



Web Services Topics 1.2 (WS-Topics)

Working Draft 01, 22 July 2004

Document identifier:

wsn-WS-Topics-1.2-draft-01

Location:

<http://docs.oasis-open.org/wsn/2004/06/wsn-WS-Topics-1.2-draft-01.pdf>

Editor:

William Vambenepe, HP <vbp@hp.com>

Abstract:

The Event-driven, or Notification-based, interaction pattern is a commonly used pattern for inter-object communications. Examples exist in many domains, for example in publish/subscribe systems provided by Message Oriented Middleware vendors, or in system and device management domains. This notification pattern is increasingly being used in a Web services context.

WS-Notification is a family of related white papers and specifications that define a standard Web services approach to notification using a topic-based publish/subscribe pattern. It includes: standard message exchanges to be implemented by service providers that wish to participate in Notifications, standard message exchanges for a notification broker service provider (allowing publication of messages from entities that are not themselves service providers), operational requirements expected of service providers and requestors that participate in notifications, and an XML model that describes topics. The WS-Notification family of documents includes: a white paper: Publish-Subscribe Notification for Web services as well as three normative specifications: [WS-BaseNotification], [WS-BrokeredNotification], and WS-Topics.

This document defines a mechanism to organize and categorize items of interest for subscription known as "topics". These are used in conjunction with the notification mechanisms defined in WS-Base Notification. WS-Topics defines three topic expression dialects that can be used as subscription expressions in subscribe request messages and other parts of the WS-Notification system. It further specifies an XML model for

33 describing metadata associated with topics. This specification should be read in
34 conjunction with the WS-Base Notification specification and the Publish-Subscribe
35 Notification for Web Services document.

36 **Status:**

37 On July 2nd 2004, this document was approved by the OASIS WS-Notification Technical
38 Committee for publication so that users of the specification have a stable draft version
39 available until the TC publishes a Committee Draft. Send comments to the editor.

40 Committee members should send comments on this specification to the [wsn@lists.oasis-](mailto:wsn@lists.oasis-open.org)
41 [open.org](mailto:wsn@lists.oasis-open.org) list. Others should subscribe to and send comments to the [comment@lists.oasis-open.org](mailto:wsn-
42 <a href=) list. To subscribe, send an email message to [comment-request@lists.oasis-open.org](mailto:wsn-
43 <a href=) with the word "subscribe" as the body of the
44 message.

45 For information on whether any patents have been disclosed that may be essential to
46 implementing this specification, and any offers of patent licensing terms, please refer to
47 the Intellectual Property Rights section of the WSN TC web page ([http://www.oasis-](http://www.oasis-open.org/committees/wsn/)
48 [open.org/committees/wsn/](http://www.oasis-open.org/committees/wsn/)).

49 **Table of Contents**

50	1	Introduction	4
51	1.1	Goals and Requirements	4
52	1.1.1	Requirements	4
53	1.1.2	Non-Goals	4
54	1.2	Notational Conventions	5
55	1.3	Namespaces	5
56	2	Terminology and Concepts	6
57	3	Topics and Topics Spaces	7
58	4	Example	9
59	5	Modeling Topic Spaces in XML	10
60	6	Modeling Topics in XML	10
61	7	Topic Expressions	13
62	7.1	SimpleTopic Expressions	13
63	7.2	ConcreteTopicPath Expressions	14
64	7.3	FullTopicPath Expressions	15
65	7.3.1	Validating FullTopicExpressions	16
66	8	AliasRefs and their resolution	18
67	9	Growing a Topic Tree	19
68	10	The “ad-hoc” Topic Space	20
69	11	NotificationProducer and Topics	21
70	12	Security Considerations	22
71	13	References	23
72		Appendix A. Acknowledgments	24
73		Appendix B. XML Schema	25
74		Appendix C. Revision History	30
75		Appendix D. Notices	31
76			

77 1 Introduction

78 The Event-driven, or Notification-based, interaction pattern is a commonly used pattern for inter-
79 object communications. Examples exist in many domains, for example in publish/subscribe
80 systems provided by Message Oriented Middleware vendors, or in system and device
81 management domains.

82 This document defines a mechanism to organize and categorize items of interest for subscription
83 known as “topics”. These are used in conjunction with the notification mechanisms defined in WS-
84 Base Notification.

85 WS-Topics defines three topic expression dialects that can be used as subscription expressions
86 in subscribe request messages and other parts of the WS-Notification system. It further specifies
87 an XML model for describing metadata associated with topics. This specification should be read
88 in conjunction with the WS-Base Notification specification and the Publish-Subscribe Notification
89 for Web Services white paper.

90 1.1 Goals and Requirements

91 The goal of the WS-Topics specification is to define a mechanism to organize and categorize
92 items of interest for subscription known as “topics”. It defines a set of topic expression dialects
93 that can be used as subscription expressions in subscribe request messages and other parts of
94 the WS-Notification system.

95 1.1.1 Requirements

96 In meeting this goal, the specification must address the following specific requirements:

- 97 § Must support resource-constrained devices. The specifications must be factored in a way
98 that allows resource-constrained devices to participate in the Notification pattern. Such
99 devices will be able to send information to, and receive information from Web services,
100 without having to implement all the features of the specifications.
- 101 § Must permit transformation and aggregation of Topics: It must be possible to construct
102 configurations (using intermediary brokers) where the Topic subscribed to by the
103 NotificationConsumer differs from the Topic published to by the NotificationProducer, yet
104 NotificationMessages from the NotificationProducer are routed to the
105 NotificationConsumer by a broker that is acting according to administratively-defined
106 rules.
- 107 § Must permit non-centralized development of a topic tree: It must be possible for actors to
108 define additional topics based on existing topics without requiring coordination with the
109 actor responsible for creating the topics that are being built on.

110 1.1.2 Non-Goals

111 The following aspects are outside the scope of these specifications:

- 112 § Defining the format of notification payloads: The data carried in notification messages is
113 application-domain specific, and this specification does not prescribe any particular
114 format for this data.

115 **1.2 Notational Conventions**

116 The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD",
117 "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be
118 interpreted as described in [RFC 2119].

119 When describing abstract data models, this specification uses the notational convention used by
120 the [XML Infoset]. Specifically, abstract property names always appear in square brackets (e.g.,
121 [some property]).

