



Web Services Topics 1.3 (WS-Topics)

OASIS Standard, 1 October 2006

Document identifier:

wsn-ws_topics-1.3-spec-os

Location:

http://docs.oasis-open.org/wsn/wsn-ws_topics-1.3-spec-os.pdf

Editors:

William Vambenepe, HP <vbp@hp.com>

Steve Graham, IBM <sggraham@us.ibm.com>

Peter Niblett, IBM <peter_niblett@uk.ibm.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 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: 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-BaseNotification. 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

34 describing metadata associated with topics. This specification should be read in
35 conjunction with the WS-Base Notification specification.

36 **Status:**

37 This document is an OASIS standard.

38 Committee members should send comments on this specification to the [wsn@lists.oasis-](mailto:wsn@lists.oasis-open.org)
39 [open.org](http://lists.oasis-open.org) list. Others may submit comments to the TC via the web form found on the
40 TC's web page at <http://www.oasis-open.org/committees/wsn>. Click the button for "Send
41 A Comment" at the top of the page. Submitted comments (for this work as well as other
42 works of the TC) are publicly archived and can be viewed at [http://lists.oasis-](http://lists.oasis-open.org/archives/wsn-comment/)
43 [open.org/archives/wsn-comment/](http://lists.oasis-open.org/archives/wsn-comment/).

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

48

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	6
56	2	Terminology and Concepts	7
57	3	Topics and Topic Namespaces	8
58	4	Example	10
59	5	Modeling Topic Namespaces in XML	12
60	6	Modeling Topics in XML	13
61	6.1	Extension Topics	14
62	7	Modeling Topic Sets in XML	16
63	8	Topic Expression Dialects	18
64	8.1	Simple TopicExpression Dialect	18
65	8.2	Concrete TopicExpression Dialect	19
66	8.3	Full TopicExpression Dialect	21
67	8.4	XPath TopicExpression Dialect	23
68	8.5	Validating TopicExpressions	23
69	9	Growing a Topic Tree	25
70	10	The “ad-hoc” Topic Namespace	26
71	11	NotificationProducers and Topics	27
72	12	Security Considerations	29
73	13	References	30
74		Appendix A. Acknowledgments	31
75		Appendix B. XML Schema	32
76		Appendix C. Revision History	37
77		Appendix D. Notices	40
78			

79 **1 Introduction**

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

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

87 WS-Topics defines four topic expression dialects that can be used as subscription expressions in
88 subscribe request messages and other parts of the WS-Notification system. It further specifies an
89 XML model for describing metadata associated with topics. This specification should be read in
90 conjunction with the WS-BaseNotification specification.

91 **1.1 Goals and Requirements**

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

96 **1.1.1 Requirements**

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

- 98 ▪ Must support resource-constrained devices. The specifications must be factored in a way
99 that allows resource-constrained devices to participate in the Notification pattern. Such
100 devices will be able to send information to, and receive information from Web services,
101 without having to implement all the features of the specifications.
- 102 ▪ Must permit transformation and aggregation of Topics: It must be possible to construct
103 configurations (using intermediary brokers) where the Topic subscribed to by the
104 NotificationConsumer differs from the Topic published to by the NotificationProducer, yet
105 Notifications from the NotificationProducer are routed to the NotificationConsumer by a
106 broker that is acting according to administratively-defined 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

111 **1.1.2 Non-Goals**

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

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

116 1.2 Notational Conventions

117 The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD",
118 "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be
119 interpreted as described in [RFC2119].

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

123 This specification uses a notational convention, referred to as "Pseudo-schemas". A Pseudo-
124 schema uses a BNF-style convention to describe attributes and elements:

- 125 • '?' denotes optionality (i.e. zero or one occurrences),
- 126 • '*' denotes zero or more occurrences,
- 127 • '+' one or more occurrences,
- 128 • '[' and ']' are used to form groups,
- 129 • '|' represents choice.
- 130 • Attributes are conventionally assigned a value which corresponds to their type, as
131 defined in the normative schema.
- 132 • Elements with simple content are conventionally assigned a value which corresponds to
133 the type of their content, as defined in the normative schema.
- 134 • The use of {any} indicates the presence of an element wildcard (<xs:any/>).
- 135 • The use of @{any} indicates the presence of an attribute wildcard (<xs:anyAttribute/>).
- 136 • In the interest of brevity, some extensibility points have been omitted from the Pseudo-
137 schemas.

138

```
139   <!-- sample pseudo-schema -->  
140   <element  
141       required_attribute_of_type_QName="xs:QName"  
142       optional_attribute_of_type_string="xs:string"? >  
143    <required_element />  
144    <optional_element /> ?  
145    <one_or_more_of_these_elements /> +  
146    [ <choice_1 /> | <choice_2 /> ] *  
147   </element>
```

148 Where there is disagreement between the separate XML schema file describing the elements
149 defined by this specification and the normative descriptive text (excluding any pseudo-schema) in
150 this document, the normative descriptive text will take precedence over the separate files. The
151 separate files take precedence over any pseudo-schema and over any schema included in the
152 appendices.

153 **1.3 Namespaces**

154 The following namespaces are used in this document:

Prefix	Namespace
xsd	http://www.w3.org/2001/XMLSchema
wsnt	http://docs.oasis-open.org/wsn/b-2
wstop	http://docs.oasis-open.org/wsn/t-1

2 Terminology and Concepts

155

156 In addition to the terminology and usage defined in the WS-BaseNotification and WS-
157 BrokeredNotification specifications, the following are the terms defined in this specification:

158

159 **Topic:**

- 160 • A Topic is the concept used to categorize Notifications and their related Notification
161 schemas.
- 162 • Topics are used as part of the matching process that determines which (if any)
163 subscribing NotificationConsumers should receive a Notification.
- 164 • When it generates a Notification, a Publisher can associate it with one or more Topics.
165 The relation between Situation (as defined in [WS-BaseNotification]) and Topic is not
166 specified by WS-Notification but MAY be specified by the designer of the Topic
167 Namespace.
- 168 • A synonym in some other publish/subscribe models is *subject*.

169

170 **Topic Namespace:**

- 171 • A forest of Topic Trees grouped together into the same namespace for administrative
172 purposes.

173

174 **Topic Tree:**

- 175 • A hierarchical grouping of Topics.

176

177 **Topic Set:**

- 178 • The collection of Topics supported by a NotificationProducer

179

180

181

3 Topics and Topic Namespaces

182 The WS-Notification specifications allow the use of Topics as a way to organize and categorize a
183 set of Notifications. The Topics mechanism provides a convenient means by which subscribers
184 can reason about Notifications of interest. Topics appear in several places within the WS-
185 Notification system. As part of the publication of a Notification, a Publisher may associate it with
186 one or more Topics. When a Subscriber creates a Subscription, it may supply a Topic filter
187 expression, associating the Subscription with one or more Topics. The NotificationProducer uses
188 these sets of Topics as part of the matching process: a Notification is delivered to a
189 NotificationConsumer if the set of Topics associated with the Subscription has a non-empty
190 intersection with the set of Topics associated with the Notification.

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

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

203 Each Topic in a Topic Namespace can have zero or more *child Topics*, and a child Topic can
204 itself contain further child Topics. A Topic without a *parent* is termed a *root Topic*. A particular root
205 Topic and all its descendents form a hierarchy (termed a *Topic Tree*).

206 The rationale for hierarchical topic structures is:

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

215 Note: Although WS-Notification permits hierarchical topic structures, there is no requirement or
216 expectation that all Topic Namespaces will contain them. It is perfectly possible for a Topic
217 Namespace to contain only root Topics (possibly only a single root Topic). A NotificationProducer
218 may restrict its Topic Set to include only Topics from Topic Namespaces that just contain root
219 Topics; even if it does include Topics from a Topic Namespace that contains topic hierarchies, it
220 may choose only to support root Topics from that Topic Namespace.

