



Web Services Topics 1.3 (WS-Topics) Committee Specification, 31 July 2006

Document identifier:

wsn-ws_topics-1.3-spec-cs-01

Location:

http://docs.oasis-open.org/wsn/wsn-ws_topics-1.3-spec-cs-01.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 published by the OASIS WS-Notification Technical Committee as a
38 "Committee Specification".

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

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

51	1	Introduction	4
52	1.1	Goals and Requirements	4
53	1.1.1	Requirements	4
54	1.1.2	Non-Goals	4
55	1.2	Notational Conventions	5
56	1.3	Namespaces	6
57	2	Terminology and Concepts	7
58	3	Topics and Topic Namespaces	8
59	4	Example	10
60	5	Modeling Topic Namespaces in XML	12
61	6	Modeling Topics in XML	13
62	6.1	Extension Topics	14
63	7	Modeling Topic Sets in XML	16
64	8	Topic Expression Dialects	18
65	8.1	Simple TopicExpression Dialect	18
66	8.2	Concrete TopicExpression Dialect	19
67	8.3	Full TopicExpression Dialect	21
68	8.4	XPath TopicExpression Dialect	23
69	8.5	Validating TopicExpressions	23
70	9	Growing a Topic Tree	25
71	10	The “ad-hoc” Topic Namespace	26
72	11	NotificationProducers and Topics	27
73	12	Security Considerations	29
74	13	References	30
75		Appendix A. Acknowledgments	31
76		Appendix B. XML Schema	32
77		Appendix C. Revision History	37
78		Appendix D. Notices	40
79			

80 **1 Introduction**

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

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

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

92 **1.1 Goals and Requirements**

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

97 **1.1.1 Requirements**

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

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

111

112 **1.1.2 Non-Goals**

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

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

117 1.2 Notational Conventions

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

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

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

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

139

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

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

154 **1.3 Namespaces**

155 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

156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181

2 Terminology and Concepts

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

Topic:

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

Topic Namespace:

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

Topic Tree:

- A hierarchical grouping of Topics.

Topic Set:

- The collection of Topics supported by a NotificationProducer

182

3 Topics and Topic Namespaces

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

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

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

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

207 The rationale for hierarchical topic structures is:

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

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

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

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

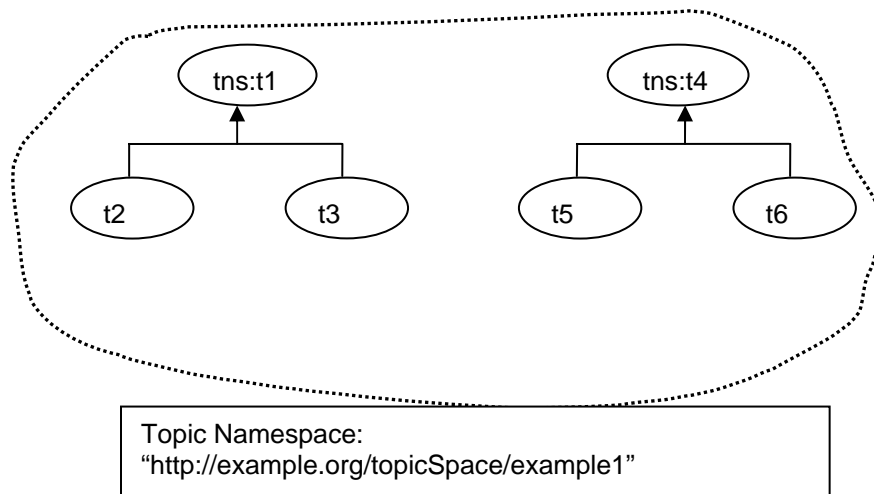
230 No Topic can contain two immediate child Topics with the same name, however Topics with the
231 same name can appear elsewhere in a Topic Tree, and no relationship is implied. Similarly two
232 separate Topic Trees in the same Topic Namespace can contain Topics with the same name;
233 these are not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

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

239 4 Example

240 | Consider a Topic Namespace that can be depicted as illustrated by [Figure 1](#). The Topic
241 Namespace is contained in the "http://example.org/topicSpace/example1" namespace. This
242 Topic Namespace has two root Topics, named t1 and t4. Topic t1 has two child Topics, t2 and t3.
243 Topic t4 has two child Topics, t5 and t6.