122 When describing concrete XML schemas, this specification uses the notational convention of
123 [WS-Security]. Specifically, each member of an element's [children] or [attributes] property is
124 described using an XPath-like notation (e.g., /x:MyHeader/x:SomeProperty/@value1). The use of
125 {any} indicates the presence of an element wildcard (<xsd:any/>). The use of @{any} indicates
126 the presence of an attribute wildcard (<xsd:anyAttribute/>).

127 **1.3 Namespaces**

128 The following namespaces are used in this document:

Prefix	Namespace
xsd	http://www.w3.org/2001/XMLSchema
wsrp	http://docs.oasis-open.org/wsrp/2004/06/wsrp-WS-ResourceProperties-1.2-draft-01.xsd
wsbf	http://docs.oasis-open.org/wsrp/2004/06/wsrp-WS-BaseFaults-1.2-draft-01.xsd
wsbfw	http://docs.oasis-open.org/wsrp/2004/06/wsrp-WS-BaseFaults-1.2-draft-01.wsdl
wsnt	http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BaseNotification-1.2-draft-01.xsd
wstop	http://docs.oasis-open.org/wsn/2004/06/wsn-WS-Topics-1.2-draft-01.xsd

129

130 **2 Terminology and Concepts**

131 Please refer to [WS-Notification WP] for a list of terms and their definitions.

3 Topics and Topics Spaces

132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174

A collection of related Topics is used to organize and categorize a set of notification messages. It provides a convenient means by which subscribers can reason about notifications of interest. Topics appear in several places within the WS-Notification system. As part of the publication of a NotificationMessage, the Publisher associates it with one or more Topics. When a Subscriber creates a Subscription, it associates the Subscription with one or more Topics. The NotificationProducer uses these Topic lists as part of the matching process: a NotificationMessage is delivered to a NotificationConsumer if the list of Topics associated with the Subscription has a non-empty intersection with the list of Topics associated with the NotificationMessage.

In order to avoid naming collisions, and to facilitate interoperation between independently developed NotificationProducers and Subscribers, every WS-Notification Topic is assigned to an XML Namespace. The set of Topics associated with a given XML Namespace is termed a *Topic Space*. Any XML namespace has the potential to scope a single collection of Topics. Of course, not every XML namespace will define a Topic Space.

It is important to understand the distinction between a Topic Space and the set of Topics (the "Topic Set") supported by a NotificationProducer. A Topic Space is just an abstract set of Topic definitions. While it is certainly possible for a given Topic Space to be used by exactly one Notification Producer, there is no expectation that this will be the case. Topics from a single Topic Space may be referenced in the Topic Sets of many different NotificationProducers. Moreover the Topic Set of a NotificationProducer MAY contain Topics from several different Topic Spaces. This concept is expanded upon in section 11.

Each Topic can have zero or more *child topics* and a child topic can itself contain further child topics. A Topic without a *parent* is termed a *root topic*. A particular root topic and all its descendents form a hierarchy (termed a *Topic Tree*).

The rationale for hierarchical topic structures is:

- § They allow Subscribers to subscribe against multiple Topics. For example a Subscriber can subscribe against an entire Topic Tree, or a subset of the Topics in a Topic Tree. This reduces the number of subscription requests that a Subscriber needs to issue if it is interested in a large sub-tree. It also means that a Subscriber can receive NotificationMessages related to descendent topics without having to be specifically aware of their existence.
- § They provide a convenient way to manage large Topic Spaces (for example when administering security policies).

Note: Although WS-Notification permits hierarchical topic structures, there is no requirement or expectation that all Topic Spaces will contain them. It is perfectly possible for a Topic Space to contain only root topics (possibly only a single root topic). A NotificationProducer is not required to support structured topics. It may restrict its Topic Set to include only topics from Topic Spaces that contain only root Topics; even if it does include topics from a Topic Space that contains topic hierarchies, it may choose only to support root topics from that Topic Space.

A Topic Space is thus a collection (forest) of Topic Trees. The Topic Space contains additional metadata relating to its member Topics. The metadata describing a particular Topic Space can be modeled as an XML document (see section 5).

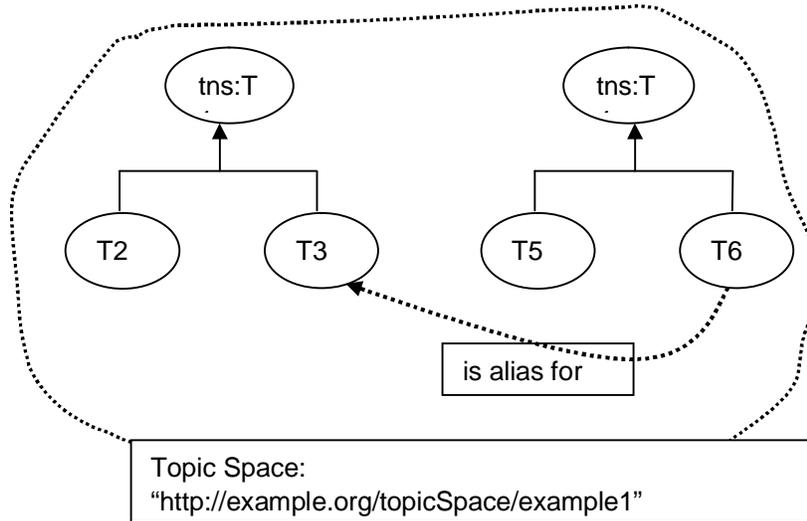
175 Each Topic has a local name, an NCName. All root topics must have unique names within their
176 Topic Space. In this way, a root Topic can be uniquely referenced by a QName formed by
177 combining the XML Namespace associated with the Topic Space and the local name of the root
178 topic. Child topics can only be referred to relative to their ancestor root topic's QName using a
179 path-based TopicExpression dialect (see section 7).

180 No Topic can contain two immediate child topics with the same name, however Topics with the
181 same name can appear elsewhere in a Topic Tree, and no relationship is implied. Similarly two
182 separate Topic Trees in the same Topic Space may contain descendent Topics with the same
183 name; these are not necessarily related to each other in any way either.

184
185
186
187
188
189

4 Example

Consider a Topic Space that can be depicted as follows. The Topic Space is contained in the "http://example.org/topicSpace/example1" namespace. This Topic Space has two root Topics, named t1 and t4. Topic t1 has two child topics, t2 and t3. Topic t4 has two child topics, t5 and t6. Topic t6 is an alias for t1's child topic t3.