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

224 Each Topic has a local name, an NCName as defined by [XML-Namespaces]. All root Topics
225 must have unique names within their Topic Namespace. In this way, a root Topic can be uniquely
226 referenced by a QName formed by combining the XML Namespace associated with the Topic
227 Namespace and the local name of the root Topic. Child Topics can be referred to relative to their
228 ancestor root Topic's QName using a path-based TopicExpression dialect (see section 8).

229 No Topic can contain two immediate child Topics with the same name, however Topics with the
230 same name can appear elsewhere in a Topic Tree, and no relationship is implied. Similarly two
231 separate Topic Trees in the same Topic Namespace can contain Topics with the same name;
232 these are not necessarily related to each other in any way either.

233 WS-Topics allows a Topic Namespace to contain one or more extensions to a Topic Tree that is
234 defined in another Topic Namespace. These extensions can be used as though they were child
235 Topics of Topics in that Topic Namespace. This mechanism allows one organization to define a
236 set of core hierarchical topic structures (in one Topic Namespace), and another organization to
237 add its own Topics (from its own separate Namespace) into this hierarchy.

238

4 Example

239

Consider a Topic Namespace that can be depicted as illustrated by Figure 1. The Topic Namespace is contained in the "http://example.org/topicSpace/example1" namespace. This Topic Namespace 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.

243

244

245

246

247

248

249

250

251

252

253

254

255

256

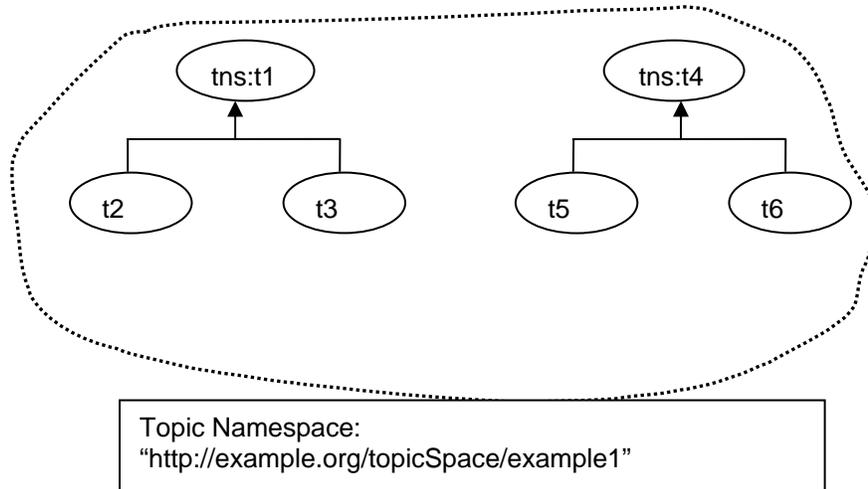


Figure 4: Example Topic Namespace

257

258

259

This Topic Namespace and its metadata can be described using the following XML document:

260

261

262

263

264

265

266

267

268

269

270

271

272

273

274

275

```
<?xml version="1.0" encoding="UTF-8"?>
<wstop:TopicNamespace name="TopicSpaceExample1"
  targetNamespace="http://example.org/topicSpace/example1"
  xmlns:tns="http://example.org/topicSpace/example1"
  xmlns:xyz="http://example.org/anotherNamespace"
  xmlns:wstop="http://docs.oasis-open.org/wsn/t-1"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://docs.oasis-open.org/wsn/t-1
    http://docs.oasis-open.org/wsn/t-1.xsd" >
  <wstop:Topic name="t1">
    <wstop:Topic name="t2" messageTypes="xyz:m1 tns:m2"/>
    <wstop:Topic name="t3" messageTypes="xyz:m3"/>
  </wstop:Topic>
  <wstop:Topic name="t4">
    <wstop:Topic name="t5" messageTypes="tns:m3"/>
    <wstop:Topic name="t6"/>
  </wstop:Topic>
</wstop:TopicNamespace>
```

276
277

```
</wstop:Topic>  
</wstop:TopicNamespace>
```

278

279 This Topic Namespace defines six Topics – the two root Topics and their four children.
280 Continuing with our example, we introduce a NotificationProducer that wishes to use three of
281 these Topics,

- 282 • The root Topic `tns:t1`
- 283 • The `t2` child of `tns:t1`
- 284 • The `t5` child of `tns:t4`

285 The NotificationProducer supports these Topics by adding them to its Topic Set. The Topic Set
286 can itself be represented as an XML document as follows:

287

288
289
290
291
292
293
294
295
296
297
298
299
300
301
302

```
<?xml version="1.0" encoding="UTF-8"?>  
<wstop:TopicSet xmlns:wstop="http://docs.oasis-open.org/wsn/t-1"  
xmlns:tns="http://example.org/topics/example1"  
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
xsi:schemaLocation="http://docs.oasis-open.org/wsn/t-1  
http://docs.oasis-open.org/wsn/t-1.xsd">  
  
  <tns:t1 wstop:topic="true">  
    <t2 wstop:topic="true"/>  
  </tns:t1>  
  <tns:t4>  
    <t5 wstop:topic="true"/>  
  </tns:t4>  
  
</wstop:TopicSet>
```

303

304 The Topic Set document has a root element called TopicSet, and each Topic supported by the
305 NotificationProducer is represented by an element in the document. The Topic name is used as
306 this element's QName, and its position in the document hierarchy matches the position of the
307 Topic in the Topic hierarchy. So root Topics (for example `tns:t1`) appear as children of the
308 TopicSet element, and other Topics are represented by elements that are children of the element
309 that corresponds to their parent Topic.

310 Elements that represent Topics are marked with a `wstop:topic` attribute taking the value "true".
311 This allows the NotificationProducer to insert additional elements that represent other items of
312 metadata; these other items can be distinguished from the elements that represent Topics since
313 they don't have `@wstop:topic="true"`. It also means that the document can represent a Topic Set
314 which includes child Topics without including their parents. In this example the TopicSet
315 document contains a `tns:t4` element, which allows it to include `tns:t4/t5`. However since the
316 `tns:t4` element does not have a `@wstop:topic="true"` the `tns:t4` it does not represent a Topic,
317 so the root Topic does not form part of this Topic Set

318

319 We describe the details behind modeling Topic Namespaces and Topics in the following sections.
[wsn-ws_topics-1.3-spec-os](#) 1 October 2006

5 Modeling Topic Namespaces in XML

320

321 The WS-Topics XML Schema contains element and type definitions used to create Topic
322 Namespace documents. A Topic Namespace document is associated with a single Topic
323 Namespace and contains the names of Topics in that Topic Namespace along with their
324 metadata. It might include all the Topics in that Topic Namespace, or just a subset of them.

325

326 The following pseudo-schema gives a non-normative description of a TopicNamespace element:

```
327 <TopicNamespace name=xsd:NCName? targetNamespace=xsd:anyURI  
328 final=xsd:boolean? >  
329   <Topic ... /*  
330 </TopicNamespace>
```

331 A TopicNamespace element is constrained in the following way:

332 /wstop:TopicNamespace

333 The top-level element in a Topic Namespace document. It contains Topic declaration
334 elements and associates them with the XML Namespace for the Topic Namespace

335 /wstop:TopicNamespace/@name

336 A name that can be assigned to the TopicNamespace element for light-weight documentation
337 purposes.

338 /wstop:TopicNamespace/@targetNameSpace

339 The XML Namespace for this Topic Namespace. It is expressed as a URI. This forms the
340 namespace component of the QName of each root Topic in the Topic Namespace.

341 /wstop:TopicNamespace/@final

342 An attribute whose value is of type *xsd:boolean*. The default value (to be assumed if the
343 attribute is omitted) is "false". If the value is "true" it indicates that any Topic which appears in
344 a NotificationProducer's Topic Set and uses this target namespace MUST have its root
345 explicitly defined in the TopicNamespace.

346 /wstop:TopicNamespace/Topic

347 The TopicNamespace has a collection of zero or more child Topic elements that define the
348 roots of the Topic Trees within the Topic Namespace. The TopicNamespace element can
349 contain any number of Topic elements. The value of /Topic/@name MUST be unique
350 amongst all root Topics defined in the TopicNamespace.

