



Web Services Brokered Notification 1.3 (WS-BrokeredNotification)

Public Review Draft 03, 31 May 2006

Document identifier:

wsn-ws_brokered_notification-1.3-spec-pr-03

Location:

http://docs.oasis-open.org/wsn/wsn-ws_brokered_notification-1.3-spec-pr-03.pdf

Editors:

Dave Chappell, Sonic Software <chappell@sonicsoftware.com>

Lily Liu, webMethods <lily.liu@webmethods.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](#)].

29 This document defines the Web services interface for the NotificationBroker. A
30 NotificationBroker is an intermediary that, among other things, allows publication of
31 messages from entities that are not themselves service providers. It includes standard
32 message exchanges to be implemented by NotificationBroker service providers along
33 with operational requirements expected of service providers and requestors that
34 participate in brokered notifications. This work relies upon WS-BaseNotification.

35 **Status:**

36 On May 31st, 2006, the OASIS WS-Notification Technical Committee approved this
37 document for publication as a Public Review Draft. Committee members should send
38 comments on this specification to the wsn@lists.oasis-open.org list. Others may submit
39 comments to the TC via the web form found on the TC's web page at [http://www.oasis-
41 open.org/committees/wsn](http://www.oasis-
40 open.org/committees/wsn). Click the button for "Send A Comment" at the top of the page.
42 Submitted comments (for this work as well as other works of the TC) are publicly archived
43 and can be viewed at <http://lists.oasis-open.org/archives/wsn-comment/>.

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

47 [The errata document for this specification is maintained at:](http://docs.oasis-open.org/wsn/wsn-ws_brokered_notification-1.3-errata.pdf)
48 http://docs.oasis-open.org/wsn/wsn-ws_brokered_notification-1.3-errata.pdf

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.....	5
55	1.2	Notational Conventions	5
56	1.3	Namespaces	7
57	1.4	Fault Definitions.....	7
58	2	Relationship to Other Specifications.....	8
59	3	Terminology and Concepts.....	9
60	4	Publishing.....	12
61	5	NotificationBroker Interface.....	15
62	5.1	NotificationBroker Resource Properties	16
63	5.2	Notify	16
64	5.3	Subscribe	16
65	5.4	GetCurrentMessage	17
66	5.5	RegisterPublisher	17
67	5.6	CreatePullPoint	17
68	6	RegisterPublisher Interface.....	18
69	6.1	RegisterPublisher	18
70	6.1.1	Example SOAP Encoding of the RegisterPublisher Message Exchange	22
71	7	PublisherRegistrationManager Interface	24
72	7.1	PublisherRegistration Resource Properties	24
73	7.2	DestroyRegistration.....	25
74	7.2.1	Example SOAP Encoding of the DestroyRegistration Message Exchange	26
75	8	Security Considerations.....	27
76	8.1	Securing PublisherRegistration.....	27
77	9	References.....	28
78	9.1	Normative	28
79	9.2	Non-Normative	28
80		Appendix A. Acknowledgments.....	30
81		Appendix B. XML Schema.....	31
82		Appendix C. WSDL 1.1.....	36
83		Appendix D. Revision History.....	41
84		Appendix E. Notices	43

85 1 Introduction

86 The Event-driven, or Notification-based, interaction pattern is a commonly used pattern for inter-
87 object communications. Examples exist in many domains, for example, in publish/subscribe
88 systems or in system and device management domains. Message brokers are involved in many
89 of these systems, such as the ones provided by Message Oriented Middleware vendors.

90 This specification defines the Web services interface for the NotificationBroker. A
91 NotificationBroker is an intermediary between message Publishers and message Subscribers. A
92 NotificationBroker decouples NotificationProducers and Notification Consumers and can provide
93 advanced messaging features such as demand-based publishing and load-balancing. A
94 NotificationBroker also allows publication of messages from entities that are not themselves
95 service providers. This is very similar to a traditional Message Oriented Middleware model.

96 The NotificationBroker interface includes standard message exchanges to be implemented by
97 NotificationBroker service providers along with operational requirements expected of service
98 providers and requestors that participate in brokered notifications.

Deleted: Common functions of Publishers and Subscribers, such as messaging dissemination and security measurements, can be implemented at the NotificationBroker to produce lightweight Producers and Consumers.

99 1.1 Goals and Requirements

100 The goal of WS-BrokeredNotification is to standardize message exchanges involved in Web
101 services publish and subscribe of a message broker. The overall [requirements](#) of WS-Notification
102 are presented in [\[WS-BaseNotification\]](#). The following section lists the specific subset of those
103 objectives realized by WS-BrokeredNotification.

Deleted: objectives

104 1.1.1 Requirements

105 In meeting this goal, the WS-BrokeredNotification specification must explicitly address the
106 following requirements:

- 107 • **Must allow for a notification broker as an intermediary.** A NotificationBroker is an
108 intermediary Web service that decouples NotificationConsumers from Publishers. A
109 notification broker can relieve a Publisher from having to implement message exchanges
110 associated with NotificationProducer; the NotificationBroker takes on the duties of
111 subscription management and distributing Notifications on behalf of the Publisher. It
112 implements NotificationProducer interface. It may implement SubscriptionManager or may
113 delegate the subscription management work to another component.
- 114 • **Must allow for federation of brokers.** It must be possible to build configurations with
115 multiple intermediary broker services in a dynamic fashion. This specification must allow for
116 a variety of broker topology usage patterns. Among other things, these allow for greater
117 scalability and permit sharing of administrative workload.
- 118 • **Must provide runtime metadata:** There must be a mechanism that lets a potential
119 Subscriber discover what elements available for a subscription are provided by a
120 NotificationBroker, and in what formats the subscription for a notification can be made.

- 121 • **Must conform to WS-BaseNotification:** A NotificationBroker must support required
122 message exchanges defined by the [WS-BaseNotification] specification. It must conform to
123 the NotificationProducer and the NotificationConsumer interfaces defined in WS-
124 BaseNotification.
- 125 • **WS-BrokeredNotification must be independent of binding-level details:** Transport
126 protocol details must be orthogonal to the subscription and the delivery of the notifications, so
127 that the specification can be used over a variety of different transports.
- 128 • **Must not exclude non-service producers and subscribers:** WS-BrokeredNotification
129 design must not exclude a non-service entity to deliver a notification message to a
130 NotificationBroker. It must not exclude a NotificationBroker to send a notification message to
131 a non-service consumer.
- 132 • **Must provide publisher registration:** WS-BrokeredNotification must define standard
133 message exchanges for registering a NotificationPublisher with a NotificationBroker.

Deleted: , so that the specification can be used over a variety of different transports.

134 1.1.2 Non-Goals

135 The following topics are outside the scope of the WS-BrokeredNotification specification:

- 136 • **Defining the format of notification payloads:** The data carried in Notification payloads is
137 application-domain specific, and WS-BrokeredNotification does not prescribe any particular
138 format for this data.
- 139 • **Defining any Events or Notifications:** The WS-BrokeredNotification specification does not
140 define any "standard" or "built-in" notification situations, events, or messages.
- 141 • **Defining the means by which NotificationBrokers are discovered by subscribers:** It is
142 beyond the scope of this specification to define the mechanisms for runtime discovery of
143 NotificationBrokers.

144 1.2 Notational Conventions

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

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

Deleted:

151 This specification uses a notational convention, referred to as "Pseudo-schemas" in a fashion
152 similar to the WSDL 2.0 Part 1 specification. A Pseudo-schema uses a BNF-style convention to
153 describe attributes and elements:

- 154 • '?' denotes optionality (i.e. zero or one occurrences),
155 • '*' denotes zero or more occurrences,
156 • '+' one or more occurrences,

- 157 • '[' and `]' are used to form groups,
- 158 • `|' represents choice.
- 159 • Attributes are conventionally assigned a value which corresponds to their type, as
- 160 defined in the normative schema.
- 161 • Elements with simple content are conventionally assigned a value which corresponds to
- 162 the type of their content, as defined in the normative schema.
- 163 • The use of {any} indicates the presence of an element wildcard (<xs:any/>).
- 164 • The use of @{any} indicates the presence of an attribute wildcard (<xs:anyAttribute/>).

165

```
166 <!-- sample pseudo-schema -->
167 <element
168     required_attribute_of_type_QName="xs:QName "
169     optional_attribute_of_type_string="xs:string"?>
170   <required_element />
171   <optional_element /> ?
172   <one_or_more_of_these_elements /> +
173   [ <choice_1 /> | <choice_2 /> ] *
174 </element>
```

175 Where there is disagreement between the separate XML schema and WSDL files describing the
176 messages defined by this specification and the normative descriptive text (excluding any pseudo-
177 schema) in this document, the normative descriptive text will take precedence over the separate
178 files. The separate files take precedence over any pseudo-schema and over any schema and
179 WSDL included in the appendices.

180 **1.3 Namespaces**

181 The following namespaces are used in this document:

Prefix	Namespace
s	http://schemas.xmlsoap.org/soap/envelope/ OR http://www.w3.org/2003/05/soap-envelope
xsd	http://www.w3.org/2001/XMLSchema
wsa	http://www.w3.org/2005/08/addressing
wsn-b	http://docs.oasis-open.org/wsn/b-2
wsn-br	http://docs.oasis-open.org/wsn/br-2
wsn-bw	http://docs.oasis-open.org/wsn/bw-2
wsn-brw	http://docs.oasis-open.org/wsn/brw-2
wsrf-bf	http://docs.oasis-open.org/wsr/bf-2
wsrf-bfw	http://docs.oasis-open.org/wsr/bfw-2

182 **1.4 Fault Definitions**

183 All faults generated by a NotificationBroker, RegisterPublisher, or PublisherRegistrationManager
184 SHOULD be compliant with the WS-BaseFaults [WS-BaseFaults] specification.

