-20030319-wd.htm#soap  
        </overviewURL>  
    </overviewDoc>  
    <categoryBag>  
        <keyedReference  
            tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4"  
            keyValue="protocol"  
        />  
    </categoryBag>
```

1140

&lt;/tModel&gt;

## 1141 B.5.3 Example of Use

1142 The SOAP Protocol tModel is used to categorise a binding tModel that corresponds to a  
1143 wsdl:binding that supports the SOAP 1.1 protocol.

```

1144 <tModel tModelKey="uuid:49662926-f4a5-4ba5-b8d0-32ab388dadda">
1145 <name>...</name>
1146 <categoryBag>
1147   <keyedReference
1148     tModelKey="uuid:fb5fb934-9a3d-39dc-9871-271f64780496"
1149     keyName="binding namespace"
1150     keyValue="http://example.com/stockquote/" />
1151   <keyedReference
1152     tModelKey="uuid:5b67c4b8-fbb8-3681-9c63-bf6b0c838dd0"
1153     keyName="WSDL type"
1154     keyValue="binding" />
1155   <keyedReference
1156     tModelKey="uuid:d3e8ef29-877e-3486-b9e2-46af338d6c85"
1157     keyName="portType reference"
1158     keyValue="uuid:e8cf1163-8234-4b35-865f-94a7322e40c3" />
1159   <keyedReference
1160     tModelKey="uuid:ee733f78-b289-3637-8ff5-1623ea4672dd"
1161     keyName="SOAP protocol"
1162     keyValue="uuid:057916d3-6ec1-3755-b847-013f0f514586" />
1163   <keyedReference
1164     tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4"
1165     keyName="types"
1166     keyValue="wsdlSpec" />
1167 </categoryBag>
1168 <overviewDoc>
1169   <overviewURL>http://location/sample.wsdl</overviewURL>
1170 </overviewDoc>
1171 </tModel>

```

## 1172 B.6 HTTP Protocol tModel

### 1173 B.6.1 Design Goals

1174 Web services can support a wide variety of protocols. Users looking for Web services may  
1175 want to search for Web services that support a specific protocol. The HTTP Protocol tModel  
1176 can be used to indicate that a Web service supports the HTTP protocol. Note that this tModel  
1177 is different from the HTTP Transport tModel. This tModel represents a protocol; for example, it  
1178 represents the http://schemas.xmlsoap.org/wsdl/http/ namespace in the WSDL specification.  
1179 The HTTP Transport tModel represents a transport.

### 1180 B.6.2 Definition

1181 **Name:** uddi.org:protocol:http  
1182 **Description:** A tModel that represents the HTTP protocol  
1183 **V3 format key:** uddi:uddi.org:protocol:http  
1184 **V1,V2 format key:** uuid:e01a4d7f-b7d6-337d-b47c-2cf3e84edd75  
1185 **Categorization:** protocol

#### 1186 B.6.2.1 V2 tModel Structure

```

1187 <tModel tModelKey="uuid:e01a4d7f-b7d6-337d-b47c-2cf3e84edd75">
1188   <name>uddi.org:protocol:http</name>
1189   <overviewDoc>
1190     <overviewURL>
1191       http://www.oasis-open.org/committees/uddi-spec/doc/tn/uddi-
1192       spec-tc-tn-wsdl-20030319-wd.htm#http
1193     </overviewURL>
1194   </overviewDoc>
1195   <categoryBag>

```

1196  
1197  
1198  
1199  
1200  
1201

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

1202

### B.6.3 Example of Use

1203  
1204

The HTTP Protocol tModel is used to categorise a binding tModel that corresponds to a wsdl:binding that supports the HTTP protocol.

1205  
1206  
1207  
1208  
1209  
1210  
1211  
1212  
1213  
1214  
1215  
1216  
1217  
1218  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236