244
245
246
247
248
249
250
251
252
253
254
255
256
257



258 *Figure 4: Example Topic Namespace*

259
260

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

261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276

```
<?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>
```

277
278

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

279

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

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

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

288

289
290
291
292
293
294
295
296
297
298
299
300
301
302
303

```
<?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>
```

304

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

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

319

320 We describe the details behind modeling Topic Namespaces and Topics in the following sections.
wsn-ws_topics-1.3-spec-cs-01 7/31/2006

321 5 Modeling Topic Namespaces in XML

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

326

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

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

332 A TopicNamespace element is constrained in the following way:

333 /wstop:TopicNamespace

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

336 /wstop:TopicNamespace/@name

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

339 /wstop:TopicNamespace/@targetNameSpace

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

342 /wstop:TopicNamespace/@final

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

347 /wstop:TopicNamespace/Topic

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

352 /wstop:TopicNamespace/{any}

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

354 /wstop:TopicNamespace/@{any}

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

356 6 Modeling Topics in XML

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

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

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

367 /wstop:Topic

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

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

373 /wstop:Topic/@name

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

380 /wstop:Topic/@messageTypes

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

387 /wstop:Topic/@final

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

394 /wstop:Topic/@parent

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

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

399 /wstop:Topic/MessagePattern

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

408 /wstop:Topic/MessagePattern/@Dialect

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

412 /wstop:Topic/Topic

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

416 /wstop:Topic/{any}

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

418 /wstop:Topic/@{any}

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

420 6.1 Extension Topics

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

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

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

436 Although it appears as a root topic in its namespace, an Extension Topic, or its descendents, can
437 only be referenced using a path-based TopicExpression dialect in which the path passes through
438 the Parent Topic. In the case where the Parent itself is Extension Topic (or is descended from
439 one) this requirement applies recursively to the Parent Topic as well. Note that if the dialect
440 permits them, wild cards can be used in the TopicExpression to avoid having to include the
441 Parent Topic(s) explicitly in the path expression.
442

443 7 Modeling Topic Sets in XML

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

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

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

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

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

464 A TopicSet document is constrained in the following way:

465 /wstop: TopicSet

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

471 /wstop: TopicSet/{ any }

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

481 /wstop: TopicSet/**/@topic

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

486 /wstop:TopicSet/@{any}

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

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

493 8 Topic Expression Dialects

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

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

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

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

505 A TopicExpression has two components:

506 /wsnt:TopicExpression/@Dialect

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

510 /wsnt:TopicExpression/{any}

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

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

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

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

522

523 8.1 Simple TopicExpression Dialect

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

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

526 This dialect is defined to standardize a very simple Topic Expression language for use by
527 resource constrained entities in the WS-Notification system that deal only with simple Topic
528 Namespaces. In this dialect the TopicExpression is simply the QName of a root Topic, consisting
wsn-ws_topics-1.3-spec-cs-01 7/31/2006

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

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

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

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

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

546 An example TopicExpression within this dialect is shown below:

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

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

557 **8.2 Concrete TopicExpression Dialect**

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

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

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

563

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

```
567 [3] TopicExpression ::= TopicPath  
568 [4] TopicPath ::= RootTopic ChildTopicExpression*
```

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

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

579 An example TopicExpression within this dialect is shown below:

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

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

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

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

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

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

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

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

609

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

612 8.3 Full TopicExpression Dialect

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

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

616

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

621

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

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

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

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

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

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

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

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

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

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

```
665 "tns:t1/*"
```

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

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

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

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

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

```
674 "tns:*"
```

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

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

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

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

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

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

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

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

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

```
689 "tns://*"
```

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

691

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

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

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

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

695

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

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

699 8.4 XPath TopicExpression Dialect

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

702

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

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

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

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

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

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

724

725 8.5 Validating TopicExpressions

726 If the NotificationProducer permits it, a TopicExpression MAY be used as a Filter in the Subscribe
727 message [WS-BaseNotification]. Such TopicExpressions might refer to one or more Topics which
wsn-ws_topics-1.3-spec-cs-01

7/31/2006

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

730 The NotificationProducer MUST validate the TopicExpression as follows:

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

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

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

743 Here are some examples to illustrate these rules:

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

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

759

9 Growing a Topic Tree

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

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

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

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

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

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

791 10The “ad-hoc” Topic Namespace

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

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

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

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

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

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

814

815 **11 NotificationProducers and Topics**

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

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

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

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

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

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

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

842 /@wstop:TopicNamespaceLocation

843 The location from which a TopicNamespace document can be retrieved

844

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

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

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

857

12 Security Considerations

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

861

862

13 References

863

864 **[RFC2119]**

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

867 **[WS-BaseNotification]**

868 "Web Services Base Notification 1.3".
869 http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-cs-01.pdf

870 **[WS-BrokeredNotification]**

871 "Web Services Brokered Notification 1.3".
872 http://docs.oasis-open.org/wsn/wsn-ws_brokered_notification-1.3-spec-cs-01.pdf