185 All faults defined by this specification MUST use the following URI for the WS-Addressing [action]
186 Message Addressing Property:

187 <http://docs.oasis-open.org/wsn/fault>.

188

2 Relationship to Other Specifications

189 | This specification builds on the basic notification mechanism defined in [\[WS-BaseNotification\]](#) by
190 adding the concept of an intermediary NotificationBroker, and describing additional variants on
191 the publisher role. A NotificationBroker takes on the role of both NotificationProducer and
192 NotificationConsumer (as defined in [\[WS-BaseNotification\]](#)), and its interactions with other
193 NotificationProducers and NotificationConsumers are largely defined by the WS-BaseNotification
194 specification.

195 This means that a NotificationBroker, implemented to conform to this specification, must also
196 conform to [\[WS-BaseNotification\]](#). Such a NotificationBroker can deliver notifications to
197 NotificationConsumers that are implemented to conform to [\[WS-BaseNotification\]](#), and can
198 subscribe to Notifications distributed by NotificationProducers as defined in [\[WS-
199 BaseNotification\]](#).

200 A NotificationBroker may support hierarchical topics as defined in [\[WS-Topics\]](#). By supporting
201 topics, NotificationBroker can manage enterprise messaging systems more efficiently.

202 WS-BrokeredNotification must be composable with other Web services specifications.

Deleted: ,

203

3 Terminology and Concepts

204 In addition to the terminology and usage described in the WS-BaseNotification specification, the
205 following are the terms defined in this specification:

206 Publisher:

- 207 • A Publisher is an entity that creates Notifications, based upon Situation(s) that it is
208 capable of detecting and translating into Notification artifacts. It does not need to be a
209 Web service.
- 210 • A Publisher can register what topics it wishes to publish with a NotificationBroker.
- 211 • A Publisher MAY be a Web service that implements the message exchanges associated
212 with the NotificationProducer interface, in which case it also distributes the Notifications
213 to the relevant NotificationConsumers.
- 214 • If a Publisher does not implement the message exchanges associated with
215 NotificationProducer, then it is not required to support the Subscribe request message
216 and does not have to maintain knowledge of the NotificationConsumers that are
217 subscribed to it; a NotificationBroker takes care of this on its behalf.

218 NotificationBroker:

- 219 • A NotificationBroker is an intermediary Web service that decouples
220 NotificationConsumers from Publishers. A NotificationBroker is capable of subscribing to
221 notifications, either on behalf of NotificationConsumers, or for the purpose of messaging
222 management. It is capable disseminating notifications on behalf of Publishers to
223 NotificationConsumers.
- 224 • A NotificationBroker aggregates NotificationProducer, NotificationConsumer, and
225 RegisterPublisher interfaces.
- 226 • Acting as an intermediary, a NotificationBroker provides additional capabilities to the
227 base NotificationProducer interface:
 - 228 ○ It can relieve a Publisher from having to implement message exchanges
229 associated with NotificationProducer; the NotificationBroker takes on the duties of
230 a SubscriptionManager (managing subscriptions) and NotificationProducer
231 (distributing Notifications) on behalf of the Publisher.
 - 232 ○ It can reduce the number of inter-service connections and references, if there are
233 many Publishers and many NotificationConsumers.
 - 234 ○ It can act as a finder service. Potential Publishers and Subscribers can in effect
235 find each other by utilizing a common NotificationBroker.
 - 236 ○ It can provide anonymous Notification, so that the Publishers and the
237 NotificationConsumers need not be aware of each other's identity.

238 • An implementation of a NotificationBroker may provide additional added-value function
239 that is beyond the scope of this specification, for example, logging Notifications, or
240 transforming Topics and/or Notification content. Additional function provided by a
241 NotificationBroker can apply to all Publishers that utilize it.

242 • It may be the factory for Subscription resources or it may delegate the subscription
243 factory to another component.

244 • A NotificationBroker provides publisher registration functions.

245 • A NotificationBroker may [bridge between WS-Notification and other publish/subscribe](#)
246 [systems.](#)

Deleted: subscribe and disseminate messages that are not WS-Notification conforming.

247 PublisherRegistration:

248 • PublisherRegistration is a resource. A PublisherRegistration represents the relationship
249 between a Publisher and a NotificationBroker, in particular, which topic(s) the publisher is
250 permitted to publish to.

251 • A PublisherRegistration resource is created when a Publisher sends the
252 RegisterPublisher request message to a NotificationBroker and the NotificationBroker
253 succeeds in processing the registration.

254 • PublisherRegistration resources can be manipulated by messages sent to a
255 PublisherRegistrationManager Web service.

256 RegisterPublisher:

257 • A RegisterPublisher is a Web service that implements the message exchanges
258 associated with the RegisterPublisher interface. A PublisherRegistration resource is
259 created as a result of a RegisterPublisher request to a NotificationBroker.

260 PublisherRegistrationManager:

261 • A PublisherRegistrationManager is a Web service that implements the message
262 exchanges associated with the PublisherRegistrationManager interface.

263 • A [PublisherRegistration](#) resource can be manipulated through
264 PublisherRegistrationManager message exchanges.

Deleted: p

Deleted: r

265 • A PublisherRegistrationManager provides services that allow a service requestor to query
266 and manipulate PublisherRegistration resources that it manages.

267 • A PublisherRegistrationManager is subordinate to the NotificationBroker, and MAY be
268 implemented by the NotificationBroker service provider. However WS-
269 BrokeredNotification permits it to be implemented by a separate service provider, should
270 an implementer so desire.

271 Demand-Based Publishing:

272 • Some Publishers may be interested in knowing whether they have any Subscribers or
273 not, since producing a Notification may be a costly process. Such Publishers can register
274 with the NotificationBroker as a Demand-Based Publisher.

- 275
276
- Demand-Based Publishers implement message exchanges associated with the NotificationProducer interface.
- 277
278
279 |
280
- The NotificationBroker subscribes to the Demand-Based Publisher. When the NotificationBroker knows that there are no Subscribers for the Notifications from a Demand-Based Publisher, it pauses its Subscription with that Publisher; when it knows that there are some Subscribers, it resumes the Subscription.
- 281
282
283
284
285
- This way the Demand-Based Publisher does not need to produce messages when there are no Subscribers, however a Demand-Based Publisher is only required to support a single Subscriber on any given Topic, and so can delegate the management of multiple Subscribers, the delivery to multiple NotificationConsumers, and other related issues (for example security) to the NotificationBroker.

4 Publishing

286

287 There are three distinct stages in the Notification process

- 288
- Observation of the Situation and its noteworthy characteristics;
 - 289 • Creation of the Notification artifact that captures the noteworthy characteristics of the
290 Situation; and
 - 291 • Distribution of copies of the Notification to zero or more interested parties.

292 Stages 1 and 2 happen largely outside of the scope of the WS-Notification architecture; this
293 specification does not restrict the means by which these stages must occur. We refer to an entity
294 that performs stages 1 and 2 as a Publisher,

295 However, the WS-Notification family of specifications does specify how dissemination of
296 messages SHOULD occur. There are two dominant patterns by which Notifications are
297 disseminated in WS-Notification: direct and brokered.

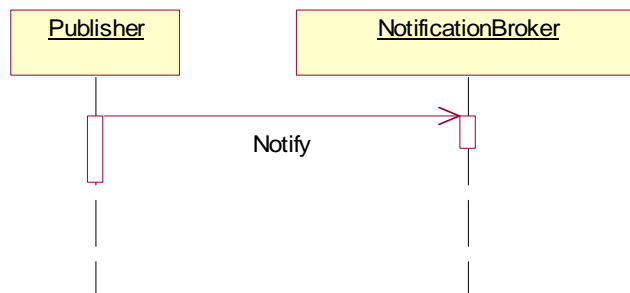
298 In the direct case, the publishing Web service implements message exchanges associated with
299 the NotificationProducer interface; it is responsible for accepting Subscribe messages and
300 sending Notifications to interested parties. The implementer of this Web service can choose to
301 program this behavior or delegate to specialized implementations of the Subscribe and
302 Notification delivery behavior. This case is addressed by the WS-BaseNotification specification
303 [[WS-BaseNotification](#)].

304 In the brokered case, an intermediary - a NotificationBroker - is responsible for disseminating
305 messages produced by one or more Publishers to zero or more NotificationConsumers.

306 There are three patterns associated with the relationship between the Publisher and the
307 NotificationBroker: simple publishing, broker-initiated publishing, and demand-based publishing.

Deleted:

308 The following figure illustrates simple publishing:



309

310 In the simple publishing scenario, the Publisher entity is responsible only for the core Publisher
311 functions - observing the Situation and formatting the Notification artifact that describes the

312 Situation. The dissemination step occurs when the Publisher sends the Notify message to the
313 NotificationBroker.

314 In the broker-initiated publishing pattern, the role of the Publisher is played by a Web service that
315 implements NotificationProducer. The act of observing the Situation and formatting the
316 Notification happens within the implementation logic of the NotificationProducer itself. The
317 Notification is disseminated by the NotificationProducer sending the Notify message to a
318 NotificationBroker. The Notification may also be disseminated by sending the Notify message to
319 any NotificationConsumers that are subscribing to the NotificationProducer.