351 /wstop:TopicNamespace/{any}

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

353 /wstop:TopicNamespace/@{any}

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

6 Modeling Topics in XML

355

356 WS-Notification defines an XML representation of a Topic that can be represented as follows:

```
357 <TopicNamespace name=... targetNamespace=...>
358   <Topic name=xsd:NCName messageTypes=list of xsd:QName?
359     final=xsd:boolean? parent=ConcreteTopicExpression? >
360     <MessagePattern>QueryExpressionType</MessagePattern>?
361     <Topic ... /*>
362   </Topic>
363   ...
364 </TopicNamespace>
```

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

366 /wstop:Topic

367 This describes the definition of a Topic. It contains a MessagePattern child element (which
368 can be omitted) followed by zero or more child Topic elements.

369 The namespace of a Topic is defined as the targetNamespace of the TopicNamespace
370 element ancestor of the Topic. As we saw in section 5, individual root Topics are modeled by
371 defining Topic child elements of the TopicNamespace element.

372 /wstop:Topic/@name

373 The NCName of this Topic. This attribute is required. These NCNames must all be unique
374 with respect to the parent element (TopicNamespace or Topic) that contains this Topic. In the
375 case of a root Topic, Topic/@name gives the local name of the Topic, while its namespace is
376 given by the @targetNamespace attribute of the containing TopicNamespace element. A root
377 Topic can be identified using a QName whose prefix is bound to this namespace and whose
378 local part is the local name.

379 /wstop:Topic/@messageTypes

380 A list of the QNames of XML global element declarations (GEDs) that define the kinds of
381 Notification that can be used with the Topic. If the list is present then a Publisher using a
382 given Topic MUST NOT generate a Notification with root element whose QName is not
383 included in this list. If the list is empty, or the attribute is not defined, then a Notification can
384 have any XML element as root. A given QName can appear multiple times in the list; second
385 or subsequent appearance of a given QName are not meaningful and SHOULD be ignored.

386 /wstop:Topic/@final

387 An attribute whose value is of type xsd:boolean. The default value (to be assumed if the
388 attribute is omitted) is "false". If the value is "true" it indicates that the NotificationProducer
389 MUST NOT use child Topics of this Topic other than those explicitly shown in this
390 TopicSpace document. This means that it is an error if a Publisher or Subscriber attempts to
391 use a TopicExpression that references child Topics of a Topic that is marked as @final="true"
392 – other than child Topics that are explicitly included in the definition of the Topic.

393 /wstop:Topic/@parent

394 An attribute whose value is a ConcreteTopicExpression. If present it designates a parent

395 Topic and indicates that this root Topic, and any child Topics descended from it, are
396 extensions of that parent. See section 6.1 for a description of extension Topics. This attribute
397 MUST NOT be used on Topics other than root Topics.

398 /wstop:Topic/MessagePattern

399 A QueryExpression. If it is present, this QueryExpression is used to describe the pattern of
400 the message that will appear on the Topic. Conceptually, the MessagePattern component
401 can be thought of as the object of a boolean() expression, evaluated against a Notification.
402 This boolean() expression, with the value of MessagePattern as parameter, is guaranteed to
403 evaluate to "true" when evaluated in the context of any Notification that is associated with the
404 Topic. The MessagePattern component constrains the Notification Messages that can be
405 used with the Topic. It is additional to the constraint contained in @messageTypes, and
406 provides a further refinement to that constraint.

407 /wstop:Topic/MessagePattern/@Dialect

408 A URI that identifies the language of the QueryExpression. WS-BaseNotification defines a
409 standard URI that identifies use of the XPath 1.0 language. Designers MAY define and use
410 other domain-specific URIs to identify the dialect of the QueryExpression.

411 /wstop:Topic/Topic

412 Declares a child Topic. A Topic can contain any number of child Topic elements; however the
413 value of the @name attribute of a child Topic must be unique amongst all the child Topics of
414 its immediate parent.

415 /wstop:Topic/{any}

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

417 /wstop:Topic/@{any}

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

419 **6.1 Extension Topics**

420 A NotificationProducer MAY support Topics that are marked as Extensions of other Topics by the
421 wstop:Topic/@parent attribute. Support for such Topics is OPTIONAL, a NotificationProducer
422 MAY choose not to support Topic Namespaces that contain Extension Topics.

423 If the @parent attribute is used, the following constraints MUST be obeyed by the designer of the
424 Topic Namespace:

- 425 1. The Topic containing the @parent attribute (the "Extension Topic") MUST be a root Topic
426 in its Topic Namespace
- 427 2. The Topic referenced by the @parent attribute (the "Parent Topic") MUST be from a
428 different Topic Namespace. It need not be a root Topic in that Namespace.
- 429 3. The Topic referenced by the @parent attribute can be an Extension Topic or the child of
430 an Extension Topic, however it MUST be possible to follow a chain of
431 Extension/parent/root Topics back to a root Topic that is not an Extension Topic.
432 Moreover a given Topic Namespace MUST NOT appear more than once in this chain.
433 This means that circular references, e.g. A extends B / B extends A are NOT permitted.
- 434 4. The Parent Topic MUST NOT be marked as final.

435 Although it appears as a root topic in its namespace, an Extension Topic, or its descendents, can
436 only be referenced using a path-based TopicExpression dialect in which the path passes through
437 the Parent Topic. In the case where the Parent itself is Extension Topic (or is descended from
438 one) this requirement applies recursively to the Parent Topic as well. Note that if the dialect
439 permits them, wild cards can be used in the TopicExpression to avoid having to include the
440 Parent Topic(s) explicitly in the path expression.
441

442 7 Modeling Topic Sets in XML

443 The WS-Topics XML Schema contains element and type definitions used to create Topic Set
444 documents. A Topic Set document gives an XML representation of the set of Topics supported by
445 a NotificationProducer. It has the wstop:TopicSet element as its document root, and contains zero
446 or more XML elements that represent the Topics in the Topic Set.

- 447 • If a Topic is defined as a root Topic of its Topic Namespace, and is not marked as an
448 Extension Topic, then it MUST appear as an immediate child of wstop:TopicSet. In
449 addition, if this Topic comes from any Namespace other than the ad-hoc Topic
450 Namespace described in section 10, then it MUST be represented by a namespace-
451 qualified element, with a Namespace name that is the targetNamespace of the Topic
452 Namespace.
- 453 • If a Topic is an Extension Topic, then it MUST NOT appear as an immediate child of
454 wstop:TopicSet, however it MUST be represented by a namespace-qualified element,
455 with a Namespace name that is the targetNamespace of the Topic Namespace.
- 456 • If a Topic is not a root Topic it MUST be represented by a non-qualified (NCName)
457 element, and MUST NOT appear as an immediate child of wstop:TopicSet.

458 Section 4 includes an example TopicSet showing both root and child Topics.

459 The following pseudo-schema gives a non-normative description of a TopicSet element:

```
460 <TopicSet>  
461   {any}*  
462 </TopicSet>
```

463 A TopicSet document is constrained in the following way:

464 /wstop: TopicSet

465 The top-level element in a Topic Set document. It contains a Topic element corresponding to
466 each supported Topic, along with OPTIONAL provider-specific additional elements. There
467 MUST NOT be a default XML namespace in scope for any of the descendents of TopicSet
468 (this ensures that all root Topics in the Topic Set can be identified by virtue of having QName
469 prefixes)

470 /wstop: TopicSet/{ any }

471 The TopicSet contains an element corresponding to each Topic that is included in the Topic
472 Set. The Topic name is used as the local part of the element name, and the element is
473 qualified with a Namespace if and only if it represents a root Topic from a Topic Namespace
474 other than the ad-hoc Topic Namespace. The position of the element in the document
475 hierarchy matches the position of the Topic in the Topic hierarchy. The TopicSet element can
476 contain additional elements that do not represent Topics in the Set – it MUST contain
477 additional, appropriately named elements where these are needed to ensure the correct
478 position in the hierarchy of the elements that do represent Topics in the Set. It MAY contain
479 additional elements that carry Producer-specific metadata.