873 **[WS-ResourceProperties]**

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

876 **[XML]**

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

879 **[XML-InfoSet]**

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

882 **[XML-Namespaces]**

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

885 **[XPath]**

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

888 **Appendix A. Acknowledgments**

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

891

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

900

Appendix B. XML Schema

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

```
902 <?xml version="1.0" encoding="UTF-8"?>
903 <!--
904
905 OASIS takes no position regarding the validity or scope of any
906 intellectual property or other rights that might be claimed to pertain
907 to the implementation or use of the technology described in this
908 document or the extent to which any license under such rights might or
909 might not be available; neither does it represent that it has made any
910 effort to identify any such rights. Information on OASIS's procedures
911 with respect to rights in OASIS specifications can be found at the
912 OASIS website. Copies of claims of rights made available for
913 publication and any assurances of licenses to be made available, or the
914 result of an attempt made to obtain a general license or permission for
915 the use of such proprietary rights by implementors or users of this
916 specification, can be obtained from the OASIS Executive Director.
917
918 OASIS invites any interested party to bring to its attention any
919 copyrights, patents or patent applications, or other proprietary rights
920 which may cover technology that may be required to implement this
921 specification. Please address the information to the OASIS Executive
922 Director.
923
924 Copyright (C) OASIS Open (2004-2006). All Rights Reserved.
925
926 This document and translations of it may be copied and furnished to
927 others, and derivative works that comment on or otherwise explain it or
928 assist in its implementation may be prepared, copied, published and
929 distributed, in whole or in part, without restriction of any kind,
930 provided that the above copyright notice and this paragraph are
931 included on all such copies and derivative works. However, this
932 document itself may not be modified in any way, such as by removing the
933 copyright notice or references to OASIS, except as needed for the
934 purpose of developing OASIS specifications, in which case the
935 procedures for copyrights defined in the OASIS Intellectual Property
936 Rights document must be followed, or as required to translate it into
937 languages other than English.
938
939 The limited permissions granted above are perpetual and will not be
940 revoked by OASIS or its successors or assigns.
941
942 This document and the information contained herein is provided on an
943 "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED,
944 INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE
945 INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED
946 WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.
947
948 -->
```

wsn-ws_topics-1.3-spec-cs-01

7/31/2006

Copyright © OASIS Open 2004-2006. All Rights Reserved.

Page 32 of 40


```

949
950
951 <xsd:schema
952   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
953   xmlns:wstop = "http://docs.oasis-open.org/wsn/t-1"
954   targetNamespace = "http://docs.oasis-open.org/wsn/t-1"
955   elementFormDefault="qualified" attributeFormDefault="unqualified">
956
957 <!-- ===== utility type definitions ===== -->
958 <xsd:complexType name="Documentation" mixed="true">
959   <xsd:sequence>
960     <xsd:any processContents="lax" minOccurs="0"
961       maxOccurs="unbounded" namespace="##any" />
962   </xsd:sequence>
963 </xsd:complexType>
964
965 <xsd:complexType name="ExtensibleDocumented" abstract="true"
966   mixed="false">
967   <xsd:sequence>
968     <xsd:element name="documentation" type="wstop:Documentation"
969       minOccurs="0" />
970   </xsd:sequence>
971   <xsd:anyAttribute namespace="##other" processContents="lax" />
972 </xsd:complexType>
973
974 <xsd:complexType name="QueryExpressionType" mixed="true">
975   <xsd:sequence>
976     <xsd:any minOccurs="0" maxOccurs="1" processContents="lax" />
977   </xsd:sequence>
978   <xsd:attribute name="Dialect" type="xsd:anyURI" use="required"/>
979 </xsd:complexType>
980
981 <!-- ===== Topic-Namespace Related ===== -->
982 <xsd:complexType name="TopicNamespaceType">
983   <xsd:complexContent>
984     <xsd:extension base="wstop:ExtensibleDocumented">
985       <xsd:sequence>
986         <xsd:element name="Topic"
987           minOccurs="0" maxOccurs="unbounded">
988           <xsd:complexType>
989             <xsd:complexContent>
990               <xsd:extension base="wstop:TopicType">
991                 <xsd:attribute name="parent"
992 type="wstop:ConcreteTopicExpression" />
993               </xsd:extension>
994             </xsd:complexContent>
995           </xsd:complexType>
996         </xsd:element>
997         <xsd:any namespace="##other"
998           minOccurs="0" maxOccurs="unbounded"
999           processContents="lax" />
1000       </xsd:sequence>
1001       <xsd:attribute name="name" type="xsd:NCName" />