Deleted:

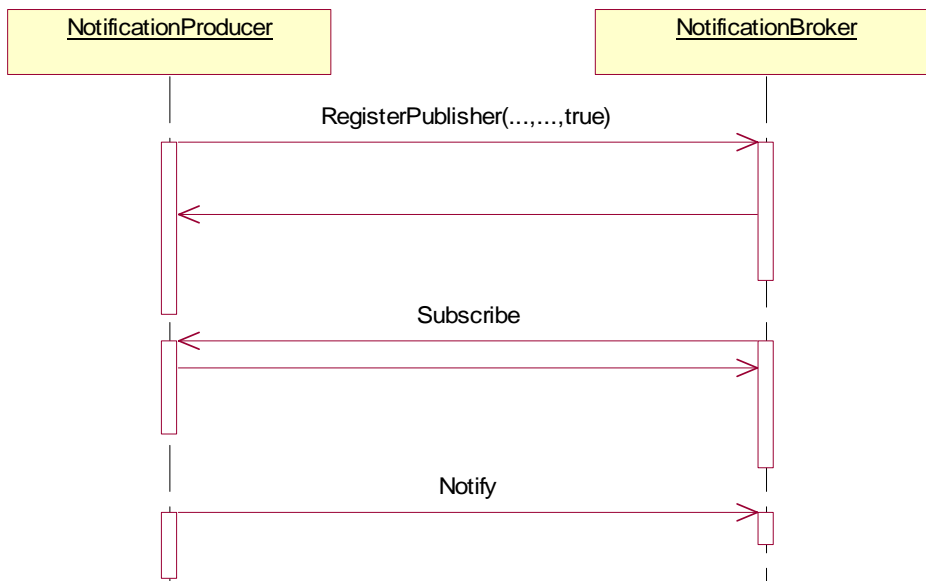
320 Note: in either of the above two cases, the NotificationBroker MAY require the Publisher to
321 register with it prior to sending the Notify message. For example, if the broker wishes to control
322 who can publish to a given Topic, it can perform an access control check during this registration.
323 However a NotificationBroker MAY allow Publishers to publish without pre-registration, if it so
324 chooses.

Deleted: choose to

325 The last pattern, the demand-based pattern, requires the Publisher to be a NotificationProducer,
326 and thereby accept the Subscribe message. Demand-based publication is intended for use in
327 cases where the act of observing the Situation or the act of formatting the Notification artifact
328 might be expensive to perform, and therefore should be avoided if there are no interested parties
329 for that Notification. [A Publisher indicates its intention to use this pattern by registering with the
330 NotificationProducer and setting the Demand component of the RegisterPublisher request
331 message to "true".](#) Based upon this style of registration, the NotificationBroker sends the
332 Subscribe message to the Publisher (recall: in this situation the Publisher must implement the
333 message exchanges associated with the NotificationProducer interface).

Deleted: . To use this pattern, the Publisher must register with the NotificationBroker, using the registration to express the intent to provide demand-based publishing only.

334 Furthermore, the NotificationBroker is expected to pause its Subscription whenever it has no



335 active Subscribers for the information provided by the Publisher. When the NotificationBroker
336 does have active Subscribers, it is obliged to resume its Subscription to the Publisher.

337

5 NotificationBroker Interface

338 The NotificationBroker interface defines a standard set of message exchanges to describe a
339 message broker, providing an intermediary between Publishers and Subscribers on a collection
340 of Topics, similar to a traditional Message Oriented Middleware model.

341 | [A](#) NotificationBroker MAY be a WS-Resource, and if it is, it MUST support the required message
342 exchanges defined by the [\[WS-ResourceProperties\]](#) specification and MAY support the optional
343 message exchanges defined by WS-ResourceProperties.

344 | A NotificationBroker MUST also support message exchanges and [MAY support](#) Resource
345 Property elements defined by the following interfaces:

- 346 • NotificationProducer
- 347 • NotificationConsumer
- 348 • RegisterPublisher

349 The NotificationBroker portType aggregates the three portTypes and is not the only way to
350 implement a broker. A distributed broker implementation can be achieved by hosting
351 NotificationProducer, NotificationConsumer, or RegisterPublisher portTypes at one or more
352 physical endpoints.

353 | The NotificationBroker [is not required to provide](#) any [specific](#) subscription durability or continuity.
354 NotificationBrokers SHOULD advertise their durability or reliability features, either through
355 policies or other means.

Deleted: does not specify

356 | NotificationBrokers MAY offer flow control and MAY implement Pull-Style notifications. If [they do](#)
357 so, NotificationBrokers SHOULD advertise these features, either through policies or other means.

358

359 5.1 NotificationBroker Resource Properties

360 In addition to the message exchanges described in this specification, a NotificationBroker MAY
361 also support the required message exchanges defined in the WS-ResourceProperties
362 specification and MAY support the optional message exchanges defined in the WS-
363 ResourceProperties specification. [In such cases](#), the Resource Properties document defined by
364 the NotificationBroker MUST include references to resource properties defined in
365 NotificationProducer and NotificationConsumer, and also MUST include a reference to the
366 following resource property element:

```
367 ...  
368     targetNamespace="http://docs.oasis-open.org/wsn/br-2">  
369 ...  
370     <xsd:element name="RequiresRegistration" type="xsd:boolean"/>  
371 ...
```

372 Furthermore, this reference MUST reflect the minOccurs and maxOccurs properties as follows:

```
373 <xsd:element ref="wsn-br:RequiresRegistration"  
374     minOccurs="1" maxOccurs="1" />
```

375 This resource property element is further constrained as follows:

376 /wsn-br:RequiresRegistration

377 The value is "true" if the NotificationBroker requires a publisher to register (see 6.1)
378 before sending it a Notify (i.e. publish) message on a Topic.

Deleted: If it does so

Deleted: The default is "false".

379 5.2 Notify

380 The NotificationBroker MUST support the Notify message exchange from the
381 NotificationConsumer interface [[WS-BaseNotification](#)], with the following clarifications/restrictions:

382 A Publisher sends a Notify message to a NotificationBroker in order to publish a Notification on a
383 given Topic. As a result of the Publisher sending this message, Notifications are delivered to all
384 NotificationConsumers subscribed on the given Topic. [The NotificationBroker may require that a
385 Publisher be registered before the Publisher sends it a Notification](#) (see 6.1).

Deleted: For some Topics (those that require a Publisher to pre-register), the sender must be a registered Publisher in order to successfully publish a Notification on the given Topic

386 5.3 Subscribe

387 [A NotificationBroker is capable of routing or producing a sequence of zero or more Notifications.](#)
388 [A Subscriber can register the interest of a NotificationConsumer to receive a subset of this](#)
389 [sequence. A Subscriber sends a Subscribe message to a NotificationBroker in order to register](#)
390 [this interest.](#)

391 The NotificationBroker MUST support the Subscribe message exchange from the
392 NotificationProducer interface [[WS-BaseNotification](#)]. A NotificationBroker MAY support any
393 TopicExpression dialect.

394 If the processing of a Subscribe message is successful, the NotificationBroker MUST produce a
395 response message, as described in [WS-BaseNotification](#), containing an endpoint reference to a
396 Subscription resource representing a Subscription created as a result of processing the

Deleted: A NotificationBroker is capable of routing or producing a sequence of zero or more Notifications. A Subscriber can register the interest of a NotificationConsumer to receive a subset of this sequence. A Subscriber sends a Subscribe message to a NotificationBroker in order to register this interest. ¶

397 Subscribe request. Otherwise, the NotificationBroker must fault. WS-BaseNotification defines a
398 set of these faults.

399 **5.4 GetCurrentMessage**

400 The NotificationBroker MUST support the GetCurrentMessage message exchange from the
401 NotificationProducer interface [[WS-BaseNotification](#)].

402 As defined in WS-BaseNotification, in response to a GetCurrentMessage message, the
403 NotificationBroker MAY return the last Notification published on a given Topic. This is a non-
404 destructive read, allowing a newly-subscribed NotificationConsumer to get the last Notification
405 that other NotificationConsumers have received.

406 **5.5 RegisterPublisher**

407 The NotificationBroker MUST support the RegisterPublisher message exchange from the
408 RegisterPublisher interface.

409 A Publisher can register its interest to publish messages through the NotificationBroker by
410 sending a RegisterPublisherRequest. The NotificationBroker is responsible for managing the
411 registration, and sending a RegisterPublisherResponse to the Publisher if the registration process
412 succeeds. Otherwise, the NotificationBroker MUST fault. These message exchanges are further
413 specified in the following Section 6.

414 **5.6 CreatePullPoint**

415 The NotificationBroker MAY support pull-style notification as defined in WS-BaseNotification and
416 attempt to create a PullPoint resource upon receiving a CreatePullPoint request. The
417 NotificationBroker does not define additional constraints to its usage of the CreatePullPoint
418 operation.

419 6 RegisterPublisher Interface

420 The RegisterPublisher interface contains message exchanges for publisher registration.
421 NotificationBroker implements the RegisterPublisher interface and is responsible for publisher
422 registration. A NotificationBroker may reject processing certain publisher registrations for reasons
423 such as lacking of authorization.