190

191 This topic space and its metadata can be described using the following XML instance document:

```
192 <?xml version="1.0" encoding="UTF-8"?>
193 <wstop:topicSpace name="TopicSpaceExample1"
194   targetNamespace="http://example.org/topicSpace/example1"
195   xmlns:tns="http://example.org/topicSpace/example1"
196   xmlns:xyz="http://example.org/anotherNamespace"
197   xmlns:wstop=
198     "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-Topics-1.2-draft-01.xsd" >
199   <wstop:topic name="t1">
200     <wstop:topic name="t2" messageTypes="xyz:m1 tns:m2"/>
201     <wstop:topic name="t3" messageTypes="xyz:m3"/>
202   </wstop:topic>
203   <wstop:topic name="t4">
204     <wstop:topic name="t5" messageTypes="tns:m3"/>
205     <wstop:topic name="t6">
206       <wstop:AliasRef
207         dialect="http://docs.oasis-open.org/wsn/2004/06/TopicExpression/Concrete" >
208         tns:t1/t3
209       </wstop:AliasRef>
210     </wstop:topic>
211   </wstop:topic>
212 </wstop:topicSpace>
```

213 We describe the details behind modeling topic spaces and topics in the following sections.

214 **5 Modeling Topic Spaces in XML**

215 The WS-Topics XML Schema contains element and type definitions used to create topic space
216 instance documents. An instance document is associated with a single Topic Space and contains
217 the names of Topics in that Topic Space along with their metadata. It may include all the topics in
218 that topic space, or just a subset of them. The following is a non-normative description of a
219 TopicSpace element:

```
220 ...  
221 <TopicSpace name=NCName? targetNamespace=anyURI ...>  
222   <Topic ... />*  
223   ...  
224 </TopicSpace>
```

225 A TopicSpace element is further constrained in the following way:

226 /wstop:TopicSpace

227 The top-level element in a topic space instance document. It contains Topic declaration
228 elements and associates them with the XML Namespace for the topic space

229 /wstop:TopicSpace/@name

230 An optional name that can be assigned to the TopicSpace element for light-weight
231 documentation purposes.

232 /wstop:TopicSpace/@targetNameSpace

233 The XML Namespace for this topic space. It is expressed as a URI. This forms the
234 namespace component of the QName of each root Topic in the Topic Space.

235 /wstop:topicSpace/Topic

236 The TopicSpace has a collection of zero or more child Topic elements that define the roots of
237 the Topic Trees within the Topic Space. The TopicSpace element may contain any number of
238 Topic elements. The value of /Topic/@name MUST be unique amongst all root Topics
239 defined in the TopicSpace.

240 /wstop:TopicSpace/{ any }

241 This is an extensibility mechanism to allow additional elements to be specified.

242 /wstop:TopicSpace/@{ any }

243 This is an extensibility mechanism to allow additional attributes to be specified.

244 **6 Modeling Topics in XML**

245 WS-Notification defines an XML representation of a Topic that can be represented in the following
246 non-normative fashion:

```
247 <TopicSpace name=... targetNamespace=...>  
248   <Topic name=NCName messageTypes=list of QName? final=boolean?>  
249     (  
250       <AliasRef>wsnt:TopicExpression</AliasRef>  
251       |  
252       <MessagePattern>wsrp:QueryExpression</MessagePattern>?
```

253
254
255
256

```
<Topic ... />*  
...  
)  
</Topic>
```

257 A Topic element is further constrained in the following way:

258 /wstop: Topic

259 This describes the definition of a Topic. Its contents MUST be either a single /AliasRef child
260 element or an optional /MessagePattern child element followed by zero or more child Topic
261 elements.

262 The namespace of a Topic is defined as the targetNamespace of the TopicSpace element
263 ancestor of the Topic. As we saw in section 5, individual root topics are modeled by defining
264 Topic child elements of the TopicSpace element.

265 /wstop: Topic/@name

266 The NCName of this topic. This attribute is required. These NCNames must all be unique
267 with respect to the parent element (TopicSpace or Topic) that contains this Topic. In the case
268 of a root Topic, the @namespace and @name attributes combine to form the QName of the
269 root Topic.

270 /wstop: Topic/@messageTypes

271 An optional list of the QNames of XML elements that define the types of NotificationMessage
272 that may be used with the Topic. A Publisher using a given Topic MUST NOT generate a
273 NotificationMessage whose type is not included in this list, although the special value xsd:any
274 indicates that any NotificationMessage type MAY be used. A given QName MAY appear
275 multiple times in the list; second or subsequent appearance of a given QName are not
276 meaningful and MAY BE ignored. If this list is empty, or the attribute not defined, the default
277 value of "xsd:any" is assumed.

278 /wstop: Topic/@final

279 An optional attribute whose value is of type xsd:boolean. The default value is "false". If the
280 value is "true" it indicates that the NotificationProducer cannot dynamically add child Topics to
281 this Topic. This means that it is an error if a Publisher or Subscriber attempts to use a
282 TopicExpression that references child Topics of a Topic that is marked as @final="true" –
283 other than child Topics that are explicitly included in the definition of the Topic.

284 /wstop: Topic/AliasRef

285 This element indicates that the Topic definition is an alias for another Topic (or set of Topics).
286 This mechanism can be used to permit alternative spellings of a given Topic name, or to
287 allow a Topic (sub)tree from one Topic Space to be imported into a Topic definition in another
288 Topic Space. The contents of an AliasRef element is a TopicExpression that may resolve to
289 multiple Topics, including further aliases (even possibly itself). Publishing or subscribing
290 using a Topic which is an alias is equivalent to publishing or subscribing to all the non-alias
291 Topics which result from the process of alias resolution. This process is described in greater
292 detail in the next section. A Topic containing an AliasRef child element MAY contain
293 @messageTypes, or @final– however if it does their values SHOULD be ignored. The
294 algorithm for resolving AliasRef elements is described in section 8.

295 /wstop: Topic/AliasRef/@dialect

296 A URI that identifies the TopicExpression dialects used in the AliasRef component. This

297 document defines the URIs for three TopicExpression languages. The designer MAY choose
298 from these URIs or use a URI associated with a TopicExpression dialect defined elsewhere.

299 /wstop:Topic/MessagePattern

300 An optional QueryExpression as defined by WS-ResourceProperties. This QueryExpression
301 is used to describe the pattern of the message that will appear on the Topic. Conceptually,
302 the MessagePattern component can be thought of as the object of an boolean() expression,
303 evaluated against a NotificationMessage. This boolean() expression, with the value of
304 MessagePattern as parameter, is guaranteed to evaluate to “true” when evaluated in the
305 context of any NotificationMessage that is associated with the Topic. The MessagePattern
306 component constrains the NotificationMessages that can be used with the Topic. It is
307 additional to the constraint contained in @messageTypes, and provides a further refinement
308 to that constraint.