```
<tModel tModelKey="uuid:49662926-f4a5-4ba5-b8d0-32ab388dadda">
<name>
  StockQuoteSoapBinding
</name>
<categoryBag>
  <keyedReference
    tModelKey="uuid:fb5fb934-9a3d-39dc-9871-271f64780496"
    keyName="binding namespace"
    keyValue="http://example.com/stockquote/" />
  <keyedReference
    tModelKey="uuid:5b67c4b8-fbb8-3681-9c63-bf6b0c838dd0"
    keyName="WSDL type"
    keyValue="binding" />
  <keyedReference
    tModelKey="uuid:d3e8ef29-877e-3486-b9e2-46af338d6c85"
    keyName="portType reference"
    keyValue="uuid:e8cf1163-8234-4b35-865f-94a7322e40c3" />
  <keyedReference
    tModelKey="uuid:ee733f78-b289-3637-8ff5-1623ea4672dd"
    keyName="HTTP protocol"
    keyValue="uuid:e01a4d7f-b7d6-337d-b47c-2cf3e84edd75" />
  <keyedReference
    tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4"
    keyName="types"
    keyValue="wsdlSpec" />
</categoryBag>
<overviewDoc>
  <overviewURL>
    http://location/sample.wsdl
  </overviewURL>
</overviewDoc>
</tModel>
```

1237

## B.7 Protocol Categorization

1238

### B.7.1 Design Goals

1239  
1240  
1241  
1242

A Web service may communicate using a variety of protocols. A WSDL binding binds a portType to a specific protocol. A user may wish to search for bindings that implement a specific protocol. The Protocol Categorization tModel provides a mechanism to capture this protocol information in the UDDI binding tModel.

1243

### B.7.2 Definition

1244  
1245  
1246  
1247  
1248  
1249  
1250  
1251

**Name:** uddi-org:wsdl:categorization:protocol  
**Description:** Category system used to describe the protocol supported by a wsdl:binding.  
**V3 format key:** uddi:uddi.org:wsdl:categorization:protocol  
**V1,V2 format key:** uuid:ee733f78-b289-3637-8ff5-1623ea4672dd  
**Categorization:** categorization  
**Checked:** no



1252

### B.7.2.1 V2 tModel Structure

1253  
1254  
1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1270  
1271

```
<tModel tModelKey="uuid:uddi.org:wSDL:categorization:protocol">
  <name>uddi-org:wSDL:categorization:protocol</name>
  <overviewDoc>
    <overviewURL>
      http://www.oasis-open.org/committees/uddi-spec/doc/tn/uddi-spec-
tc-tn-wSDL-20030319-wd.htm#protocol
    </overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference keyName="types"
      keyValue="categorization"
      tModelKey=" uuid:clacf26d-9672-4404-9d70-
39b756e62ab4" />
    <keyedReference keyName="types"
      keyValue="unchecked"
      tModelKey=" uuid:clacf26d-9672-4404-9d70-
39b756e62ab4" />
  </categoryBag>
</tModel>
```

1272

### B.7.3 Example of Use

1273

The Protocol category scheme is used to indicate the protocol that a binding supports.

1274  
1275  
1276  
1277  
1278  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1290  
1291  
1292  
1293  
1294  
1295  
1296  
1297  
1298  
1299  
1300  
1301