424 6.1 RegisterPublisher

425 The RegisterPublisher message is used by the Publisher to confirm its ability to publish on a
426 given Topic or set of Topics. If an entity wishes to register a publisher, it must send a
427 RegisterPublisher request message to the NotificationBroker. The format of the RegisterPublisher
428 request message is:

Deleted: MUST

```
429 ...  
430 <wsn-br:RegisterPublisher>  
431   <wsn-br:PublisherReference>  
432     wsa:EndpointReferenceType  
433   </wsn-br:PublisherReference?>  
434   <wsn-br:Topic Dialect = "xsd:anyURI">  
435     {any} ?  
436   </wsn-br:Topic> *  
437   <wsn-br:Demand>  
438     xsd:boolean  
439   </wsn-br:Demand?>  
440   <wsn-br:InitialTerminationTime>  
441     xsd:dateTime  
442   </wsn-br:InitialTerminationTime?>  
443   {any} *  
444 </wsn-br:RegisterPublisher>  
445 ...
```

446
447 The [WS-Addressing](#) [action] Message Addressing Property MUST contain the URI
448 <http://docs.oasis-open.org/wsn/brw-2/RegisterPublisher/RegisterPublisherRequest>.

449
450 The components of the RegisterPublisher request message are further described as follows:
451 /wsn-br:RegisterPublisher/PublisherReference

452 | An endpoint reference element from WS-Addressing [[WS-Addressing](#)], used to identify
453 | an entity that wishes to become a Publisher. This component **MUST** appear if the /wsn-
454 | br:RegisterPublisher/Demand component has value "true". If this component is missing,
455 | the Publisher is either not a Web service, or does not wish to receive messages from the
456 | NotificationBroker. Deleted: OPTIONAL

457 | /wsn-br:RegisterPublisher/Topic

458 | A set of TopicExpressions that identifies one or more Topics. If included, the given
459 | Publisher is registered to publish only on the set of Topics identified by this component. If
460 | this is missing the Publisher is registered to publish on any Topic supported by the
461 | NotificationBroker.

462 | /wsn-br:RegisterPublisher/Demand

463 | A Boolean element with the default value "false". If its value is "true", then the intent of the
464 | Publisher is to use a demand-based model from the NotificationBroker (see Section 4). In
465 | this case, the NotificationBroker **MUST** observe the rules associated with demand-based Deleted: must
466 | publishing, including establishing a Subscription with the Publisher on those Topics and
467 | pausing/resuming those Subscriptions as the NotificationBroker receives Subscriptions
468 | for those Topics.

469 | /wsn-br:RegisterPublisher/InitialTerminationTime

470 | This component contains the service requestor's suggestion for the initial termination
471 | time of the PublisherRegistration resource being created. This time is relative to the time
472 | source used by the NotificationBroker. If the NotificationBroker is unable or unwilling to
473 | set the TerminationTime to the given value or greater, the RegisterPublisher request
474 | **MUST** return an UnacceptableInitialTerminationTimeFault message. If the value is not "in
475 | the future" relative to the current time as known by the NotificationBroker, the
476 | RegisterPublisher request **MUST** also return an UnacceptableInitialTerminationTimeFault
477 | message.

478 | The use of the xsi:nil attribute with value "true" indicates there is no scheduled
479 | termination time requested for the RegisterPublisher. If the element does not include the
480 | time zone designation, the value of the element **MUST** be interpreted as universal time
481 | (UTC).

482 | The publisher should take care when choosing a value for InitialTerminationTime, and
483 | any subsequent values that modify the TerminationTime property of the publisher
484 | registration. It is **RECOMMENDED** that the publisher choose termination time values that
485 | are significantly (several [orders of magnitude](#)) greater than the network latency expected
486 | in the interaction between the publisher and the broker. In so doing, the designer avoids
487 | undesirable results, such as the termination time having expired prior to the receipt of the
488 | published message. The [[WS-ResourceLifetime](#)] specification (Section 5.1 Regarding
489 | time) contains further suggestions on how designers should reason about time values in
490 | a WS-Resource Lifetime application.

491 | If this component is not included, the initial value of the TerminationTime resource
492 | property is dependent on the implementation of the NotificationBroker.

493 | /wsn-br:RegisterPublisher/{any}

494 The RegisterPublisher request message allows for open content, in order to
495 accommodate elements that may be needed by extensions built on WS-
496 BrokeredNotification.

497 If a /wsn-br:RegisterPublisher/Topic component is included in the message, the
498 NotificationBroker MUST register the Web service specified by the /wsn-
499 br:RegisterPublisher/PublisherReference component as a Publisher on the set of Topics
500 identified by the /wsn-br:RegisterPublisher/Topic component. If for any reason the registration
501 fails, the NotificationBroker MUST fault.

502 As part of the processing of a RegisterPublisher request, the NotificationBroker creates a
503 PublisherRegistration resource representing the registration. A new resource is created
504 regardless of whether the same Publisher has previously registered with the NotificationBroker.
505 The NotificationBroker MUST return a PublisherRegistrationReference in the response to the
506 RegisterPublisher request.

Deleted: and may return a
ConsumerReference

507 PublisherRegistrationReference is a WS-Addressing endpoint reference and includes the address
508 of a PublisherRegistrationManager service.

Deleted: .

509 ConsumerReference is a WS-Addressing endpoint reference to a NotificationConsumer that
510 accepts notifications from this registered Publisher. If Demand value is false in the
511 RegisterPublisher request, the NotificationBroker MUST include a ConsumerReference in the
512 response.

Deleted: subscribes

Deleted: to

Deleted: published by

513 If the NotificationBroker accepts the RegisterPublisher request message, it must respond with a
514 message of the following form:

```
515 ...  
516 <wsn-br:RegisterPublisherResponse>  
517   <wsn-br:PublisherRegistrationReference>  
518     <wsa:Address>  
519       Address of PublisherRegistration Manager  
520     </wsa:Address>  
521     ...  
522   </wsn-br:PublisherRegistrationReference>  
523   <wsn-br:ConsumerReference>  
524     <wsa:Address>  
525       Address of a NotificationConsumer with which the  
526       Publisher is registered  
527     </wsa:Address>  
528     ...  
529   </wsn-br:ConsumerReference?>  
530 </wsn-br:RegisterPublisherResponse>  
531 ...
```

532 The WS-Addressing [action] Message Addressing Property MUST contain the URI

533 <http://docs.oasis-open.org/wsn/brw-2/RegisterPublisher/RegisterPublisherResponse>

534 The components of the RegisterPublisher response message are further described as follows:

535 /wsn-br:RegisterPublisherResponse/PublisherRegistrationReference

536 A WS-Addressing endpoint reference to the PublisherRegistration resource created by
537 the RegisterPublisher request message. This element MUST be present in the

538 RegisterPublisher response message. The NotificationBroker may choose to include
539 extra information such as reference parameters in this reference.

540 /wsn-br:RegisterPublisherResponse/ConsumerReference

541 A WS-Addressing endpoint reference to a NotificationConsumer resource that accepts
542 notifications for this publisher registration.

543 Any Notification Messages sent by the Publisher (and considered to take place under this
544 registration) MUST be sent to this endpoint reference.

545 The NotificationBroker MAY use this as a mechanism for identifying the Publisher as
546 having registered.

547 If the NotificationBroker does not respond to the RegisterPublisher request message with the
548 RegisterPublisherResponse message, then it MUST send a fault. The NotificationBroker MUST
549 fault if it rejects the publisher registration. This specification defines the following faults associated
550 with failure to process the RegisterPublisher request message:

551

552 ResourceUnknownFault

- 553 • The NotificationBroker is acting as a WS-Resource, and the resource identified in the
554 message is not known to the Web service. This fault is specified by the WS-Resource
555 [WS-Resource] specification.

556 InvalidTopicExpressionFault

- 557 • The TopicExpression presented in the request message is invalid. This fault is specified
558 in WS-BaseNotification.

559 TopicNotSupportedFault

- 560 • The TopicExpression does not match any Topic supported by the NotificationBroker. This
561 fault is specified in WS-BaseNotification.

562 PublisherRegistrationRejectedFault

- 563 • The publisher registration is rejected by the NotificationBroker. The NotificationBroker
564 MAY provide a hint in the fault message indicating why the registration is rejected.

565 PublisherRegistrationFailedFault

- 566 • The publisher registration process has failed. The NotificationBroker MAY include a hint
567 in the fault message indicating why the registration is failed.

568 UnacceptableInitialTerminationTimeFault

- 569 • The value of InitialTerminationTime specified in the RegisterPublisher request message
570 is not acceptable to the NotificationBroker. The NotificationBroker MAY include a hint in
571 the fault message indicating why the value is unacceptable.