480 /wstop: TopicSet/**/@topic

481 This is an attribute of type xsd:boolean, used to distinguish elements that represent Topics in
482 the set from those that do not. An element in the content of wstop:TopicSet MUST have a
483 wstop:@topic attribute with a value of "true" if and only if it represents a Topic in the Topic
484 Set.

485 /wstop:TopicSet/@{any}

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

487 If a Topic is defined as an Extension of another Topic then its Parent Topic MUST be represented
488 by an element in the TopicSet (although it need not have wstop:@topic="true"), and the element
489 representing the Extension Topic MUST be a child of the element representing the Parent Topic.
490 This means that all Extension Topics can be referenced using paths that include the root Topic
491 from the Parent Topic's Topic Namespace.

492 8 Topic Expression Dialects

493 Topics are referred to by TopicExpressions. There are several places in WS-Notification where
494 these expressions can appear:

- 495 ▪ As a component of the Subscribe message request to a NotificationProducer;
- 496 ▪ As a component of a Notification message sent to a NotificationConsumer or
497 NotificationBroker;
- 498 ▪ In the TopicExpression Resource Property element(s) associated with the
499 NotificationProducer role

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

```
501 <wsnt:TopicExpression Dialect= xsd:anyURI?>  
502   {any}?  
503 </wsnt:TopicExpression>
```

504 A TopicExpression has two components:

505 /wsnt:TopicExpression/@Dialect

506 The Dialect component contains a URI which identifies the type of grammar used in the
507 TopicExpression. This URI may be one from the set defined in this document, or may be a
508 URI defined elsewhere.

509 /wsnt:TopicExpression/{any}

510 The content of the TopicExpression is an expression in the grammar defined by the
511 expression language identified by the @Dialect component.

512 The purpose of a TopicExpression is to identify a set of one or more Topics. These Topics can
513 come from one or more Topic Namespaces.

514 This specification defines a number of Dialects that can be used to construct TopicExpressions.

515 These Dialects make use of Namespace prefixes as defined in [XML-Namespaces]. The
516 namespace declarations that specify the mapping of a prefix to an actual namespace URI can be
517 found amongst any namespace declaration in scope for the TopicExpression. Note: Some XML
518 processors might modify the namespace declarations. Designers should be aware that such
519 transforms exist and might render the expression incoherent; as it is likely the change in
520 namespace declaration will not update a QName embedded within a string.

521

522 8.1 Simple TopicExpression Dialect

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

```
524 http://docs.oasis-open.org/wsn/t-1/TopicExpression/Simple
```

525 This dialect is defined to standardize a very simple Topic Expression language for use by
526 resource constrained entities in the WS-Notification system that deal only with simple Topic
527 Namespaces. In this dialect the TopicExpression is simply the QName of a root Topic, consisting
wsn-ws_topics-1.3-spec-os 1 October 2006

528 of a namespace prefix that identifies the Topic Space, and a local name that identifies the root
529 Topic within that Topic Space.

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

```
533 [1] TopicExpression ::= RootTopic  
534 [2] RootTopic ::= QName  
535 [ vc: If a namespace prefix is included in the RootTopic, it must correspond to a valid  
536 Topic Namespace definition and the local name must correspond to the name of a root  
537 Topic defined in that namespace.]
```

538 Because the only valid TopicExpression in this dialect is a QName, only root Topics can be
539 addressed by this grammar. For those entities that support only this dialect of TopicExpression,
540 only simple Topic Namespaces (TopicNamespaces that only define root Topics) SHOULD be
541 used.

542 Although an Extension Topic is a root Topic in its own namespace, Extension Topics can not be
543 referenced using this dialect. An Extension Topic MUST only be referenced using a path than
544 includes its Parent Topic.

545 An example TopicExpression within this dialect is shown below:

```
546 ...  
547 xmlns:tns="http://example.org/topics/example1 "  
548 ...  
549  
550 <wsnt:TopicExpression  
551 Dialect="http://docs.oasis-open.org/wsn/t-1/TopicExpression/Simple">  
552 tns:t1  
553 </wsnt:TopicExpression>
```

554 This TopicExpression identifies the root Topic t1 within the Topic Namespace corresponding to
555 the namespace prefix tns:.

556 8.2 Concrete TopicExpression Dialect

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

```
558 http://docs.oasis-open.org/wsn/t-1/TopicExpression/Concrete
```

559 The Concrete TopicExpression is used to identify a single Topic within a Topic Namespace, using
560 a path notation. As it uses a path notation, it can identify any Topic within a Topic Namespace – it
561 is not limited to root Topics.

562

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

```
566 [3] TopicExpression ::= TopicPath  
567 [4] TopicPath ::= RootTopic ChildTopicExpression*
```

568 [5] RootTopic ::= QName
 569 [vc: If a namespace prefix is included in the RootTopic, it must correspond to a valid
 570 Topic Namespace definition and the local name must correspond to the name of a root
 571 Topic defined in that namespace.]
 572 [6] ChildTopicExpression ::= '/' ChildTopicName
 573 [7] ChildTopicName ::= QName | NCName
 574 [vc: The NCName or local part of the QName, must correspond to the name of a Topic
 575 within the descendant path from the RootTopic, where each forward slash denotes
 576 another level of child Topic elements in the path.]

577 Note: White space is not permitted within a Concrete TopicExpression.

578 An example TopicExpression within this dialect is shown below:

```
579 ...
580     xmlns:tns="http://example.org/topics/example1"
581 ...
582
583 <wsnt:TopicExpression
584     Dialect="http://docs.oasis-open.org/wsn/t-1/TopicExpression/Concrete">
585     tns:t1/t3
586 </wsnt:TopicExpression>
```

587 This TopicExpression identifies the Topic named "t3", child of Topic tns:t1.

588 As with XPath, this TopicExpression syntax uses the slash ("/") to describe *child of*.

589 This dialect allows namespace prefixes to be included in the path. Prefixes are used to switch
 590 between namespaces when passing from a parent Topic to an Extension Topic as shown in the
 591 following example:

```
592 ...
593     xmlns:tns1=http://example.org/topics/example1"
594     xmlns:tns2=http://example.org/topics/example2"
595 ...
596 <wsnt:TopicExpression
597     Dialect="http://docs.oasis-open.org/wsn/t-1/TopicExpression/Concrete">
598     tns1:t1/tns2:t3
599 </wsnt:TopicExpression>
```

600 This TopicExpression identifies the Topic named "t3" from Topic Namespace
 601 http://example.org/topics/example2", which was defined in that namespace as an extension of
 602 Topic t1 from Topic Namespace <http://example.org/topics/example1>".

603 An Extension Topic can only be referenced using a path than includes its Parent Topic in the
 604 manner just shown. In this example it would not be valid to attempt to refer to the topic by using
 605 the expression tns2:t3.

606 Namespace prefixes MUST only be used on root Topics (this includes Extension Topics since
 607 these are by definition root Topics).

608

609 Note: The Simple TopicExpression dialect defined in the previous section is a subset of the
 610 Concrete TopicExpression dialect.

611 8.3 Full TopicExpression Dialect

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

614 <http://docs.oasis-open.org/wsn/t-1/TopicExpression/Full>

615

616 This dialect allows TopicExpressions that identify more than one Topic (possibly from multiple
617 Topic Namespaces). It extends the Concrete TopicExpression dialect, in the sense that every
618 expression in the Concrete TopicExpression dialect is also valid in the Full TopicExpression
619 dialect, and has the same meaning.

620

621 Full TopicExpressions are XPath 1.0 [XPATH] relative location path expressions with some
622 additional syntactic constraints listed in this section. The XPath expression is evaluated over a
623 NotificationProducer's TopicSet document as defined in section 7. The TopicExpression identifies
624 the set of Topics that correspond to the elements in the node-set that results from evaluating the
625 location path contained in the TopicExpression, using standard XPath 1.0. The initial context
626 node for this evaluation is the wstop:TopicSet root element. Note that some of the elements
627 returned by the evaluation might not correspond to Topics (these are elements which do not have
628 @topic="true").