```
<tModel tModelKey="uuid:49662926-f4a5-4ba5-b8d0-32ab388dadda">
  <name>StockQuoteSoapBinding</name>
  <categoryBag>
    <keyedReference
      tModelKey="uuid:fb5fb934-9a3d-39dc-9871-271f64780496"
      keyName="binding namespace"
      keyValue="http://example.com/stockquote/" />
    <keyedReference
      tModelKey="uuid:5b67c4b8-fbb8-3681-9c63-bf6b0c838dd0"
      keyName="WSDL type"
      keyValue="binding" />
    <keyedReference
      tModelKey="uuid:d3e8ef29-877e-3486-b9e2-46af338d6c85"
      keyName="portType reference"
      keyValue="uuid:e8cf1163-8234-4b35-865f-94a7322e40c3" />
    <keyedReference
      tModelKey="uuid:clacf26d-9672-4404-9d70-39b756e62ab4"
      keyName="types"
      keyValue="wSDLSpec" />
    <keyedReference
      tModelKey="uuid:ee733f78-b289-3637-8ff5-1623ea4672dd"
      keyName="WSDL binding supports the SOAP protocol"
      keyValue="uddi:057916d3-6ec1-3755-b847-013f0f514586" />
  </categoryBag>
  <overviewDoc>
    <overviewURL>http://location/sample.wSDL</overviewURL>
  </overviewDoc>
</tModel>
```

1302

## B.8 Transport Categorization

1303

### B.8.1 Design Goals

1304  
1305  
1306  
1307

A Web service may communicate using a variety of transports. A WSDL binding binds a portType to a specific transport protocol. A user may wish to search for bindings that implement a specific transport protocol. The Transport Categorization tModel provides a mechanism to capture this transport information in the UDDI binding tModel.

1308

### B.8.2 Definition

1309  
1310  
1311

**Name:** uddi-org:wSDL:categorization:transport  
**Description:** Category system used to describe the transport supported by a wSDL:binding.

1312 **V3 format key:** uddi:uddi.org:wSDL:categorization:transport  
1313  
1314 **V1,V2 format key:** uuid:4eecd58-d3b0-3a6f-a466-9cce01cb1273  
1315 **Categorization:** categorization  
1316 **Checked:** no

### 1317 **B.8.2.1 V2 tModel Structure**

```
1318 <tModel tModelKey="uuid:uddi.org:wSDL:categorization:transport">  
1319 <name>uddi-org:wSDL:categorization:transport</name>  
1320 <overviewDoc>  
1321 <overviewURL>  
1322 http://www.oasis-open.org/committees/uddi-spec/doc/tn/uddi-spec-  
1323 tc-tn-wSDL-20030319-wd.htm#transport  
1324 </overviewURL>  
1325 </overviewDoc>  
1326 <categoryBag>  
1327 <keyedReference keyName="types"  
1328 keyValue="categorization"  
1329 tModelKey="uuid:c1acf26d-9672-4404-9d70-  
1330 39b756e62ab4"/>  
1331 <keyedReference keyName="types"  
1332 keyValue="unchecked"  
1333 tModelKey="uuid:c1acf26d-9672-4404-9d70-  
1334 39b756e62ab4"/>  
1335 </categoryBag>  
1336 </tModel>
```

### 1337 **B.8.3 Example of Use**

1338 The Transport category scheme is used to indicate the transport that a binding supports.