Deleted: succeed in

Deleted: ing

572 6.1.1 Example SOAP Encoding of the RegisterPublisher Message 573 Exchange

574 The following is a non-normative example of a RegisterPublisher request message using SOAP
575 [1.1 \[SOAP 1.1\]](#) or [SOAP 1.2 \[SOAP 1.2\]](#):

```
576 <s:Envelope ... >  
577   <s:Header>  
578     <wsa:Action>  
579       http://docs.oasis-open.org/wsn/brw-  
580 2/RegisterPublisher/RegisterPublisherRequest  
581     </wsa:Action>  
582     ...  
583   </s:Header>  
584   <s:Body>  
585     <wsn-br:RegisterPublisher>  
586       <wsn-br:PublisherReference>  
587         <wsa:Address>  
588           http://www.example.org/PublisherEndpoint  
589         </wsa:Address>  
590         <wsa:ReferenceParameters>  
591           <npex: NPResourceDisambiguator>  
592             uuid:84decd55-7d3f-65ad-ac44-675d9fce5d22  
593           </npex: NPResourceDisambiguator>  
594         </wsa:ReferenceParameters>  
595       </wsn-br:PublisherReference>  
596       <wsn-br:Topic Dialect="http://docs.oasis-open.org/wsn/t-  
597 1/TopicExpression/Simple">  
598         npex:SomeTopic  
599       </wsn-br:Topic>  
600       <wsn-br:Demand>  
601         true  
602       </wsn-br:Demand>  
603       <wsn-br:InitialTerminationTime>  
604         2003-12-25T00:00:00.000000Z  
605       </wsn-br:InitialTerminationTime>  
606     </wsn-br:RegisterPublisher>  
607   </s:Body>  
608 </s:Envelope>
```

609
610 The following is a non-normative example of a RegisterPublisher response message using
611 SOAP:

```
612 <s:Envelope ... >  
613   <s:Header>  
614     <wsa:Action>  
615       http://docs.oasis-open.org/wsn/brw-  
616 2/RegisterPublisher/RegisterPublisherResponse  
617     </wsa:Action>  
618     ...  
619   </s:Header>
```

```
620 <s:Body>
621   <wsn-br:RegisterPublisherResponse>
622     <wsn-br:PublisherRegistrationReference>
623       <wsa:Address>
624         http://www.example.org/PublisherRegistrationManager
625       </wsa:Address>
626       <wsa:ReferenceParameters>
627         <npex:NPubResourceId>
628           uuid:95fefeb3-f37d-5dfe-44fe-221d9fceed99
629         </npex:NPubResourceId>
630       </wsa:ReferenceParameters>
631     </wsn-br:PublisherRegistrationReference>
632     <wsn-br:ConsumerReference>
633       <wsa:Address>
634         http://www.example.org/NotificationConsumer
635       </wsa:Address>
636       ...
637     </wsn-br:ConsumerReference>
638   </wsn-br:RegisterPublisherResponse>
639 </s:Body>
640 </s:Envelope>
```

7 PublisherRegistrationManager Interface

641

642 The PublisherRegistrationManager interface defines message exchanges to manipulate
643 PublisherRegistration resources. The PublisherRegistrationManager MAY [expose](#)
644 [PublisherRegistrations as WS-Resources](#), and if [it does then](#), the PublisherRegistrationManager
645 MUST support the immediate termination interface defined by WS-ResourceLifetime and it MAY
646 support the scheduled termination interface defined by WS-ResourceLifetime.

647 If the PublisherRegistrationManager does not respond to a request message with a response
648 message defined in this specification, then it MUST send a fault. The WS-ResourceProperties
649 and WS-ResourceLifetime [specifications](#) define some of these fault messages.

Deleted: be a

Deleted: it is

Deleted: ,

Deleted: WS-Resource

Deleted: RF

Deleted:

Deleted: RF

Deleted:

Deleted: d

Deleted: If it does so

7.1 PublisherRegistration Resource Properties

650
651 In addition to the message exchanges described in this specification, a
652 PublisherRegistrationManager MAY also support the required message exchanges defined in the
653 WS-ResourceProperties specification and MAY support the optional message exchanges defined
654 in the WS-ResourceProperties specification. [In such cases](#), the Resource Properties document
655 defined by the PublisherRegistrationManager MUST also include references to the following
656 resource property elements:

```
657 .....  
658     targetNamespace="http://docs.oasis-open.org/wsn/br-2"  
659     ...  
660     <xsd:element name="PublisherReference"  
661         type="wsa:EndpointReferenceType"/>  
662     <xsd:element name="Topic" type="wsn-b:TopicExpressionType"/>  
663     <xsd:element name="Demand" type="xsd:boolean" />  
664     <xsd:element name="CreationTime" type="xsd:dateTime" />  
665     ...
```

666 Furthermore, these references MUST reflect the minOccurs and maxOccurs properties as
667 follows:

```
668     <xsd:element ref="wsn-br:PublisherReference"  
669         minOccurs="0" maxOccurs="1" />  
670     <xsd:element ref="wsn-br:Topic"  
671         minOccurs="0" maxOccurs="unbounded" />  
672     <xsd:element ref="wsn-br:Demand"  
673         minOccurs="1" maxOccurs="1" />  
674     <xsd:element ref="wsn-br:CreationTime"  
675         minOccurs="0" maxOccurs="1" />
```

676 These resource property elements are further constrained as follows:

677 /wsn-br:PublisherReference, /wsn-br:Topic, and /wsn-br:Demand

678 These elements are defined in the description of the RegisterPublisher request message
679 (see 6.1).

680 /wsn-br:CreationTime

681 Indicates the date and time at which the PublisherRegistration was created. [This](#)
682 [component MAY be omitted, for example by resource-constrained devices that cannot](#)
683 [associate a creation time with PublisherRegistration resources](#) they create.

684 If [PublisherRegistration](#) is [exposed as](#) a WS-Resource, the [PublisherRegistrationManager MAY](#)
685 [permit the](#) following resource properties [to](#) be modified by [a](#) requestor, by sending a
686 SetResourceProperties request message as defined in the WS-ResourceProperties specification:

- [/wsn-br:Topic](#) and [/wsn-br:Demand](#),

688 Note: /wsn-br:Demand may not take the value "true" if there is no /wsn-
689 br:PublisherReference resource property element in the resource property document.
690

691 7.2 DestroyRegistration

692 The PublisherRegistrationManager interface provides a destroy operation, providing a means by
693 which a requestor can terminate the [PublisherRegistration](#) resource. The DestroyRegistration
694 request message has the following form:

```

695 <wsn-br:DestroyRegistration>
696 {any} *
697 </wsn-br:DestroyRegistration>
698
699

```

700 The WS-Addressing [action] Message Addressing Property MUST contain the URI
701 <http://docs.oasis-open.org/wsn/brw-2/PublisherRegistrationManager/DestroyRegistrationRequest>.

702 The DestroyRegistration request message allows for open content and contains an extension
703 component

704 /wsn-br:DestroyRegistration/{any}.

705

706 Upon receipt of the DestroyRegistration request, the PublisherRegistrationManager MUST
707 attempt to destroy [the PublisherRegistration resource](#). If the DestroyRegistration request
708 message is successfully processed, the PublisherRegistrationManager MUST respond with the
709 following message:

```

710 <wsn-br:DestroyRegistrationResponse>
711 {any} *
712 </wsn-br:DestroyRegistrationResponse>
713
714

```

715 The WS-Addressing [action] Message Addressing Property MUST contain the URI

716 [http://docs.oasis-open.org/wsn/brw-](http://docs.oasis-open.org/wsn/brw-2/PublisherRegistrationManager/DestroyRegistrationResponse)
717 [2/PublisherRegistrationManager/DestroyRegistrationResponse](http://docs.oasis-open.org/wsn/brw-2/PublisherRegistrationManager/DestroyRegistrationResponse).

718 If the PublisherRegistrationManager does not respond to the DestroyRegistration request
719 message with the DestroyRegistrationResponse message, then it MUST send a fault. This

wsn-ws_brokered_notification-1.3-spec-pr-03

5/31/2006

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

Page 25 of 43

Deleted: This is an optional component, supporting resource constrained devices which cannot associate a creation time with PublisherRegistration resources

Comment: WSN 3.31 resolution

Deleted: Manager

Deleted: MAY

Deleted: the

Deleted: Expression

Deleted: ¶

Comment: WSN 2.65 resolution

Deleted: publisher registration manager

Deleted: To terminate PublisherRegistrationManager resource, a requestor MUST send a DestroyRegistration request message to the PublisherRegistrationManager.

Deleted: .

Deleted: itself

Deleted: /

Deleted: ¶

720 specification defines the following faults associated with failure to process the
721 DestroyRegistration request message:

722 ResourceUnknownFault

- 723 • The PublisherRegistration is a WS-Resource, and the resource identified in the message
724 is not known to the Web service. This fault is specified by the WS-Resource [WS-
725 Resource] specification.

Deleted: Manager

726 ResourceNotDestroyedFault

- 727 • The PublisherRegistrationManager was unable to destroy the PublisherRegistration,
728 resource for some reason.

Deleted: Manager

729 7.2.1 Example SOAP Encoding of the DestroyRegistration Message 730 Exchange

731 The following is a non-normative example of a DestroyRegistration request message using
732 SOAP:

```
733 <s:Envelope ... >  
734   <s:Header>  
735     <wsa:Action>  
736       http://docs.oasis-open.org/wsn/brw-  
737 2/PublisherRegistrationManager/DestroyRegistrationRequest  
738     </wsa:Action>  
739     ...  
740   </s:Header>  
741   <s:Body>  
742     <wsn-br:DestroyRegistration/>  
743   </s:Body>  
744 </s:Envelope>
```

745 The following is a non-normative example of a DestroyRegistration response message using
746 SOAP:

```
747 <s:Envelope ... >  
748   <s:Header>  
749     <wsa:Action>  
750       http://docs.oasis-open.org/wsn/brw-  
751 2/PublisherRegistrationManager/DestroyRegistrationResponse  
752     </wsa:Action>  
753     ...  
754   </s:Header>  
755   <s:Body>  
756     <wsn-br:DestroyRegistrationResponse/>  
757   </s:Body>  
758 </s:Envelope>
```

8 Security Considerations

759

760 Baseline security considerations for WS-Notification are discussed in WS-BaseNotification
761 specification. This section only covers additional broker specific security measurements.