629 The Full TopicExpression dialect does not permit the use of the entire XPath language. This
630 specification provides syntactic constraints on the contents of the Full TopicExpression that limit
631 the constructs that can be used.

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

```
636 [1] TopicExpression ::= TopicPath | ConjoinedTopicExpression
637 [2] ConjoinedTopicExpression ::= TopicExpression Conjunction
638                               TopicExpression
639 [3] Conjunction ::= '['
640 [4] TopicPath ::= RootTopic ChildTopicExpression*
641 [5] RootTopic ::= NamespacePrefix? ('/')? (NCName | '*' )
642 [ vc: If a namespace prefix is included in the RootTopic, it must correspond to a valid
643 Topic Namespace definition and the local name must correspond to the name of a root
644 Topic defined in that namespace.]
645 [6] NamespacePrefix ::= NCName ':'
646 [7] ChildTopicExpression ::= '/' '/'? (ChildTopicName | '*' | '.')
647 [8] ChildTopicName ::= QName | NCName
648 [ vc: The NCName must correspond to the name of a topic within the descendant path
649 from the RootTopic, where each forward slash denotes another level of child Topic
650 elements in the path.]
```

651 As with the ConcreteTopicExpression, the ChildTopicName [8] MAY contain a namespace prefix
652 to allow an expression to include an extension Topic. Namespace prefixes MUST only be used
653 on root Topics (note that an extension Topic is by definition a root Topic).

654 Note: An Extension Topic is not permitted to appear as the as an immediate child of the
655 wstop:TopicSet element. This means that an Extension Topic can only be referenced using a
656 path than includes its Parent Topic (possibly wildcarded).

657 Note: White space is not permitted within a Full TopicExpression.

658 Note: The Concrete TopicExpression dialect defined in the previous section is a subset of the Full
659 TopicExpression dialect that contains no wildcards, '/' separators, or '|' operators.

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

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

```
664 "tns:t1/*"
```

665 This TopicExpression identifies all of the child Topics of the root Topic t1. Note that this
666 TopicExpression does not include the root Topic t1 itself, and it does not include any
667 grandchildren or further descendents of t1.

668 Wildcard characters can be interspersed with fixed child Topic names, to build up longer paths,
669 for example:

```
670 "tns:t1/*/t3"
```

671 This TopicExpression identifies all grandchildren of tns:t1 that have the name t3.

672 The wildcard * can also be used in place of a root Topic name, for example:

```
673 "tns:*"
```

674 This TopicExpression identifies all root Topics in the tns: Topic Namespace.

675 As in full XPath the // separator is used to identify all descendents (subject of course to the
676 constraints implied by the remainder of the path), not just immediate children.

677 If the TopicExpression ends with the characters "/*." this indicates that the TopicExpression
678 matches a Topic sub-tree. For example:

```
679 "tns:t1/t3/*."
```

680 This identifies the sub-tree consisting of tns:t1/t3 and all its descendents.

681 If the TopicExpression ends with the characters "/*" this indicates that the TopicExpression
682 matches all the descendents of a Topic. For example:

```
683 "tns:t1/t3/*"
```

684 This identifies the sub-tree consisting of the descendents of tns:t1/t3 but, unlike the previous
685 example, does not include tns:t1/t3 itself.

686 To include all the Topics in the entire Topic Namespace the following TopicExpression can be
687 used:

```
688 "tns://*"
```

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

690

```
"tns:t1//t3"
```

691 This TopicExpression identifies all descendents of tns:t1 that have the name t3.

692 A Full TopicExpression can contain two or more wildcards (both * and //).

693 Full TopicExpressions can be combined together with the conjunction operator as follows:

694

```
"tns:t1/t2|tns:t4/t5"
```

695 A Full TopicExpression using | can include root Topics from different Topic Namespaces. Note: a
696 Full TopicExpression containing a conjunction operator is equivalent to the set union of the
697 Topics described by combining the TopicExpression on either side of the conjunction operator.

698 8.4 XPath TopicExpression Dialect

699 This specification defines a fully conformant XPath 1.0 TopicExpression dialect with the following
700 URI:

701

```
http://www.w3.org/TR/1999/REC-xpath-19991116
```

702 This dialect allows TopicExpressions that identify more than one Topic (possibly from multiple
703 Topic Namespaces). It extends the Full TopicExpression dialect, in the sense that every
704 expression in the Full TopicExpression dialect is also valid in the XPath TopicExpression dialect,
705 and has the same meaning.

706 The XPath TopicExpression is evaluated over the NotificationProducer's TopicSet document in
707 the same way as the Full TopicExpression that is described section 8.3. The only difference
708 between the two dialects is that the XPath TopicExpression permits a richer set of selection
709 possibilities, since the full range of XPath 1.0 is available.

710 Any valid XPath expression is permitted, however if an expression does not return a node-set
711 containing elements that correspond to Topics then it does not identify any Topics. For example,
712 the following XPath expressions are valid XPath TopicExpressions, but none of them identify any
713 Topics, so including any of these as a Filter in a Subscribe request will result in no Notifications
714 being delivered to the NotificationConsumer:

- 715 • 123
- 716 • //@topic=true
- 717 • //@topic
- 718 • //*[@topic=false]

719 The first of these evaluates to a number and the second is a boolean. Neither of these are node-
720 sets, so neither identifies any Topics. The third of these evaluates to a node-set, but it is a node-
721 set that only contains attributes. The last one evaluates to a node-set that contains elements, but
722 it only selects the elements that do not correspond to Topics.

723

724 8.5 Validating TopicExpressions

725 If the NotificationProducer permits it, a TopicExpression MAY be used as a Filter in the Subscribe
726 message [WS-BaseNotification]. Such TopicExpressions might refer to one or more Topics which

727 might or might not exist in the Topic Namespace, or in the Topic Set supported by the
728 NotificationProducer.

729 The NotificationProducer MUST validate the TopicExpression as follows:

730 If the TopicExpression explicitly refers to a Topic that is not permitted by the Topic Namespace,
731 then the NotificationProducer MUST respond with a Fault. A Topic is not permitted if it is a root
732 Topic that is not defined in the Topic Namespace, and that Topic Namespace has @final="true",
733 or if it descends from a root Topic that is not defined in the Topic Namespace, and that Topic
734 Namespace has @final="true". A Topic is also not permitted if it, or any of its ancestors, are not
735 defined in the Topic Namespace and are the child of a Topic that is defined with @final='true'.

736 If the NotificationProducer has a fixed Topic Set, and the intersection of the Topics selected by
737 the TopicExpression with this Topic Set is empty, then the NotificationProducer MUST respond
738 with a Fault.

739 If the TopicExpression has a path that references a Topic Namespace that is not supported by
740 the NotificationProducer then the NotificationProducer MAY respond with a Fault, regardless of
741 whether the Topic Set is fixed or not

742 Here are some examples to illustrate these rules:

743 Suppose that Topic Namespace tns1 (with @final="true") contains root Topics tns1:A (@final=
744 "true") and tns1:B (@final = "false"), and that NotificationProducer (X) has a fixed Topic Set
745 consisting just of tns1:B.

- 746 ▪ Any subscribe with a TopicExpression containing tns1:D is rejected
- 747 ▪ Any subscribe with a TopicExpression containing tns1:A/X is rejected
- 748 ▪ A subscribe to tns1:B/X is rejected, but would be permitted if X did not have a fixed
749 Topic Set.
- 750 ▪ A subscribe to tns1:A is rejected, but would be permitted if X did not have a fixed Topic
751 Set.
- 752 ▪ A subscribe to tns1:* is permitted (and is equivalent in this case to a subscribe to
753 tns1:B)
- 754 ▪ A subscribe to tns1://* is permitted (and is equivalent in this case to a subscribe to
755 tns1:B)
- 756 ▪ A subscribe to tns1:A|tns1:B is permitted (and is equivalent in this case to a
757 subscribe to tns1:B)

758

9 Growing a Topic Tree

759 If a Topic in the Topic Namespace is marked with the 'final' attribute with value="true", then no
760 further child Topics can be added dynamically to that Topic.