309 /wstop:Topic/MessagePattern/@dialect

310 A URI that identifies the language of the QueryExpression. WS-ResourceProperties defines
311 standard URIs for XPath 1.0 and XPath 2.0 languages. Designers MAY define and use other
312 domain-specific URIs to identify the dialect of the QueryExpression.

313 /wstop:Topic/Topic

314 Declares a child Topic. A Topic may contain any number of child Topic elements; however
315 the value of the @name attribute of a child Topic must be unique amongst all the child Topics
316 of its immediate parent.

317 /wstop:Topic/{any}

318 This is an extensibility mechanism to allow additional elements to be specified.

319 /wstop:Topic/@{any}

320 This is an extensibility mechanism to allow additional attributes to be specified.

321 7 Topic Expressions

322 Topics are referred to by TopicExpressions. There are several places in WS-Notification where
323 these expressions are used:

- 324 § As a component of the Subscribe message request to a NotificationProducer;
- 325 § As a component of the Notify message to a NotificationConsumer or NotificationBroker;
- 326 § In the Topics Resource Property element(s) associated with the NotificationProducer role
- 327 § In the aliasRef attribute of a Topic element.

328 A non-normative syntax for a TopicExpression is shown below:

```
329 <wsnt:TopicExpression dialect=anyURI?>  
330   dialect-specific expression  
331   ...  
332 </wsnt:TopicExpression>
```

333 A topic expression has two components:

334 /wsnt:TopicExpression/@dialect

335 The dialect component contains a URI which identifies the type of grammar used in the
336 TopicExpression. This URI may be one from the set defined in this document, or may be a
337 URI defined elsewhere.

338 /wsnt:TopicExpression/{any}

339 The content of the TopicExpression is an expression in the grammar defined by the
340 expression language identified by the @dialect component.

341 The purpose of a TopicExpression is to identify a relevant set of Topics from one or more Topic
342 Spaces.

343 7.1 SimpleTopic Expressions

344 This specification defines a simple TopicExpression dialect with the following URI:

```
345 http://docs.oasis-open.org/wsn/2004/06/TopicExpression/Simple
```

346 This dialect is defined to standardize a very simple Topic Expression language for use by
347 resource constrained entities in the WS-Notification system that deal only with simple Topic
348 Spaces.

349 A TopicExpression in this dialect is a token (as defined by XML Schema) with an additional
350 constraint on its format. The constraint is the token must contain a TopicExpression. The
351 grammar is defined using the simple Extended Backus Naur Form (EBNF) also used in [XML]:

```
352 [1] TopicExpression ::= RootTopic  
353 [2] RootTopic ::= QName  
354 [ vc: If a namespace is included in the RootTopic, it must correspond to a valid Topic  
355 Space definition and the local name must correspond to the name of a root topic defined  
356 in that namespace.]
```

357 Because the only valid TopicExpression in this dialect is a QName, only root topics can be
358 addressed by this grammar. For those entities that support only this dialect of TopicExpression,
359 only simple topic spaces, those that define only root topics, SHOULD be used.

360 An example TopicExpression within this dialect is shown below:

```

361 ...
362   xmlns:tns=...
363
364 <wsnt:TopicExpression
365   dialect=" http://docs.oasis-open.org/wsn/2004/06/TopicExpression/Simple">
366   tns:t1
367 </wsnt:TopicExpression>

```

368 This TopicExpression identifies the root Topic t1 within the Topic Space corresponding to the
369 namespace prefix tns.

370 7.2 ConcreteTopicPath Expressions

371 This specification defines a simple path-based TopicExpression dialect with the following URI:

```
372 http://docs.oasis-open.org/wsn/2004/06/TopicExpression/Concrete
```

373 The ConcreteTopicPath expression is used to identify a single Topic within a Topic Space, using
374 a path notation.

375 A TopicExpression in this dialect is a token (as defined by XML Schema) with an additional
376 constraint on its format. The constraint is the token must contain a TopicExpression. The
377 grammar is defined using the simple Extended Backus Naur Form (EBNF) also used in [XML]:

```

378 [3] TopicExpression ::= TopicPath
379 [4] TopicPath ::= RootTopic ChildTopicExpression*
380 [5] RootTopic ::= QName
381 [ vc: If a namespace is included in the RootTopic, it must correspond to a valid Topic
382 Space Document and the local name must correspond to the name of a root topic defined
383 in that namespace.]
384 [6] ChildTopicExpression ::= '/' ChildTopicName
385 [7] ChildTopicName ::= NCName
386 [ vc: The NCName must correspond to the name of a topic within the descendant path
387 from the RootTopic, where each forward slash denotes another level of child topic
388 elements in the path.]

```

389 Note: White space is not permitted within a ConcreteTopicPath expression.

390 An example TopicExpression within this dialect is shown below:

```

391 ...
392   xmlns:tns=...
393
394 <wsnt:TopicExpression
395   dialect=" http://docs.oasis-open.org/wsn/2004/06/TopicExpression/Concrete">
396   tns:t1/t3
397 </wsnt:TopicExpression>

```

398 The TopicPath expression identifies the Topic named "t3", child of Topic tns:t1.

399 As with XPath, this TopicPath expression syntax uses the slash ("/") to describe *child of*.

400 Note: The simple Topic Expression dialect defined in the previous section is a subset of the
401 ConcreteTopicPath Expression dialect.

402 7.3 FullTopicPath Expressions

403 This specification defines a fully featured path-based TopicExpression dialect with the following
404 URI:

405 <http://docs.oasis-open.org/wsn/2004/06/TopicExpression/Full>

406 FullTopicPath expressions consist are made up of XPath [XPath] relative location path
407 expressions with optional Namespace prefixes. The XPath expression is evaluated over a
408 document whose nodes are made up of the topics in the topic space, and where topics include
409 their child topics as contained XML elements (note that this document is not the same as the
410 topic space document described earlier, but can be derived from it). The document root element
411 is not itself a topic, so that root topics in the topic space appear as first-level children of the
412 document root. The TopicExpression selects the set of topics that correspond to the node-set that
413 results from evalutaing the location path contained in the TopicExpression, using standard XPath.
414 The initial context node for this evaluation is the document root element.