8.1 Securing PublisherRegistration

762

763

764 In addition to the security policies for Notification process and Subscription process, WS-
765 BrokeredNotification should provide policies such that:

766

1. only authorized Publishers can register with a NotificationBroker

767

2. only messages of authorized Publishers and of registered topics, can be accepted by a
768 NotificationBroker

Deleted: the

769

3. only authorized principals can modify or delete PublisherRegistration resource

770

771 Given that WS-BrokeredNotification may implement WS-ResourceProperties and WS-
772 ResourceLifetime, the security considerations outlined in these specifications need to be taken
773 into account where appropriate. Authorization policies for those Resource Properties should be
774 put in place so that the implications of providing the state information (through
775 GetResourceProperty request messages) or through notification of state change and modification
776 of the resource properties (through SetResourceProperty request messages), are taken into
777 account.

777

778 A NotificationBroker can support the security measurements of NotificationProducers and
779 NotificationConsumers mentioned in WS-BaseNotification. Acting as an intermediary, [A](#)
780 NotificationBroker MAY also provide convenience to security management, including but not
781 limited to:

781

- Controlling who can publish on a topic at publisher registration time

782

- Refusing to relay messages from unauthorized publishers

783

- Imposing security measurements on all messaging routing through the broker

784

- Providing convenience in messaging security management based on topics.

785

786 NotificationBrokers SHOULD advertise, whether through policy assertions or other means, what
787 security measures they take.

Deleted: ¶

787

9 References

9.1 Normative

790	[RFC2119]		
791		S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels", IETF	
792		RFC 2119, March 1997, http://www.ietf.org/rfc/rfc2119.txt	Deleted: ,
793	[XML]		Deleted:
794		"Extensible Markup Language (XML) 1.0", W3C Recommendation.	Deleted: , IETF RFC 2119,
795		http://www.w3.org/TR/REC-xml	Deleted: ,
796	[XML-Infoset]		
797		"XML Information Set", W3C Recommendation. http://www.w3.org/TR/xml-infoset/	
798	[WS-Addressing]		
799		"Web Services Addressing 1.0 - Core", W3C Recommendation.	
800		http://www.w3.org/TR/ws-addr-core	Deleted: ¶
801	[WS-BaseNotification]		Deleted: .
802		"Web Services Base Notification 1.3".	
803		http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-pr-03.pdf	Deleted: d
804	[WS-Topics]		Deleted: 01
805		"Web Services Topics 1.3".	Deleted: .
806		http://docs.oasis-open.org/wsn/wsn-ws_topics-1.3-spec-pr-02.pdf	Deleted: WS-
807	[WS-Resource]		Deleted: T
808		"Web Services Resource 1.2", OASIS Standard.	Deleted: cd
809		http://docs.oasis-open.org/wsr/wsr-ws_resource-1.2-spec-os.pdf	Deleted: 1
810	[WS-ResourceLifetime]		Deleted: pr-02
811		"Web Services Resource Lifetime 1.2", OASIS Standard.	Deleted: .
812		http://docs.oasis-open.org/wsr/wsr-ws_resource_lifetime-1.2-spec-os.pdf	Deleted: pr-02
813	[WS-ResourceProperties]		Deleted: .
814		"Web Services Resource Properties 1.2", OASIS Standard.	Deleted: pr-02
815		http://docs.oasis-open.org/wsr/wsr-ws_resource_properties-1.2-spec-os.pdf	Deleted: .
816	[WS-BaseFaults]		Deleted: pr-02
817		"Web Services Base Faults 1.2", OASIS Standard.	Deleted: .
818		http://docs.oasis-open.org/wsr/wsr-ws_base_faults-1.2-spec-os.pdf	Deleted: pr-02
819			Deleted: pr-02

9.2 Non-Normative

821	[SOAP 1.1]	
822		"Simple Object Access Protocol (SOAP) 1.1"
823		http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

824
825
826
827
828
829
830

[SOAP 1.2]

[“SOAP Version 1.2 Part 1: Messaging Framework”, W3C Recommendation.](http://www.w3.org/TR/soap12-part1/)
<http://www.w3.org/TR/soap12-part1/>

Deleted: .

[WS-Security]

[“Web Services Security: SOAP Message Security 1.0”, OASIS Standard.](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf)
<http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf>

Deleted: .

Deleted: ¶

831 Appendix A. Acknowledgments

832 The following individuals were members of the committee during the development of this
833 specification:

834 Sid Askary, Individual, Fred Carter, AmberPoint, Martin Chapman, Oracle, Dave Chappell, Sonic
835 Software, Rick Cobb, KnowNow, Ugo Corda, SeeBeyond Technology Corporation, John Fuller,
836 Individual, Stephen Graham, IBM, David Hull, Tibco, Hideharu Kato, Hitachi, Lily Liu,
837 webMethods, Tom Maguire, IBM, Susan Malaika, IBM, Samuel Meder, Argonne National
838 Laboratory, Bryan Murray, Hewlett-Packard, Peter Niblett, IBM, Sanjay Patil, SAP, Mark Peel,
839 Novell, Matt Roberts, IBM, Igor Sedukhin, Computer Associates, David Snelling, Fujitsu, Latha
840 Srinivasan, Hewlett-Packard, William Vambenepe, Hewlett-Packard, Kirk Wilson, Computer
841 Associates.

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

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

847 Tim Banks (IBM), Nick Butler (IBM), Doug Davis (IBM), John Dinger (IBM), Don Ferguson (IBM),
848 Jeff Frey (IBM), Andreas Koepfel (SAP), Heather Kreger (IBM), Amy Lewis (TIBCO Software),
849 Kevin Liu (SAP), Nataraj Nagaratnam (IBM), Martin Nally (IBM), Jeff Nick (IBM), Jay Parikh
850 (Akamai Technologies), Claus von Riegen (SAP), Rick Rineholt (IBM), John Rofrano (IBM),
851 Shivajee Samdarshi (TIBCO Software), Igor Sedukhin (Computer Associates), Eugène
852 Sindambiwe (SAP), Jay Unger (IBM), Bill Wehl (Akamai Technologies), Mark Weitzel (IBM), Dan
853 Wolfson (IBM).

854

Appendix B. XML Schema

855
856

The XML types and elements used in WS-BrokeredNotification are defined in the following XML Schema

857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
OASIS takes no position regarding the validity or scope of any
intellectual property or other rights that might be claimed to pertain
to the implementation or use of the technology described in this
document or the extent to which any license under such rights might or
might not be available; neither does it represent that it has made any
effort to identify any such rights. Information on OASIS's procedures
with respect to rights in OASIS specifications can be found at the OASIS
website. Copies of claims of rights made available for publication and
any assurances of licenses to be made available, or the result of an
attempt made to obtain a general license or permission for the use of
such proprietary rights by implementors or users of this specification,
can be obtained from the OASIS Executive Director.

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

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

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

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

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

Deleted: 5

902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953

```
<xsd:schema
  xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wsn-br="http://docs.oasis-open.org/wsn/br-2"
  xmlns:wsn-b="http://docs.oasis-open.org/wsn/b-2"
  xmlns:wsrf-bf="http://docs.oasis-open.org/wsrf/bf-2"
  xmlns:wstop="http://docs.oasis-open.org/wsn/t-1"
  targetNamespace="http://docs.oasis-open.org/wsn/br-2"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">

<!-- ===== Imports ===== -->

  <xsd:import namespace="http://www.w3.org/2005/08/addressing"
    schemaLocation="http://www.w3.org/2005/08/addressing/ws-
addr.xsd" />

  <xsd:import namespace="http://docs.oasis-open.org/wsrf/bf-2"
    schemaLocation="http://docs.oasis-open.org/wsrf/bf-
2.xsd" />

  <xsd:import namespace="http://docs.oasis-open.org/wsn/b-2"
    schemaLocation="http://docs.oasis-open.org/wsn/b-2.xsd" />

  <xsd:import namespace="http://docs.oasis-open.org/wsn/t-1"
    schemaLocation="http://docs.oasis-open.org/wsn/t-1.xsd" />

<!-- ===== Resource Properties for NotificationBroker ===== -->
  <xsd:element name="RequiresRegistration" type="xsd:boolean" />

<!-- ===== Resource Properties for PublisherRegistration ===== -->
  <xsd:element name="PublisherReference"
    type="wsa:EndpointReferenceType" />
  <xsd:element name="ConsumerReference"
    type="wsa:EndpointReferenceType" />
  <xsd:element name="Topic"
    type="wsn-b:TopicExpressionType" />
  <xsd:element name="Demand"
    type="xsd:boolean" />
  <xsd:element name="CreationTime"
    type="xsd:dateTime" />
  <xsd:element name="NotificationBrokerRP">
    <xsd:complexType>
      <xsd:sequence>
        <!-- From NotificationProducer -->
        <xsd:element ref="wsn-b:TopicExpression"
          minOccurs="0" maxOccurs="unbounded" />
        <xsd:element ref="wsn-b:FixedTopicSet"
          minOccurs="0" maxOccurs="1" />
        <xsd:element ref="wsn-b:TopicExpressionDialect" />
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```