761 If a Topic is not marked with the 'final' attribute with value="true", then a NotificationProducer
762 could potentially add further child Topics to that Topic within its Topic Set, and permit
763 Subscriptions to such child Topics. This specification does not define the circumstances under
764 which this occurs, and it is up to the NotificationProducer to determine if and when it permits
765 additional children (it is not obligated to allow children to be added just because a Topic has been
766 marked with final="false").

767 Similarly, if the TopicNamespace is not marked with the 'final' attribute with value="true", then a
768 NotificationProducer MAY add root Topics to its Topic Set that use that Topic Namespace's URI
769 but which were not defined in the TopicNamespace document.

770 When a NotificationProducer accepts Topics that are not previously defined in the Topic
771 Namespace, it adds them to its TopicSet document, but it is not obliged to update any actual
772 document that contains the Topic Namespace definition. Rather, the extension exists only for that
773 NotificationProducer and any Publisher or Subscriber that interacts with it. Circumstances under
774 which a NotificationProducer is permitted to add new child Topics to a Topic include:

- 775 ▪ A Subscriber attempting to subscribe using a TopicExpression that suggests one or more
776 new child Topics;
- 777 ▪ A Publisher attempting to publish using a TopicExpression that suggests a new child
778 Topic;
- 779 ▪ The NotificationProducer implementation encountering a new circumstance that doesn't
780 fit well with any of the existing child Topics (for example a new company starts trading on
781 a stock market, and a stock ticker service wishes to include it);
- 782 ▪ An administrator explicitly adding support for a new child Topic using some administrative
783 portType (not defined by any WS-Notification specification) implemented by the
784 NotificationProducer.

785 If a Notification Producer accepts a new Topic into its Topic Set, then messages produced on that
786 new Topic are eligible for selection by any wild-carded subscriptions that were in effect before the
787 Topic was added. The NotificationProducer MUST behave as if each subscription's
788 TopicExpression is re-evaluated against the Topic Set as each message is processed, although
789 implementers are free to choose any approach that produces this effect.

10 The “ad-hoc” Topic Namespace

790

791 Associating a Topic Namespace with an XML namespace provides an unambiguous naming
792 scheme for Topics. This is important when two entities which have no prior knowledge of each
793 other attempt (for example a Subscriber which has just discovered a NotificationBroker) to
794 interact.

795 However, there are circumstances where someone wishes to implement a Publisher for which
796 there is no suitable pre-existing Topic Namespace – and where the implementer does not wish to
797 incur the overhead of creating a new Topic Namespace (assigning a unique namespace, and
798 creating the TopicNamespace element within some XML instance document).

799 To help such users, WS-Notification defines a special built-in Topic Namespace called the *ad-hoc*
800 Topic Namespace.

801 The ad-hoc Topic Namespace has no pre-defined root Topics, but it is not final and so it allows
802 new root Topics to be added dynamically (in the same way that a non-final Topic allows new child
803 Topics to be added to it). Any Topic that is added dynamically to the ad-hoc Topic Namespace
804 itself permits the addition of further child Topics, and allows any type of Notification element to be
805 associated with it.

806 The ad-hoc Topic Namespace is indicated by omitting the namespace URI, i.e. a namespace of
807 "", and is accessed by using TopicExpressions which are unqualified.

808 A NotificationProducer or Subscriber can use this Topic Namespace to define *ad-hoc Topics*
809 dynamically, without having to associate them with their own Topic Namespace. Caution should
810 be used when employing ad-hoc Topics, as there is no way for a NotificationConsumer to
811 distinguish between them and other similarly-named ad-hoc Topics supported by any number of
812 NotificationProducers.

813

814 **11 NotificationProducers and Topics**

815 A NotificationProducer MAY use Topics to group Notifications related to some Situation (see
816 [WS-BaseNotification] for a definition of NotificationProducer, Notification and Situation). A
817 NotificationProducer can support zero or more Topics, and these can come from multiple Topic
818 Namespaces. A NotificationProducer can support an entire Topic Tree, or just a subset of the
819 Topics in a Topic Tree.

820 The NotificationProducer MAY support Resource Properties [WS-ResourceProperties] that
821 indicate the set of Topics that it expects to handle. WS-BaseNotification defines two resource
822 properties that can be used for this purpose.

- 823 1. The NotificationProducer MAY support the wstop:TopicSet resource property, which
824 returns the entire Topic Set as a single XML element as defined in section 7,
- 825 2. The NotificationProducer MAY support the wstop:TopicExpression resource property.
826 This resource property returns a list of TopicExpressions covering the set of supported
827 Topics.

828 The first approach has the advantage that the ResourceProperty returns the document used to
829 evaluate Topic subscription filters that use the Full or XPATH dialects. It allows the
830 NotificationProducer to insert producer-specific metadata that can be used in filters constructed
831 using the XPATH dialect.

832 The second approach is simpler in the case where the NotificationProducer only supports Simple
833 or Concrete Topic Expression dialects (it is merely the list of supported expressions). It could be
834 more concise in cases where NotificationProducers support Full or XPath Topic Expression
835 dialects since such a NotificationProducer could use a wildcarded TopicExpression to cover more
836 than one Topic.

837 A NotificationProducer is free to support either, both, or neither of these ResourceProperties.

838 This specification defines the following global attribute which MAY be included in the value
839 returned by a ResourceProperty query. It is RECOMMENDED that NotificationProducers include
840 this attribute in TopicExpression ResourceProperty values.

841 /@wstop:TopicNamespaceLocation

842 The location from which a TopicNamespace document can be retrieved

843

844 The set of Topics supported by the NotificationProducer MAY change over time. Reasons for the
845 set of Topics changing include:

- 846 ▪ The NotificationProducer supporting additional Topics from a Topic Namespace that is
847 already partially supported;
- 848 ▪ The NotificationProducer supporting additional Topics from a Topic Namespace not
849 previously supported;
- 850 ▪ The NotificationProducer supporting extension Topics to a (new or already supported)
851 Topic Namespace, as discussed in section 9;
- 852 ▪ The NotificationProducer ceasing to support Topics previously listed.

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

856 **12 Security Considerations**

857 Security considerations related to the use of Topics are discussed in [WS-BaseNotification] and in
858 [WS-BrokeredNotification]. It is recommended that implementations allow authorization policies
859 be specified at the granularity of the Topic.

860

861

13References

862

[RFC2119]

863

864 S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels", IETF
865 RFC 2119, March 1997, <http://www.ietf.org/rfc/rfc2119.txt>.

[WS-BaseNotification]

866

867 "Web Services Base Notification 1.3", OASIS Standard.
868 http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf

[WS-BrokeredNotification]

869

870 "Web Services Brokered Notification 1.3", OASIS Standard.
871 http://docs.oasis-open.org/wsn/wsn-ws_brokered_notification-1.3-spec-os.pdf

[WS-ResourceProperties]

872

873 "Web Services Resource Properties 1.2", OASIS Standard.
874 http://docs.oasis-open.org/wsr/wsrf-ws_resource_properties-1.2-spec-os.pdf

[XML]

875

876 "Extensible Markup Language (XML)", W3C Recommendation.
877 <http://www.w3.org/TR/REC-xml>

[XML-InfoSet]

878

879 "XML Information Set", W3C Recommendation.
880 <http://www.w3.org/TR/xml-infoset/>

[XML-Namespaces]

881

882 "Namespaces in XML 1.1", W3C Recommendation.
883 <http://www.w3.org/TR/xml-names11/>

[XPath]

884

885 "XML Path Language (XPath) Version 1.0", W3C Recommendation.
886 <http://www.w3.org/TR/xpath>

887 **Appendix A. Acknowledgments**

888 The following individuals were members of the committee during the development of this
889 specification:

890

891 Sid Askary, Fred Carter (AmberPoint), Martin Chapman (Oracle), Dave Chappell (Sonic
892 Software), Rick Cobb (KnowNow), Ugo Corda (SeeBeyond Technology Corporation), John Fuller,
893 Stephen Graham (IBM), David Hull (Tibco), Hideharu Kato (Hitachi), Lily Liu (webMethods, Inc.),
894 Tom Maguire (IBM), Susan Malaika (IBM), Samuel Meder (Argonne National Laboratory), Bryan
895 Murray (Hewlett-Packard), Peter Niblett (IBM), Sanjay Patil (SAP), Mark Peel (Novell), Matt
896 Roberts (IBM), Igor Sedukhin (Computer Associates), David Snelling (Fujitsu), Latha Srinivasan
897 (Hewlett-Packard), William Vambenepe (Hewlett-Packard) and Kirk Wilson (Computer
898 Associates).