```

```

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

```

```

1055     <xsd:complexContent>
1056         <xsd:extension base="wstop:ExtensibleDocumented">
1057             <xsd:sequence>
1058                 <xsd:any namespace="##other"
1059                     minOccurs="0" maxOccurs="unbounded"
1060                     processContents="lax"/>
1061             </xsd:sequence>
1062         </xsd:extension>
1063     </xsd:complexContent>
1064 </xsd:complexType>
1065
1066 <xsd:element name="TopicSet" type="wstop:TopicSetType"/>
1067 <xsd:attribute name="topic" type="xsd:boolean" default="false"/>
1068
1069 <!-- ===== Topic Expression Related ===== -->
1070
1071 <xsd:simpleType name="FullTopicExpression">
1072     <xsd:restriction base="xsd:token">
1073         <xsd:annotation>
1074             <xsd:documentation>
1075                 TopicPathExpression ::= TopicPath ( '|' TopicPath )*
1076                 TopicPath          ::= RootTopic ChildTopicExpression*
1077                 RootTopic          ::= NamespacePrefix? ('/'?)? (NCName | '*')
1078                 NamespacePrefix ::= NCName ':'
1079                 ChildTopicExpression ::= '/' '/'? (QName | NCName | '*' | '.')
1080
1081             </xsd:documentation>
1082         </xsd:annotation>
1083         <xsd:pattern value=
1084             "([\i-[:]][\c-[:]]*:?)(//)?([\i-[:]][\c-[:]]*|\*)(/|//)(([\i-
1085 [:]][\c-[:]]*:?)[\i-[:]][\c-[:]]*|\*|[\.]))*\|([\i-[:]][\c-
1086 [:]]*:?)(//)?([\i-[:]][\c-[:]]*|\*)(/|//)(([\i-[:]][\c-[:]]*:?)[\i-
1087 [:]][\c-[:]]*|\*|[\.])*)*"
1088         </xsd:pattern>
1089     </xsd:restriction>
1090 </xsd:simpleType>
1091
1092 <xsd:simpleType name="ConcreteTopicExpression">
1093     <xsd:restriction base="xsd:token">
1094         <xsd:annotation>
1095             <xsd:documentation>
1096                 The pattern allows strings matching the following EBNF:
1097                 ConcreteTopicPath ::= RootTopic ChildTopic*
1098                 RootTopic          ::= QName
1099                 ChildTopic         ::= '/' (QName | NCName)
1100
1101             </xsd:documentation>
1102         </xsd:annotation>
1103         <xsd:pattern value=
1104             "(([\i-[:]][\c-[:]]*:?)[\i-[:]][\c-[:]]*)/(([\i-[:]][\c-[:]]*:?)[\i-
1105 [:]][\c-[:]]*)*" >
1106         </xsd:pattern>
1107     </xsd:restriction>

```

1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122

```
</xsd:simpleType>  
<xsd:simpleType name="SimpleTopicExpression">  
  <xsd:restriction base="xsd:QName">  
    <xsd:annotation>  
      <xsd:documentation>  
The pattern allows strings matching the following EBNF:  
  RootTopic      ::=   QName  
      </xsd:documentation>  
    </xsd:annotation>  
  </xsd:restriction>  
</xsd:simpleType>  
</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 wsrf-rp: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.

1124

1125

Appendix D. Notices

1126 OASIS takes no position regarding the validity or scope of any intellectual property or other rights
1127 that might be claimed to pertain to the implementation or use of the technology described in this
1128 document or the extent to which any license under such rights might or might not be available;
1129 neither does it represent that it has made any effort to identify any such rights. Information on
1130 OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS
1131 website. Copies of claims of rights made available for publication and any assurances of licenses
1132 to be made available, or the result of an attempt made to obtain a general license or permission
1133 for the use of such proprietary rights by implementors or users of this specification, can be
1134 obtained from the OASIS Executive Director.

1135

1136 OASIS invites any interested party to bring to its attention any copyrights, patents or patent
1137 applications, or other proprietary rights which may cover technology that may be required to
1138 implement this specification. Please address the information to the OASIS Executive Director.

1139

1140 Copyright (C) OASIS Open (2004-2006). All Rights Reserved.

1141

1142 This document and translations of it may be copied and furnished to others, and derivative works
1143 that comment on or otherwise explain it or assist in its implementation may be prepared, copied,
1144 published and distributed, in whole or in part, without restriction of any kind, provided that the
1145 above copyright notice and this paragraph are included on all such copies and derivative works.
1146 However, this document itself may not be modified in any way, such as by removing the copyright
1147 notice or references to OASIS, except as needed for the purpose of developing OASIS
1148 specifications, in which case the procedures for copyrights defined in the OASIS Intellectual
1149 Property Rights document must be followed, or as required to translate it into languages other
1150 than English.

1151

1152 The limited permissions granted above are perpetual and will not be revoked by OASIS or its
1153 successors or assigns.

1154

1155 This document and the information contained herein is provided on an "AS IS" basis and OASIS
1156 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO
1157 ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE
1158 ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A
1159 PARTICULAR PURPOSE.