415 The FullTopicPath dialect does not permit the use of the entire XPath language. This specification
416 provides syntactic constraints on the contents of the FullTopicPathExpression, that limits the
417 constructs that can be used

418 A TopicExpression in this dialect is a token (as defined by XML Schema) with an additional
419 constraint on its format. The constraint is the token must conform to production rule [1] in the
420 following grammar. This grammar is defined using the simple Extended Backus Naur Form
421 (EBNF) also used in [XML]:

```
422 [1] TopicExpression ::= TopicPath | ConjoinedTopicExpression
423 [2] ConjoinedTopicExpression ::= TopicExpression Conjunction
424                               TopicExpression
425 [3] Conjunction ::= '|'
426 [4] TopicPath ::= RootTopic ChildTopicExpression*
427 [5] RootTopic ::= NamespacePrefix? ('//')? (NCName | '*' )
428 [ vc: If a namespace is included in the RootTopic, it must correspond to a valid Topic
429 Space Document and the local name must correspond to the name of a root topic defined
430 in that namespace.]
431 [6] NamespacePrefix ::= NCName ':'
432 [7] ChildTopicExpression ::= '/' '/'? (ChildTopicName | '*' | '.')
433 [8] ChildTopicName ::= NCName
434 [ vc: The NCName must correspond to the name of a topic within the descendant path
435 from the RootTopic, where each forward slash denotes another level of child topic
436 elements in the path.]
```

437 In this grammar, each TopicPath [4] is to be interpreted as an XPath location path evaluated over
438 the document derived from the Topic Space designated by the NamespacePrefix.

439 Note: White space is not permitted within a FullTopicPath expression.

440 Note: The ConcreteTopicPath dialect defined in the previous section is a subset of the
441 FullTopicPath Expression that contains no wildcards and no '|' operators.

442 The dialect is further explained by the following examples (for the sake of brevity, the examples
443 show only the content of the TopicExpression element):

444 The wildcard character * is used to identify a node-set consisting of a collection of child Topics.
445 For example

446 `"tns:t1/*"`

447 This TopicExpression identifies all the child Topics of the root Topic t1. Note that this
448 TopicExpression does not include the root Topic t1 itself, and it does not include any
449 grandchildren or further descendents of t1.
450 Wildcard characters may be interspersed with fixed child Topic names, to build up longer paths,
451 for example:

```
“tns:t1/*t3”
```

452 This TopicExpression identifies all grandchildren of tns:t1 that have the name t3.
453 The wildcard * may also be used in place of a root topic name, for example:

```
“tns:*”
```

454 This TopicExpression identifies all root topics in the tns: Topic Space.
455 As in full XPATH the // separator is used to identify all descendents (subject of course to the
456 constraints implied by the remainder of the path), not just immediate children.
457 If the TopicExpression ends with the characters “/.” this indicates that the TopicExpression
458 matches a Topic subtree. For example:

```
“tns:t1/t3/.”
```

459 This identifies the subtree consisting of tns:t1/t3 and all its descendents.
460 If the TopicExpression ends with the characters “//*” this indicates that the TopicExpression
461 matches all the descendents of a topic. For example:

```
“tns:t1/t3//*”
```

462 This identifies the subtree consisting of the descendents of tns:t1/t3 but, unlike the previous
463 example, does not include tns:t1/t3 itself.
464 To include all the topics in the entire Topic Space the following TopicExpression can be used:

```
“tns://*”
```

465 The // separator can also be used in the middle of a TopicExpression, for example

```
“tns:t1//t3”
```

466 This TopicExpression identifies all descendents of tns:t1 that have the name t3.
467 A TopicExpression MAY contain two or more wildcards (both * and //).
468 TopicExpressions may be combined together with the conjunction operator as follows:

```
“tns:t1/t2 | tns:t4/t5”
```

469 A TopicExpression using | can include root Topics from different Topic Spaces. Note: a
470 TopicExpression containing a conjunction operator is equivalent to the set union of the Topics
471 described by combining the TopicExpression on either side of the conjunction operator.

472 **7.3.1 Validating FullTopicExpressions**

473 If the NotificationProducer permits it, the FullTopicExpression dialect can be used as the
474 TopicExpression in the Subscribe message [WS-BaseNotification]. Such TopicExpressions MAY
475 refer to one or more topics which may or may not exist in the Topic Space, or in the Topic Set
476 supported by the NotificationProducer.

484 The NotificationProducer MUST validate the TopicExpression as follows.

485 § If the TopicExpression explicitly refers to a Topic that is not permitted by the Topic
486 Space, then the NotificationProducer MUST respond with a Fault. A Topic is not
487 permitted if it is a root topic, or a descendent of a root topic, that is not defined in the
488 Topic Space. A Topic is also not permitted if it, or any of its ancestors, are not defined in
489 the Topic Space and are the child of a Topic that is defined with @final='true'.

490 § If the NotificationProducer has a fixed Topic Set, and the intersection of the topics
491 selected by the TopicExpression with this Topic Set is empty, then the
492 NotificationProducer MUST respond with a Fault.

493 Here are some examples to illustrate these rules:

494 Suppose that Topic Space tns1 contains root topics tns1:A (@final= "true") and tns1:B (@final =
495 "false"), and that NotificationProducer X has a fixed topic set consisting just of tns1:B.

496 § Any subscribe with a TopicExpression containing tns1:D is rejected

497 § Any subscribe with a TopicExpression containing tns1:A/X is rejected

498 § A subscribe to tns1:B/X is rejected, but would be permitted if X did not have a fixed topic
499 set.

500 § A subscribe to tns1:A is rejected, but would be permitted if X did not have a fixed topic
501 set.

502 § A subscribe to tns1:* is permitted (and is equivalent in this case to a subscribe to tns1:B)

503 § A subscribe to tns1://* is permitted (and is equivalent in this case to a subscribe to
504 tns1:B)

505 § A subscribe to tns1:A | tns1:B is permitted (and is equivalent in this case to a subscribe to
506 tns1:B)

507

8 AliasRefs and their resolution

508

The AliasRef is an optional child element of a Topic element that indicates that the Topic is an

509

alias for another Topic (or combination of topics). This mechanism can be used to permit

510

alternative spellings of a given Topic name, or to allow a Topic (sub)tree from one TopicSpace to

511

be *imported* into a Topic definition in another Topic Space. In this example Topic t6 is defined as

512

an alias for tns:t1/t3

513

```
<wstop:topic name="t6">
```

514

```
<wstop:AliasRef
```

515

```
dialect=" http://docs.oasis-open.org/wsn/2004/06/TopicExpression/Concrete " >
```

516

```
tns:t1/t3
```

517

```
</wstop:AliasRef>
```

518

```
</wstop:topic>
```

519

An AliasRef MAY contain any TopicExpression, including those expressing wild cards or '|'

520

operators. This means that an AliasRef might reference another AliasRef Topic definition, or

521

might be a wild card expression that includes a mixture of alias and non-alias definitions.