Appendix B. XML Schema

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

```
901 <?xml version="1.0" encoding="UTF-8"?>
902 <!--
903
904 OASIS takes no position regarding the validity or scope of any
905 intellectual property or other rights that might be claimed to pertain
906 to the implementation or use of the technology described in this
907 document or the extent to which any license under such rights might or
908 might not be available; neither does it represent that it has made any
909 effort to identify any such rights. Information on OASIS's procedures
910 with respect to rights in OASIS specifications can be found at the
911 OASIS website. Copies of claims of rights made available for
912 publication and any assurances of licenses to be made available, or the
913 result of an attempt made to obtain a general license or permission for
914 the use of such proprietary rights by implementors or users of this
915 specification, can be obtained from the OASIS Executive Director.
916
917 OASIS invites any interested party to bring to its attention any
918 copyrights, patents or patent applications, or other proprietary rights
919 which may cover technology that may be required to implement this
920 specification. Please address the information to the OASIS Executive
921 Director.
922
923 Copyright (C) OASIS Open (2004-2006). All Rights Reserved.
924
925 This document and translations of it may be copied and furnished to
926 others, and derivative works that comment on or otherwise explain it or
927 assist in its implementation may be prepared, copied, published and
928 distributed, in whole or in part, without restriction of any kind,
929 provided that the above copyright notice and this paragraph are
930 included on all such copies and derivative works. However, this
931 document itself may not be modified in any way, such as by removing the
932 copyright notice or references to OASIS, except as needed for the
933 purpose of developing OASIS specifications, in which case the
934 procedures for copyrights defined in the OASIS Intellectual Property
935 Rights document must be followed, or as required to translate it into
936 languages other than English.
937
938 The limited permissions granted above are perpetual and will not be
939 revoked by OASIS or its successors or assigns.
940
941 This document and the information contained herein is provided on an
942 "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED,
943 INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE
944 INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED
945 WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.
946
947 -->
```

948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001

```
<xsd:schema
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:wstop = "http://docs.oasis-open.org/wsn/t-1"
  targetNamespace = "http://docs.oasis-open.org/wsn/t-1"
  elementFormDefault="qualified"  attributeFormDefault="unqualified">
<!-- ===== utility type definitions ===== -->
<xsd:complexType name="Documentation" mixed="true">
  <xsd:sequence>
    <xsd:any processContents="lax" minOccurs="0"
      maxOccurs="unbounded" namespace="##any" />
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="ExtensibleDocumented" abstract="true"
  mixed="false">
  <xsd:sequence>
    <xsd:element name="documentation" type="wstop:Documentation"
      minOccurs="0" />
  </xsd:sequence>
  <xsd:anyAttribute namespace="##other" processContents="lax" />
</xsd:complexType>

<xsd:complexType name="QueryExpressionType" mixed="true">
  <xsd:sequence>
    <xsd:any minOccurs="0" maxOccurs="1" processContents="lax" />
  </xsd:sequence>
  <xsd:attribute name="Dialect" type="xsd:anyURI" use="required"/>
</xsd:complexType>

<!-- ===== Topic-Namespace Related ===== -->
<xsd:complexType name="TopicNamespaceType">
  <xsd:complexContent>
    <xsd:extension base="wstop:ExtensibleDocumented">
      <xsd:sequence>
        <xsd:element name="Topic"
          minOccurs="0" maxOccurs="unbounded">
          <xsd:complexType>
            <xsd:complexContent>
              <xsd:extension base="wstop:TopicType">
                <xsd:attribute name="parent"
          type="wstop:ConcreteTopicExpression" />
              </xsd:extension>
            </xsd:complexContent>
          </xsd:complexType>
        </xsd:element>
        <xsd:any namespace="##other"
          minOccurs="0" maxOccurs="unbounded"
          processContents="lax" />
      </xsd:sequence>
      <xsd:attribute name="name" type="xsd:NCName" />
      <xsd:attribute name="targetNamespace" type="xsd:anyURI "
```

```

1002         use="required" />
1003         <xsd:attribute name="final" type="xsd:boolean"
1004             default="false" />
1005     </xsd:extension>
1006 </xsd:complexContent>
1007 </xsd:complexType>
1008
1009 <xsd:element name="TopicNamespace" type="wstop:TopicNamespaceType">
1010     <xsd:unique name="rootTopicUniqueness">
1011         <xsd:selector xpath="wstop:Topic" />
1012         <xsd:field xpath="@name" />
1013     </xsd:unique>
1014 </xsd:element>
1015
1016 <xsd:attribute name="topicNamespaceLocation" type="xsd:anyURI" />
1017
1018
1019
1020 <!-- ===== Topic Related ===== -->
1021
1022 <xsd:complexType name="TopicType">
1023     <xsd:complexContent>
1024         <xsd:extension base="wstop:ExtensibleDocumented">
1025             <xsd:sequence>
1026                 <xsd:element name="MessagePattern"
1027                     type="wstop:QueryExpressionType"
1028                     minOccurs="0" maxOccurs="1" />
1029                 <xsd:element name="Topic" type="wstop:TopicType"
1030                     minOccurs="0" maxOccurs="unbounded">
1031                     <xsd:unique name="childTopicUniqueness">
1032                         <xsd:selector xpath="wstop:topic" />
1033                         <xsd:field xpath="@name" />
1034                     </xsd:unique>
1035                 </xsd:element>
1036                 <xsd:any namespace="##other" minOccurs="0"
1037                     maxOccurs="unbounded" />
1038             </xsd:sequence>
1039             <xsd:attribute name="name" use="required" type="xsd:NCName" />
1040             <xsd:attribute name="messageTypes">
1041                 <xsd:simpleType>
1042                     <xsd:list itemType="xsd:QName" />
1043                 </xsd:simpleType>
1044             </xsd:attribute>
1045             <xsd:attribute name="final" type="xsd:boolean"
1046                 default="false" />
1047         </xsd:extension>
1048     </xsd:complexContent>
1049 </xsd:complexType>
1050
1051 <!-- ===== Topic Set Related ===== -->
1052
1053 <xsd:complexType name="TopicSetType">
1054     <xsd:complexContent>
1055         <xsd:extension base="wstop:ExtensibleDocumented">