```
1339 <tModel tModelKey="uuid:49662926-f4a5-4ba5-b8d0-32ab388dadda">  
1340 <name>StockQuoteSoapBinding</name>  
1341 <categoryBag>  
1342 <keyedReference  
1343 tModelKey="uuid:fb5fb934-9a3d-39dc-9871-271f64780496"  
1344 keyName="binding namespace"  
1345 keyValue="http://example.com/stockquote/" />  
1346 <keyedReference  
1347 tModelKey="uuid:5b67c4b8-fbb8-3681-9c63-bf6b0c838dd0"  
1348 keyName="WSDL type"  
1349 keyValue="binding" />  
1350 <keyedReference  
1351 tModelKey="uuid:d3e8ef29-877e-3486-b9e2-46af338d6c85"  
1352 keyName="portType reference"  
1353 keyValue="uuid:e8cf1163-8234-4b35-865f-94a7322e40c3" />  
1354 <keyedReference  
1355 tModelKey="uuid:c1acf26d-9672-4404-9d70-39b756e62ab4"  
1356 keyName="types"  
1357 keyValue="wSDLSpec" />  
1358 <keyedReference  
1359 tModelKey="uuid:hashed key"  
1360 keyName="WSDL binding protocol"  
1361 keyValue="uddi:057916d3-6ecl-3755-b847-013f0f514586" />  
1362 <keyedReference  
1363 tModelKey="uuid:4eecd58-d3b0-3a6f-a466-9cce01cb1273"  
1364 keyName="WSDL transport protocol"  
1365 keyValue="uuid:68DE9E80-AD09-469D-8A37-088422BFBC36" />  
1366 </categoryBag>  
1367 <overviewDoc>  
1368 <overviewURL>http://location/sample.wSDL</overviewURL>  
1369 </overviewDoc>  
1370 </tModel>  
1371
```

## 1372 B.9 WSDL Address tModel

### 1373 B.9.1 Design Goals

1374 A service provider may not want to specify the address of a service port in the  
1375 uddi:accessPoint element and instead require the user to retrieve a WSDL file to obtain the  
1376 service address. UDDI V2 does not provide a built-in mechanism to indicate that the endpoint  
1377 address should be obtained from a WSDL file. This document describes an approach to  
1378 provide a mechanism using existing UDDI V2 features. This approach requires that the  
1379 bindingTemplate indicate that the WSDL file must be retrieved to obtain the address  
1380 information. The WSDL Address tModel provides such a mechanism. A V2 bindingTemplate  
1381 includes a tModelInstanceInfo element that references this tModel to indicate that the address  
1382 information must be retrieved from the WSDL file.

### 1383 B.9.2 Definition

1384 **Name:** uddi-org:wSDL:address  
1385 **Description:** A tModel used to indicate the WSDL address option  
1386 **V3 format key:** uddi:uddi.org:wSDL:address  
1387 **V1,V2 format key:** uuid:2646df99-ec31-3c67-80e2-5743d0c0e829  
1388 **Categorization:** none

#### 1389 B.9.2.1 V2 tModel Structure

```
1390 <tModel tModelKey="uuid: " >  
1391   <name>uddi-org:wSDL:address</name>  
1392   <description xml:lang="en">  
1393     This tModel is used to specify the URL fact that the address must be obtained  
1394     from the WSDL deployment file.  
1395   </description>  
1396   <overviewDoc>  
1397     <overviewURL>  
1398       http://www.oasis-open.org/committees/uddi-spec/doc/tn/uddi-spec-tc-  
1399       tn-wSDL-20030319-wd.htm#Address  
1400     </overviewURL>  
1401   </overviewDoc>  
1402 </tModel>
```

### 1403 B.9.3 Valid Values

1404 There are no valid values associated with this tModel, it is simply a marker.

### 1405 B.9.4 Example of Use

1406 If a service provider requires the user to retrieve the service endpoint from a WSDL file rather  
1407 than from the UDDI bindingTemplate, the accessPoint element must have a value of "WSDL"  
1408 and a URLType attribute value of "other":

```
1409 <bindingTemplate  
1410   bindingKey="f793c521-0daf-434c-8700-0e32da232e74"  
1411   serviceKey="102b114a-52e0-4af4-a292-02700da543d4">  
1412   <accessPoint URLType="other">WSDL</accessPoint>  
1413 <tModelInstanceDetails>  
1414   <tModelInstanceInfo  
1415     tModelKey="uuid:2646df99-ec31-3c67-80e2-5743d0c0e829">  
1416     <tModelInstanceInfo>  
1417     ...  
1418   </tModelInstanceInfo>  
1419 </tModelInstanceDetails>  
</bindingTemplate
```

1420

## C Using XPointer in overviewURL

1421

### C.1 XPointer Syntax

1422

In this mapping of WSDL to UDDI, a UDDI entity describes a particular element within a WSDL file. The particular WSDL element described SHOULD be determined by using the metadata contained within the entity's categoryBag, and either the UDDI entity's name or the instanceParms value specified in the tModelInstanceInfo that relates to the binding that a port implements. Alternatively, the overviewURL value MAY contain a fragment identifier that identifies the particular WSDL element.

1428

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.

1429

1430

If the optional fragment identifier is used, the syntax defined by XPointer [5] MUST be used for the fragment identifier. It should be noted that at the time of writing this Technical Note, XPointer is a set of Working Draft documents and is therefore subject to change.

1431

1432

1433

#### C.1.1 Example of Use

1434

Referring to the WSDL Sample in Section 3.1, the StockQuotePortType tModel may reference the wsdl:portType element directly from the overviewURL using XPointer syntax.

1435

1436

1437

1438

1439

1440

1441

1442

1443

1444

1445

1446

1447

1448

1449

1450

1451

1452

1453

1454

1455

1456

1457

1458