```

954         minOccurs="0" maxOccurs="unbounded" />
955     <xsd:element ref="wstop:TopicSet"
956
957         minOccurs="0" maxOccurs="1" />
958     <!-- NotificationBroker specific -->
959     <xsd:element ref="wsn-br:RequiresRegistration"
960         minOccurs="1" maxOccurs="1" />
961     </xsd:sequence>
962 </xsd:complexType>
963 </xsd:element>
964 <!-- ===== Resource Properties for PublisherRegistration ===== -->
965 <xsd:element name="PublisherRegistrationRP">
966     <xsd:complexType>
967         <xsd:sequence>
968             <xsd:element ref="wsn-br:PublisherReference"
969                 minOccurs="0" maxOccurs="1" />
970             <xsd:element ref="wsn-br:Topic"
971                 minOccurs="0" maxOccurs="unbounded" />
972             <xsd:element ref="wsn-br:Demand"
973                 minOccurs="1" maxOccurs="1" />
974             <xsd:element ref="wsn-br:CreationTime"
975                 minOccurs="0" maxOccurs="1" />
976         </xsd:sequence>
977     </xsd:complexType>
978 </xsd:element>
979
980 <!-- ===== Message Types for NotificationBroker ===== -->
981 <xsd:element name="RegisterPublisher">
982     <xsd:complexType>
983         <xsd:sequence>
984             <xsd:element name="PublisherReference"
985                 type="wsa:EndpointReferenceType"
986                 minOccurs="0" maxOccurs="1" />
987             <xsd:element name="Topic"
988                 type="wsn-b:TopicExpressionType"
989                 minOccurs="0" maxOccurs="unbounded" />
990             <xsd:element name="Demand"
991                 type="xsd:boolean" default="false"
992                 minOccurs="0" maxOccurs="1" />
993             <xsd:element name="InitialTerminationTime"
994                 type="xsd:dateTime"
995                 minOccurs="0" maxOccurs="1" />
996             <xsd:any namespace="##other" processContents="lax"
997                 minOccurs="0" maxOccurs="unbounded" />
998         </xsd:sequence>
999     </xsd:complexType>
1000 </xsd:element>
1001
1002 <xsd:element name="RegisterPublisherResponse">
1003     <xsd:complexType>
1004         <xsd:sequence>
1005             <xsd:element name="PublisherRegistrationReference"

```

```

Deleted: <!--
From WS-
ResourceLifetime
ScheduledResourceTermin
ation -->¶
<xsd:element ref="wsn-
b:CurrentTime" ¶
minOccurs="0"
maxOccurs="1" /> ¶
<xsd:element ref="wsn-
b:TerminationTime" ¶
minOccurs="1"
maxOccurs="1" />¶
<!--
PublisherRegistration
specific -->¶

```

```

1006         type="wsa:EndpointReferenceType"
1007         minOccurs="1" maxOccurs="1" />
1008     <xsd:element name="ConsumerReference"
1009                 type="wsa:EndpointReferenceType"
1010                 minOccurs="0" maxOccurs="1" />
1011
1012         </xsd:sequence>
1013     </xsd:complexType>
1014 </xsd:element>
1015
1016 <xsd:complexType name="PublisherRegistrationRejectedFaultType">
1017     <xsd:complexContent>
1018         <xsd:extension base="wsrf-bf:BaseFaultType" />
1019     </xsd:complexContent>
1020 </xsd:complexType>
1021 <xsd:element name="PublisherRegistrationRejectedFault"
1022             type="wsn-br:PublisherRegistrationRejectedFaultType" />
1023
1024 <xsd:complexType name="PublisherRegistrationFailedFaultType">
1025     <xsd:complexContent>
1026         <xsd:extension base="wsrf-bf:BaseFaultType" />
1027     </xsd:complexContent>
1028 </xsd:complexType>
1029 <xsd:element name="PublisherRegistrationFailedFault"
1030             type="wsn-br:PublisherRegistrationFailedFaultType" />
1031
1032
1033
1034 <xsd:element name="DestroyRegistration">
1035     <xsd:complexType>
1036         <xsd:sequence>
1037             <xsd:any namespace="##other" processContents="lax"
1038                 minOccurs="0" maxOccurs="unbounded" />
1039         </xsd:sequence>
1040         <xsd:anyAttribute />
1041     </xsd:complexType>
1042 </xsd:element>
1043
1044 <xsd:element name="DestroyRegistrationResponse">
1045     <xsd:complexType>
1046         <xsd:sequence>
1047             <xsd:any namespace="##other" processContents="lax"
1048                 minOccurs="0" maxOccurs="unbounded" />
1049         </xsd:sequence>
1050         <xsd:anyAttribute />
1051     </xsd:complexType>
1052 </xsd:element>
1053
1054 <xsd:complexType name="ResourceNotDestroyedFaultType">
1055     <xsd:complexContent>
1056         <xsd:extension base="wsrf-bf:BaseFaultType" />
1057     </xsd:complexContent>

```

1058
1059
1060
1061
1062

```
</xsd:complexType>  
<xsd:element name="ResourceNotDestroyedFault "  
            type="wsn-br:ResourceNotDestroyedFaultType" />  
</xsd:schema>
```

1063 Appendix C. WSDL 1.1

1064 The following illustrates the WSDL 1.1 for the Web service methods described in this
1065 specification:

```
1066 <?xml version="1.0" encoding="utf-8"?>
1067 <!--
1068 OASIS takes no position regarding the validity or scope of any
1069 intellectual property or other rights that might be claimed to pertain
1070 to the implementation or use of the technology described in this
1071 document or the extent to which any license under such rights might or
1072 might not be available; neither does it represent that it has made any
1073 effort to identify any such rights. Information on OASIS's procedures
1074 with respect to rights in OASIS specifications can be found at the OASIS
1075 website. Copies of claims of rights made available for publication and
1076 any assurances of licenses to be made available, or the result of an
1077 attempt made to obtain a general license or permission for the use of
1078 such proprietary rights by implementors or users of this specification,
1079 can be obtained from the OASIS Executive Director.
1080
1081 OASIS invites any interested party to bring to its attention any
1082 copyrights, patents or patent applications, or other proprietary rights
1083 which may cover technology that may be required to implement this
1084 specification. Please address the information to the OASIS Executive
1085 Director.
1086
1087 Copyright (C) OASIS Open (2004-2006). All Rights Reserved.
1088
1089 This document and translations of it may be copied and furnished to
1090 others, and derivative works that comment on or otherwise explain it or
1091 assist in its implementation may be prepared, copied, published and
1092 distributed, in whole or in part, without restriction of any kind,
1093 provided that the above copyright notice and this paragraph are included
1094 on all such copies and derivative works. However, this document itself
1095 may not be modified in any way, such as by removing the copyright notice
1096 or references to OASIS, except as needed for the purpose of developing
1097 OASIS specifications, in which case the procedures for copyrights
1098 defined in the OASIS Intellectual Property Rights document must be
1099 followed, or as required to translate it into languages other than
1100 English.
1101
1102 The limited permissions granted above are perpetual and will not be
1103 revoked by OASIS or its successors or assigns.
1104
1105 This document and the information contained herein is provided on an "AS
1106 IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED,
1107 INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE
1108 INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED
1109 WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.
```

Deleted: 5

```

1110 -->
1111
1112 <wsdl:definitions name="WS-BrokeredNotification"
1113   xmlns="http://schemas.xmlsoap.org/wsdl/"
1114   xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
1115   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1116   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1117   xmlns:wsa="http://www.w3.org/2005/08/addressing"
1118   xmlns:wsn-br="http://docs.oasis-open.org/wsn/br-2"
1119   xmlns:wsn-brw="http://docs.oasis-open.org/wsn/brw-2"
1120   xmlns:wsn-b="http://docs.oasis-open.org/wsn/b-2"
1121   xmlns:wsn-bw="http://docs.oasis-open.org/wsn/bw-2"
1122   xmlns:wsrf-bf="http://docs.oasis-open.org/wsrf/bf-2"
1123   xmlns:wsrf-rw="http://docs.oasis-open.org/wsrf/rw-2"
1124   targetNamespace="http://docs.oasis-open.org/wsn/brw-2">
1125
1126 <!-- ===== Imports ===== -->
1127   <wsdl:import namespace="http://docs.oasis-open.org/wsrf/rw-2"
1128     location="http://docs.oasis-open.org/wsrf/rw-2.wsdl"/>
1129
1130   <wsdl:import namespace="http://docs.oasis-open.org/wsn/bw-2"
1131     location="http://docs.oasis-open.org/wsn/bw-2.wsdl"/>
1132
1133 <!-- ===== Types Definitions ===== -->
1134   <wsdl:types>
1135     <xsd:schema>
1136       <xsd:import
1137         namespace="http://docs.oasis-open.org/wsn/br-2"
1138         schemaLocation="http://docs.oasis-open.org/wsn/br-2.xsd"/>
1139     </xsd:schema>
1140   </wsdl:types>
1141
1142 <!-- ===== NotificationBroker::RegisterPublisher =====
1143   RegisterPublisher(PublisherReference, TopicExpression* ,
1144     [Demand], [InitialTerminationTime])
1145   returns: WS-Resource qualified EPR to a PublisherRegistration -->
1146   <wsdl:message name="RegisterPublisherRequest">
1147     <wsdl:part name="RegisterPublisherRequest"
1148       element="wsn-br:RegisterPublisher"/>
1149   </wsdl:message>
1150
1151   <wsdl:message name="RegisterPublisherResponse">
1152     <wsdl:part name="RegisterPublisherResponse"
1153       element="wsn-br:RegisterPublisherResponse"/>
1154   </wsdl:message>
1155
1156   <wsdl:message name="PublisherRegistrationRejectedFault">
1157     <wsdl:part name="PublisherRegistrationRejectedFault"
1158       element="wsn-br:PublisherRegistrationRejectedFault"/>
1159   </wsdl:message>
1160
1161   <wsdl:message name="PublisherRegistrationFailedFault">

```