```

```

1056     <xsd:sequence>
1057         <xsd:any namespace="##other"
1058             minOccurs="0" maxOccurs="unbounded"
1059             processContents="lax"/>
1060     </xsd:sequence>
1061 </xsd:extension>
1062 </xsd:complexContent>
1063 </xsd:complexType>
1064
1065 <xsd:element name="TopicSet" type="wstop:TopicSetType"/>
1066 <xsd:attribute name="topic" type="xsd:boolean" default="false"/>
1067
1068 <!-- ===== Topic Expression Related ===== -->
1069
1070 <xsd:simpleType name="FullTopicExpression">
1071     <xsd:restriction base="xsd:token">
1072         <xsd:annotation>
1073             <xsd:documentation>
1074                 TopicPathExpression ::= TopicPath ( '|' TopicPath ) *
1075                 TopicPath          ::= RootTopic ChildTopicExpression *
1076                 RootTopic           ::= NamespacePrefix? ( '/' )? ( NCName | '*' )
1077                 NamespacePrefix ::= NCName ':'
1078                 ChildTopicExpression ::= '/' '/'? ( QName | NCName | '*' | '.' )
1079
1080             </xsd:documentation>
1081         </xsd:annotation>
1082         <xsd:pattern value=
1083             "([\i-[:]][\c-[:]]*)?(//)?([\i-[:]][\c-[:]]*\|\\*)(/|//)(([\i-
1084 [[:]][\c-[:]]*)?[\i-[:]][\c-[:]]*\|\\*|.)))*(\|([\i-[:]][\c-
1085 [[:]]*)?(/)?([\i-[:]][\c-[:]]*\|\\*)(/|//)(([\i-[:]][\c-[:]]*)?[\i-
1086 [[:]][\c-[:]]*\|\\*|.)))*">
1087         </xsd:pattern>
1088     </xsd:restriction>
1089 </xsd:simpleType>
1090
1091 <xsd:simpleType name="ConcreteTopicExpression">
1092     <xsd:restriction base="xsd:token">
1093         <xsd:annotation>
1094             <xsd:documentation>
1095                 The pattern allows strings matching the following EBNF:
1096                 ConcreteTopicPath ::= RootTopic ChildTopic*
1097                 RootTopic          ::= QName
1098                 ChildTopic         ::= '/' ( QName | NCName )
1099
1100             </xsd:documentation>
1101         </xsd:annotation>
1102         <xsd:pattern value=
1103             "(([\i-[:]][\c-[:]]*)?[\i-[:]][\c-[:]]*)(/([\i-[:]][\c-[:]]*)?[\i-
1104 [[:]][\c-[:]]*)*" >
1105         </xsd:pattern>
1106     </xsd:restriction>
1107 </xsd:simpleType>
1108
1109 <xsd:simpleType name="SimpleTopicExpression">

```

```
1110     <xsd:restriction base="xsd:QName">
1111         <xsd:annotation>
1112             <xsd:documentation>
1113 The pattern allows strings matching the following EBNF:
1114     RootTopic          ::=   QName
1115
1116             </xsd:documentation>
1117         </xsd:annotation>
1118     </xsd:restriction>
1119 </xsd:simpleType>
1120
1121 </xsd:schema>
```

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.
b	2005-06-27	Sid Askary	<ul style="list-style-type: none"> - Added the Section on security - Added the section on faults - Added the concepts from white paper -Corrected typos -Removed references to White Paper - NotificationMessage w/ Notification - Updated status section - Replaced Notional Conventions <p>TODO:</p> <ul style="list-style-type: none"> - AI 85 - Rewrite of Chapter 5. - Incorporate new Namespace in Schema
c	2005-07-06	Peter Niblett	<p>Updated to use new Namespaces</p> <p>Removed aliases (WSN 4.5)</p> <p>TopicSpace changed to Topic Namespace (WSN 4.2)</p> <p>Added section describing Topic Set document and made corresponding adjustments to the schema and to the definition of FullTopicSet (WSN 4.2)</p> <p>Added an XPath 1.0 Topic Expression Dialect (WSN 4.3)</p> <p>Use wsnt:QueryExpressionType instead of wsrp:QueryExpressionType (WSN</p>

Rev	Date	By Whom	What
			<p>4.26)</p> <p>Updated the references</p> <p>New acknowledgements section</p> <p>Changed SimpleTopicExpression to be xsd:QName instead of xsd:token with a pattern (WSN 4.20)</p> <p>Removed the “special” @messageTypes value of xsd:any, and removed the default value for this attribute from the XML Schema (WSN 4.27)</p> <p>Added “final” attribute to TopicNamespace (WSN 4.22)</p> <p>Renamed the adhoc namespace to “” (WSN 4.9)</p> <p>Added sentence on wildcard resolution with growing topic sets (WSN 4.16)</p> <p>Added global TopicNamespaceLocation attribute (WSN4.21)</p>
d	2005-09-26	Peter Niblett	<p>Corrections to some of the amendments in c, following issue resolution review</p> <p>Term Topic Path changed to become Topic Expression (AI 85)</p>
e	2005-11-24	Peter Niblett	<p>Domain-specific extensions to TopicNamespaces (WSN 4.4)</p> <p>Updated references to and namespace URIs for other WSN specifications (AI 138)</p> <p>Removed reference to WSDL 2.0 (AI 136)</p> <p>Removed section 1.4 (Fault Definitions) as it is not relevant to this specification</p> <p>Replaced section 12 (Security Considerations) with pointers to [WS BaseNotification] and [WS BrokeredNotification], since the material contained was duplicative and not all relevant to this specification</p> <p>Added discussion of TopicSet and</p>

Rev	Date	By Whom	What
			<p>TopicExpression RPs (WSN 4.28)</p> <p>Miscellaneous other corrections (WSN 4.28)</p> <p>Discussion of Namespace prefix binding in TopicExpressions (WSN 4.23 and WSN 4.24)</p> <p>Added description of TopicNamespaceLocation attribute (WSN 4.21)</p> <p>Widened scope of 8.5 to cover all TopicExpressions, not just Full and XPath,</p>
f	2005-12-03	Peter Niblett	Revised the resolution of issue 4.26 to avoid circular dependency of schemas (QueryExpressionType is now defined in this schema).
g	2005-12-06	Peter Niblett	<p>Corrected the namespace and description of TopicSpaceLocation attribute (WSN 4.21)</p> <p>Corrected schemaLocations in the TopicNamespace and TopicSet examples (AI 138)</p> <p>Reworded the definition of wstop:Topic/@parent, and reworded bullet 3 of 6.1 (WSN 4.4)</p> <p>Revised words at the start of section 7, to make them clearer (WSN 4.2)</p>
wd-02a	2006-03-31	Peter Niblett	Miscellaneous errata
wd-02b	2006-05-22	Peter Niblett	WSN 4.29. Specified that an Extension Topic (or child of an Extension Topic) can only be referenced by using path expressions that include the parent of the Extension Topic. If the dialect permits them, wild card characters can be used so that the Parent Topic name does not need to be included explicitly.

1123

1124

Appendix D. Notices

1125 OASIS takes no position regarding the validity or scope of any intellectual property or other rights
1126 that might be claimed to pertain to the implementation or use of the technology described in this
1127 document or the extent to which any license under such rights might or might not be available;
1128 neither does it represent that it has made any effort to identify any such rights. Information on
1129 OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS
1130 website. Copies of claims of rights made available for publication and any assurances of licenses
1131 to be made available, or the result of an attempt made to obtain a general license or permission
1132 for the use of such proprietary rights by implementors or users of this specification, can be
1133 obtained from the OASIS Executive Director.

1134

1135 OASIS invites any interested party to bring to its attention any copyrights, patents or patent
1136 applications, or other proprietary rights which may cover technology that may be required to
1137 implement this specification. Please address the information to the OASIS Executive Director.

1138

1139 Copyright (C) OASIS Open (2004-2006). All Rights Reserved.

1140

1141 This document and translations of it may be copied and furnished to others, and derivative works
1142 that comment on or otherwise explain it or assist in its implementation may be prepared, copied,
1143 published and distributed, in whole or in part, without restriction of any kind, provided that the
1144 above copyright notice and this paragraph are included on all such copies and derivative works.
1145 However, this document itself may not be modified in any way, such as by removing the copyright
1146 notice or references to OASIS, except as needed for the purpose of developing OASIS
1147 specifications, in which case the procedures for copyrights defined in the OASIS Intellectual
1148 Property Rights document must be followed, or as required to translate it into languages other
1149 than English.

1150

1151 The limited permissions granted above are perpetual and will not be revoked by OASIS or its
1152 successors or assigns.

1153

1154 This document and the information contained herein is provided on an "AS IS" basis and OASIS
1155 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO
1156 ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE
1157 ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A
1158 PARTICULAR PURPOSE.