522

An AliasRef is resolved into a set of zero or more non-aliased Topics using the following rules:

523

§ If the AliasRef is a concrete TopicPath expression that is not an alias, then the alias

524

resolves to the Topic identified by that concreteTopicPath expression.

525

§ If the AliasRef is a concreteTopicPath expression that is itself identifies an alias, then

526

resolution proceeds recursively from this alias.

527

§ If the AliasRef is a FullTopicPath expression with *, // but no '|' operators, then the alias

528

resolves to this set of Topics (no deeper examination of aliases is performed in this

529

case).

530

§ If the AliasRef is a FullTopicPath expression containing '|' operators, then each

531

component of the expression is treated separately and creates a new resolution branch.

532

Resolution proceeds on each branch individually, using rules 1,2,3, the resolution of each

533

branch is aggregated to the resolved Topic set.

534

§ If a circular reference is encountered (AliasRef pointing directly or indirectly back to itself)

535

then the branch in question contributes nothing to the resolved Topic set.

536

If a TopicExpression is supplied as a parameter on a message exchange defined by any of the

537

WS-Notification specifications, it is subjected to the alias resolution process described above. The

538

resulting resolved Topic set is then used in place of the original parameter. If the resolved Topic

539

set is empty, the operation MUST fail. If the operation required a concrete topic and the resolved

540

set contains multiple topics, or contains wild card topic expressions, then the operation MUST fail.

541 **9 Growing a Topic Tree**

542 If a Topic in the TopicSpace is marked with the final attribute, with value="true", then no further
543 child Topics can be added dynamically to that Topic.

544 If a Topic is not marked with the final attribute with value="true", then a NotificationProducer could
545 potentially add further child Topics to that Topic, and permit Subscriptions to such child Topics.
546 This specification does not define the circumstances under which this occurs, and it is up to the
547 NotificationProducer to determine if and when it permits additional children (it is not obligated to
548 allow children to be added just because a Topic may be marked with final="false").

549 When a NotificationProducer accepts Topics that are not previously defined in the TopicSpace, it
550 is not obliged to update any actual instance document that contains the TopicSpace definition.

551 Rather, the extension exists only for that NotificationProducer and any Subscriber that interacts
552 with it. Circumstances under which a NotificationProducer MAY add new child Topics to a Topic
553 include:

554 § a Subscriber attempting to subscribe to a TopicExpression that suggests a new child
555 Topic;

556 § a Publisher attempting to publish to a TopicExpression that suggests a new child Topic;

557 § the NotificationProducer implementation encountering a new circumstance that doesn't fit
558 well with any of the existing child Topics (for example a new company starts trading on a
559 stock market, and a stock ticker service wishes to include it);

560 § an administrator explicitly adding support for a new child Topic using some administrative
561 portType (not defined by any WS-Notification specification) implemented by the
562 NotificationProducer.

563 **10The “ad-hoc” Topic Space**

564 Associating a TopicSpace with an XML namespace provides an unambiguous naming scheme
565 for Topics. This is important when two entities which have no prior knowledge of each other
566 attempt (for example a Subscriber which has just discovered a NotificationBroker) to interact.

567 However there are circumstances where someone wishes to implement a Publisher for which
568 there is no suitable pre-existing TopicSpace – and where the implementer does not wish to incur
569 the overhead of creating a new TopicSpace (assigning a unique namespace, and creating the
570 TopicSpace element within some XML instance document).

571 To help such users, WS-Notification defines a special built-in TopicSpace called the *ad-hoc*
572 TopicSpace.

573 The ad-hoc TopicSpace has no pre-defined root Topics, but allows new root Topics to be added
574 dynamically (in the same way that a non-final Topic allows new child Topics to be added to it).
575 Any Topic that is added dynamically to the ad-hoc TopicSpace itself permits the addition of
576 further child Topics, and allows any type of NotificationMessage element to be associated with it.
577 There is no concept of Topic aliasing in the ad-hoc TopicSpace.

578 The ad-hoc TopicSpace is defined by the following namespace URI ([http://docs.oasis-
579 open.org/wsn/2004/06/TopicSpaces/adHoc](http://docs.oasis-open.org/wsn/2004/06/TopicSpaces/adHoc)) and is accessed using TopicExpressions that
580 reference this namespace.

581 A NotificationProducer or Subscriber can use this TopicSpace to define *ad-hoc Topics*
582 dynamically, without having to associate them with their own TopicSpace. Caution should be
583 used when employing ad-hoc Topics, as there is no way for a NotificationConsumer to distinguish
584 between it and other similarly-named ad-hoc Topics supported by any number of
585 NotificationProducers.

586 **11 NotificationProducer and Topics**

587 A NotificationProducer uses Topics to group NotificationMessages related to some Situation. A
588 NotificationProducer can support one or more Topics, from multiple Topic Spaces. A
589 NotificationProducer can support an entire Topic Tree, or just a subset of the Topics in that Topic
590 Tree. The set of Topics currently supported by a NotificationProducer can be determined by
591 accessing the Topics Resource Property element (see [WS-BaseNotification]). This Resource
592 Property contains the set of Topics that the NotificationProducer expects to handle.

593 The list of Topics supported by the NotificationProducer MAY change over time. Reasons for the
594 set of Topics changing include:

- 595 § The NotificationProducer supporting additional Topics from a TopicSpace that is already
596 partially supported;
- 597 § The NotificationProducer supporting additional Topics from a TopicSpace not previously
598 supported;
- 599 § The NotificationProducer supporting extension Topics to a (new or already supported)
600 TopicSpace, as discussed in the previous section;
- 601 § The NotificationProducer ceasing to support Topics previously listed.

602 This specification does not require a NotificationProducer to support any or all of the types of
603 changes just listed, and does not dictate the set of conditions under which the list of supported
604 Topics will change.

605 **12Security Considerations**

606 A non-normative discussion of the security scenarios and considerations associated with the
607 entire family of WS-Notification specifications is contained in [WS-Notification WP].