```
<tModel tModelKey="uuid:e8cf1163-8234-4b35-865f-94a7322e40c3" >
  <name>
    StockQuotePortType
  </name>
  <categoryBag>
    <keyedReference
      tModelKey="uuid:fb5fb934-9a3d-39dc-9871-271f64780496"
      keyName="portType target namespace"
      keyValue="http://example.com/stockquote/"
    />
    <keyedReference
      tModelKey="uuid:5b67c4b8-fbb8-3681-9c63-bf6b0c838dd0"
      keyName="WSDL Entity Type"
      keyValue="portType"
    />
  </categoryBag>
  <overviewDoc>
    <overviewURL>
      http://location/sample.wsdl#xmlns(wsdl=http://schemas.xmlsoap.org/wsdl/)
      xpointer(/wsdl:definitions/wsdl:portType[@name="StockQuotePortType"]).
    </overviewURL>
  </overviewDoc>
</tModel>
```

1459

---

1460 **D Acknowledgments**

1461 The following individuals were members of the committee during the development of this  
1462 technical note:

1463           Andrew Hately, IBM  
1464           Sam Lee, Oracle  
1465           Alok Srivastava, Oracle  
1466           Claus von Riegen, SAP

1467

---

## E Revision History

1468

Rev	Date	By Whom	What
20021022	22 Oct 2002	John Colgrave and Karsten Januszewski	First draft of V2.0 TN
20021114	14 Nov 2002	Tony Rogers and Anne Thomas Manes	Second draft of V2.0 TN for TC discussion
20030319	19 Mar 2003	John Colgrave, Anne Thomas Manes and Tony Rogers	Final draft of V2.0 TN for TC review

1469

1470

---

## F Notices

1471 OASIS takes no position regarding the validity or scope of any intellectual property or other  
1472 rights that might be claimed to pertain to the implementation or use of the technology  
1473 described in this document or the extent to which any license under such rights might or might  
1474 not be available; neither does it represent that it has made any effort to identify any such  
1475 rights. Information on OASIS's procedures with respect to rights in OASIS specifications can  
1476 be found at the OASIS website. Copies of claims of rights made available for publication and  
1477 any assurances of licenses to be made available, or the result of an attempt made to obtain a  
1478 general license or permission for the use of such proprietary rights by implementors or users  
1479 of this specification, can be obtained from the OASIS Executive Director.

1480 OASIS invites any interested party to bring to its attention any copyrights, patents or patent  
1481 applications, or other proprietary rights which may cover technology that may be required to  
1482 implement this specification. Please address the information to the OASIS Executive Director.

1483 **Copyright © OASIS Open 2003. All Rights Reserved.**

1484 This document and translations of it may be copied and furnished to others, and derivative  
1485 works that comment on or otherwise explain it or assist in its implementation may be  
1486 prepared, copied, published and distributed, in whole or in part, without restriction of any kind,  
1487 provided that the above copyright notice and this paragraph are included on all such copies  
1488 and derivative works. However, this document itself does not be modified in any way, such as  
1489 by removing the copyright notice or references to OASIS, except as needed for the purpose  
1490 of developing OASIS specifications, in which case the procedures for copyrights defined in  
1491 the OASIS Intellectual Property Rights document must be followed, or as required to translate  
1492 it into languages other than English.

1493 The limited permissions granted above are perpetual and will not be revoked by OASIS or its  
1494 successors or assigns.

1495 This document and the information contained herein is provided on an "AS IS" basis and  
1496 OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT  
1497 LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL  
1498 NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY  
1499 OR FITNESS FOR A PARTICULAR PURPOSE.