13References

- 608
- 609 **[RFC2119]** S. Bradner, *Key words for use in RFCs to Indicate Requirement*
610 *Levels*, <http://www.ietf.org/rfc/rfc2119.txt>, IETF RFC 2119, March
611 1997.
- 612 **[WS-BaseNotification]** <http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BaseNotification-1.2-draft-03.pdf>
613
- 614 **[WS-BrokeredNotification]**<http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BrokeredNotification-1.2-draft-01.pdf>
615
- 616 **[WS-Notification WP]** <http://www.oasis-open.org/committees/download.php/6661/WSNpubsub-1-0.pdf>
617
- 618 **[WS-Security]** <http://www.oasis-open.org/committees/download.php/5531/oasis-200401-wss-soap-message-security-1.0.pdf>
619
620
- 621 **[XML]** <http://www.w3.org/TR/REC-xml>
- 622 **[XML-Infoset]** <http://www.w3.org/TR/xml-infoset/>
- 623 **[XPATH]** <http://www.w3.org/TR/xpath>

624 **Appendix A. Acknowledgments**

625 The following individuals were members of the committee during the development of this
626 specification:

627 Geoff Acton, LSC Group Ltd, Sid Askary, Individual, Jeff Bohren, OpenNetwork, Fred Carter,
628 AmberPoint, Martin Chapman, Oracle, Dave Chappell, Sonic Software, Ugo Corda, SeeBeyond
629 Technology Corporation, Glen Daniels, Sonic Software, John Fuller, Individual, Stephen Graham,
630 IBM, David Hull, Tibco, Anish Karmarkar, Oracle, John Kemp, Nokia, Paul Lipton, Computer
631 Associates, Lily Liu, webMethods, Tom Maguire, IBM, Susan Malaika, IBM, David Martin, IBM,
632 Samuel Meder, Argonne National Laboratory, Jeff Mischkinsky, Oracle, Bryan Murray, Hewlett-
633 Packard, Peter Niblett, IBM, Dave Orchard, BEA Systems, Sanjay Patil, SAP, Greg Pavlik,
634 Oracle, Mark Peel, Novell, Mark Piller, webMethods, Ian Robinson, IBM, Igor Sedukhin,
635 Computer Associates, David Snelling, Fujitsu, Latha Srinivasan, Hewlett-Packard, John
636 Tollefsrud, Sun Microsystems, Steve Tuecke, Globus / Argonne National Laboratory, Jem
637 Treadwell, Hewlett-Packard, William Vambenepe, Hewlett-Packard, Alan Weissberger, NEC

638 Special thanks to the Global Grid Forum's Open Grid Services Infrastructure working group,
639 which defined the OGSi v1.0 specification which was a large inspiration for the ideas expressed
640 in this specification.

641 In addition, the following people who are not members of the committee made contributions to
642 this specification:

643 Tim Banks (IBM), Nick Butler (IBM), Doug Davis (IBM), John Dinger (IBM), Don Ferguson (IBM),
644 Jeff Frey (IBM), Andreas Koepfel (SAP), Heather Kreger (IBM), Amy Lewis (TIBCO Software),
645 Kevin Liu (SAP), Nataraj Nagaratnam (IBM), Martin Nally (IBM), Jeff Nick (IBM), Jay Parikh
646 (Akamai Technologies), Claus von Riegen (SAP), Rick Rineholt (IBM), John Rofrano (IBM),
647 Shivajee Samdarshi (TIBCO Software), Eugène Sindambiwe (SAP), Jay Unger (IBM), Bill Weihl
648 (Akamai Technologies), Mark Weitzel (IBM), Dan Wolfson (IBM).

649 Appendix B. XML Schema

650 The XML types and elements used in this specification are defined in the following XML Schema:

```
651 <?xml version="1.0" encoding="UTF-8"?>
652 <!--
653
654 OASIS takes no position regarding the validity or scope of any intellectual property or
655 other rights that might be claimed to pertain to the implementation or use of the
656 technology described in this document or the extent to which any license under such
657 rights might or might not be available; neither does it represent that it has made any
658 effort to identify any such rights. Information on OASIS's procedures with respect to
659 rights in OASIS specifications can be found at the OASIS website. Copies of claims of
660 rights made available for publication and any assurances of licenses to be made
661 available, or the result of an attempt made to obtain a general license or permission for
662 the use of such proprietary rights by implementors or users of this specification, can be
663 obtained from the OASIS Executive Director.
664
665 OASIS invites any interested party to bring to its attention any copyrights, patents or
666 patent applications, or other proprietary rights which may cover technology that may be
667 required to implement this specification. Please address the information to the OASIS
668 Executive Director.
669
670 Copyright (C) OASIS Open (2004). All Rights Reserved.
671
672 This document and translations of it may be copied and furnished to others, and
673 derivative works that comment on or otherwise explain it or assist in its implementation
674 may be prepared, copied, published and distributed, in whole or in part, without
675 restriction of any kind, provided that the above copyright notice and this paragraph are
676 included on all such copies and derivative works. However, this document itself may not
677 be modified in any way, such as by removing the copyright notice or references to
678 OASIS, except as needed for the purpose of developing OASIS specifications, in which
679 case the procedures for copyrights defined in the OASIS Intellectual Property Rights
680 document must be followed, or as required to translate it into languages other than
681 English.
682
683 The limited permissions granted above are perpetual and will not be revoked by OASIS
684 or its successors or assigns.
685
686 This document and the information contained herein is provided on an "AS IS" basis
687 and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING
688 BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION
689 HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF
690 MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.
691
692 -->
```

```

693
694
695 <xsd:schema
696   xmlns="http://www.w3.org/2001/XMLSchema"
697   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
698   xmlns:wsrp=
699     "http://docs.oasis-open.org/wsrp/2004/06/wsrp-WS-ResourceProperties-1.2-draft-
700 01.xsd"
701   xmlns:wsnt=
702     "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BaseNotification-1.2-draft-01.xsd"
703   xmlns:wstop=
704     "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-Topics-1.2-draft-01.xsd"
705   targetNamespace=
706     "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-Topics-1.2-draft-01.xsd"
707   elementFormDefault="qualified" attributeFormDefault="unqualified">
708
709 <!-- ===== Imports ===== -->
710
711   <xsd:import namespace=
712     "http://docs.oasis-open.org/wsrp/2004/06/wsrp-WS-ResourceProperties-1.2-draft-
713 01.xsd"
714     schemaLocation=
715     "http://docs.oasis-open.org/wsrp/2004/06/wsrp-WS-ResourceProperties-1.2-draft-
716 01.xsd"
717   />
718   <xsd:import namespace=
719     "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BaseNotification-1.2-draft-01.xsd"
720     schemaLocation=
721     "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BaseNotification-1.2-draft-01.xsd"
722   />
723
724 <!-- ===== utility type definitions ===== -->
725   <xsd:complexType name="Documentation" mixed="true">
726     <xsd:sequence>
727       <xsd:any processContents="lax" minOccurs="0"
728         maxOccurs="unbounded" namespace="##any"/>
729     </xsd:sequence>
730   </xsd:complexType>
731
732   <xsd:complexType name="ExtensibleDocumented" abstract="true"
733     mixed="false">
734     <xsd:sequence>
735       <xsd:element name="documentation" type="wstop:Documentation"
736         minOccurs="0" />
737     </xsd:sequence>
738     <xsd:anyAttribute namespace="##other" processContents="lax" />
739   </xsd:complexType>

```

```

740
741 <!-- ===== Topic-Space Related ===== -->
742 <xsd:complexType name="TopicSpaceType">
743   <xsd:complexContent>
744     <xsd:extension base="wstop:ExtensibleDocumented">
745       <xsd:sequence>
746         <xsd:element name="Topic" type="wstop:TopicType"
747           minOccurs="0" maxOccurs="unbounded"/>
748         <xsd:any namespace="##other"
749           minOccurs="0" maxOccurs="unbounded"
750           processContents="lax"/>
751       </xsd:sequence>
752       <xsd:attribute name="name" type="xsd:NCName"/>
753       <xsd:attribute name="targetNamespace" type="xsd:anyURI"
754         use="required"/>
755     </xsd:extension>
756   </xsd:complexContent>
757 </xsd:complexType>
758
759 <xsd:element name="TopicSpace" type="wstop:TopicSpaceType">
760   <xsd:unique name="rootTopicUniqueness">
761     <xsd:selector xpath="wstop:Topic"/>
762     <xsd:field xpath="@name"/>
763   </xsd:unique>
764 </xsd:element>
765
766 <!-- ===== Topic Related ===== -->
767
768 <xsd:group name="NonAliasTopicDefinition">
769   <xsd:sequence>
770     <xsd:element name="MessagePattern"
771       type="wsrp:QueryExpressionType"
772       minOccurs="0" maxOccurs="1" />
773     <xsd:element name="Topic" type="wstop:TopicType"
774       minOccurs="0" maxOccurs="unbounded">
775       <xsd:unique name="childTopicUniqueness">
776         <xsd:selector xpath="wstop:topic"/>
777         <xsd:field xpath="@name"/>
778       </xsd:unique>
779     </xsd:element>
780   </xsd:sequence>
781 </xsd:group>
782
783 <xsd:complexType name="TopicType">
784   <xsd:complexContent>
785     <xsd:extension base="wstop:ExtensibleDocumented">
786     <xsd:sequence>

```

```

787 <xsd:choice>
788   <xsd:element name="AliasRef"
789     type="wsnt:TopicExpressionType"
790     minOccurs="1" maxOccurs="1" />
791   <xsd:group ref="wstop:NonAliasTopicDefinition" />
792 </xsd:choice>
793 <xsd:any namespace="##other" minOccurs="0"
794   maxOccurs="unbounded"/>
795 </xsd:sequence>
796 <xsd:attribute name="name" use="required" type="xsd:NCName"/>
797 <xsd:attribute name="messageTypes" default="xsd:any">
798   <xsd:simpleType>
799     <xsd:list itemType="xsd:QName"/>
800   </xsd:simpleType>
801 </xsd:attribute>
802 <xsd:attribute name="final" type="xsd:boolean"
803   default="false"/>
804 </xsd:extension>
805 </xsd:complexContent>
806 </xsd:complexType>
807
808 <!-- ===== Topic Expression Related ===== -->
809
810 <xsd:simpleType name="FullTopicPathExpression">
811   <xsd:restriction base="xsd:token">
812     <xsd:annotation>
813       <xsd:documentation>
814         TopicPathExpression ::= TopicPath ( '|' TopicPath )*
815         TopicPath ::= RootTopic ChildTopicExpression*
816         RootTopic ::= NamespacePrefix? ('/')? (NCName | '*')
817         NamespacePrefix ::= NCName ':'
818         ChildTopicExpression ::= '/' '/'? (NCName | '*'| '.')
819
820       </xsd:documentation>
821     </xsd:annotation>
822     <xsd:pattern value=
823       "([i-:]][c-:]*)?(//)?([i-:]][c-:]*\|)*((//)([i-:]][c-:]*\|[\.]*)*\|([i-:]][c-
824       [:]]*?)?(//)?([i-:]][c-:]*\|)*((//)([i-:]][c-:]*\|[\.]*)*">
825     </xsd:pattern>
826   </xsd:restriction>
827 </xsd:simpleType>
828
829 <xsd:simpleType name="ConcreteTopicPathExpression">
830   <xsd:restriction base="xsd:token">
831     <xsd:annotation>
832       <xsd:documentation>
833       The pattern allows strings matching the following EBNF:

```

```

834 ConcreteTopicPath ::= RootTopic ChildTopic*
835 RootTopic ::= QName
836 ChildTopic ::= '/' NCName
837
838 </xsd:documentation>
839 </xsd:annotation>
840 <xsd:pattern value=
841 "(( [N-:]][\c-:]*:)? [N-:][\c-:]*)(/[N-:][\c-:]*)*" >
842 </xsd:pattern>
843 </xsd:restriction>
844 </xsd:simpleType>
845
846 <xsd:simpleType name="SimpleTopicExpression">
847 <xsd:restriction base="xsd:token">
848 <xsd:annotation>
849 <xsd:documentation>
850 The pattern allows strings matching the following EBNF:
851 RootTopic ::= QName
852
853 </xsd:documentation>
854 </xsd:annotation>
855 <xsd:pattern value="([N-:][\c-:]*:)?([N-:][\c-:]*)" >
856 </xsd:pattern>
857 </xsd:restriction>
858 </xsd:simpleType>
859
860 </xsd:schema>

```

861

Appendix C. Revision History

Rev	Date	By Whom	What
wd-01	2004-06-04	William Vambenepe	Initial version created from submission by contributing companies. Minor modifications made to reflect OASIS formatting and namespace URI choices.

862

863 **Appendix D. Notices**

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

873

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

877

878 Copyright (C) OASIS Open (2004). All Rights Reserved.

879

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

889

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

892

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