```

1162     <wsdl:part name="PublisherRegistrationFailedFault"
1163         element="wsn-br:PublisherRegistrationFailedFault" />
1164 </wsdl:message>
1165
1166 <wsdl:message name="DestroyRegistrationRequest">
1167     <wsdl:part name="DestroyRegistrationRequest"
1168         element="wsn-br:DestroyRegistration" />
1169 </wsdl:message>
1170
1171 <wsdl:message name="DestroyRegistrationResponse">
1172     <wsdl:part name="DestroyRegistrationResponse"
1173         element="wsn-br:DestroyRegistrationResponse" />
1174 </wsdl:message>
1175
1176 <wsdl:message name="ResourceNotDestroyedFault">
1177     <wsdl:part name="ResourceNotDestroyedFault"
1178         element="wsn-br:ResourceNotDestroyedFault" />
1179 </wsdl:message>
1180
1181 <!-- ===== PortType Definitions ===== -->
1182
1183 <!-- ===== RegisterPublisher ===== -->
1184 <wsdl:portType name="RegisterPublisher">
1185     <wsdl:operation name="RegisterPublisher">
1186         <wsdl:input message="wsn-brw:RegisterPublisherRequest" />
1187         <wsdl:output message="wsn-brw:RegisterPublisherResponse" />
1188         <wsdl:fault name="ResourceUnknownFault"
1189             message="wsrf-rw:ResourceUnknownFault" />
1190         <wsdl:fault name="InvalidTopicExpressionFault"
1191             message="wsn-bw:InvalidTopicExpressionFault" />
1192         <wsdl:fault name="TopicNotSupportedFault"
1193             message="wsn-bw:TopicNotSupportedFault" />
1194         <wsdl:fault name="PublisherRegistrationRejectedFault"
1195             message="wsn-brw:PublisherRegistrationRejectedFault" />
1196         <wsdl:fault name="PublisherRegistrationFailedFault"
1197             message="wsn-brw:PublisherRegistrationFailedFault" />
1198         <wsdl:fault name="UnacceptableInitialTerminationTimeFault"
1199             message="wsn-bw:UnacceptableInitialTerminationTimeFault" />
1200     </wsdl:operation>
1201 </wsdl:portType>
1202
1203 <!-- ===== NotificationBroker PortType Definition ===== -->
1204 <wsdl:portType name="NotificationBroker">
1205     <!-- ===== extends NotificationConsumer ===== -->
1206     <wsdl:operation name="Notify">
1207         <wsdl:input message="wsn-bw:Notify" />
1208     </wsdl:operation>
1209
1210     <!-- ===== extends NotificationProducer ===== -->
1211     <wsdl:operation name="Subscribe">
1212         <wsdl:input message="wsn-bw:SubscribeRequest" />
1213         <wsdl:output message="wsn-bw:SubscribeResponse" />

```

```

1214 <wsdl:fault name="ResourceUnknownFault"
1215         message="wsrf-rw:ResourceUnknownFault" />
1216 <wsdl:fault name="InvalidFilterFault"
1217         message="wsn-bw:InvalidFilterFault" />
1218 <wsdl:fault name="TopicExpressionDialectUnknownFault"
1219         message="wsn-bw:TopicExpressionDialectUnknownFault" />
1220 <wsdl:fault name="InvalidTopicExpressionFault"
1221         message="wsn-bw:InvalidTopicExpressionFault" />
1222 <wsdl:fault name="TopicNotSupportedFault"
1223         message="wsn-bw:TopicNotSupportedFault" />
1224 <wsdl:fault name="InvalidProducerPropertiesExpressionFault"
1225         message="wsn-bw:InvalidProducerPropertiesExpressionFault" />
1226 <wsdl:fault name="InvalidMessageContentExpressionFault"
1227         message="wsn-bw:InvalidMessageContentExpressionFault" />
1228 <wsdl:fault name="UnacceptableInitialTerminationTimeFault"
1229         message="wsn-bw:UnacceptableInitialTerminationTimeFault" />
1230 <wsdl:fault name="UnrecognizedPolicyRequestFault"
1231         message="wsn-bw:UnrecognizedPolicyRequestFault" />
1232 <wsdl:fault name="UnsupportedPolicyRequestFault"
1233         message="wsn-bw:UnsupportedPolicyRequestFault" />
1234 <wsdl:fault name="NotifyMessageNotSupportedFault"
1235         message="wsn-bw:NotifyMessageNotSupportedFault" />
1236 <wsdl:fault name="SubscribeCreationFailedFault"
1237         message="wsn-bw:SubscribeCreationFailedFault" />
1238 </wsdl:operation>
1239 <wsdl:operation name="GetCurrentMessage">
1240 <wsdl:input message="wsn-bw:GetCurrentMessageRequest" />
1241 <wsdl:output message="wsn-bw:GetCurrentMessageResponse" />
1242 <wsdl:fault name="ResourceUnknownFault"
1243         message="wsrf-rw:ResourceUnknownFault" />
1244 <wsdl:fault name="TopicExpressionDialectUnknownFault"
1245         message="wsn-bw:TopicExpressionDialectUnknownFault" />
1246 <wsdl:fault name="InvalidTopicExpressionFault"
1247         message="wsn-bw:InvalidTopicExpressionFault" />
1248 <wsdl:fault name="TopicNotSupportedFault"
1249         message="wsn-bw:TopicNotSupportedFault" />
1250 <wsdl:fault name="NoCurrentMessageOnTopicFault"
1251         message="wsn-bw:NoCurrentMessageOnTopicFault" />
1252 <wsdl:fault name="MultipleTopicsSpecifiedFault"
1253         message="wsn-bw:MultipleTopicsSpecifiedFault" />
1254 </wsdl:operation>
1255
1256 <!-- ===== extends RegisterPublisher ===== -->
1257 <wsdl:operation name="RegisterPublisher">
1258 <wsdl:input message="wsn-brw:RegisterPublisherRequest" />
1259 <wsdl:output message="wsn-brw:RegisterPublisherResponse" />
1260 <wsdl:fault name="ResourceUnknownFault"
1261         message="wsrf-rw:ResourceUnknownFault" />
1262 <wsdl:fault name="InvalidTopicExpressionFault"
1263         message="wsn-bw:InvalidTopicExpressionFault" />
1264 <wsdl:fault name="TopicNotSupportedFault"
1265         message="wsn-bw:TopicNotSupportedFault" />

```

```

1266     <wsdl:fault name="PublisherRegistrationRejectedFault"
1267             message="wsn-brw:PublisherRegistrationRejectedFault" />
1268     <wsdl:fault name="PublisherRegistrationFailedFault"
1269             message="wsn-brw:PublisherRegistrationFailedFault" />
1270     <wsdl:fault name="UnacceptableInitialTerminationTimeFault"
1271             message="wsn-bw:UnacceptableInitialTerminationTimeFault" />
1272     </wsdl:operation>
1273
1274 </wsdl:portType>
1275
1276 <!-- ===== PublisherRegistrationManager PortType Definition ===== -->
1277 <wsdl:portType name="PublisherRegistrationManager">
1278
1279     <!--====DestroyRegistration:ImmediateResourceTermination====-->
1280     <wsdl:operation name="DestroyRegistration">
1281         <wsdl:input name="DestroyRegistrationRequest"
1282                 message="wsn-brw:DestroyRegistrationRequest" />
1283         <wsdl:output name="DestroyRegistrationResponse"
1284                 message="wsn-brw:DestroyRegistrationResponse" />
1285         <wsdl:fault name="ResourceUnknownFault"
1286                 message="wsrf-rw:ResourceUnknownFault" />
1287         <wsdl:fault name="ResourceNotDestroyedFault"
1288                 message="wsn-brw:ResourceNotDestroyedFault" />
1289     </wsdl:operation>
1290 </wsdl:portType>
1291 </wsdl:definitions>

```


Appendix D. Revision History

Rev	Date	By Whom	What
1.2 01	2004-05-12	Lily Liu	Initial version
1.2 02	2004-06-07	Dave Chappell	Updates and consistency check w/ other WS-N specs
1.2 03	2004-06-24	Lily Liu, Dave Chappell	Addition of a Goals and Requirements section and minor format changes
1.2 03	2004-07-12	Lily Liu	Addition of a status paragraph
1.3 01a – 1.3 01e	2005-02-01	Dave Chappell, Lily Liu	Series of issue resolution and consistency reviews with WS-BaseNotification
1.3 01f	2005-06-10	Lily Liu	Issues: 3.1, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 3.10, 3.11, 3.12, 3.13, 3.14, 3.15, 3.16, 3.17, 3.18, 3.19, 3.20 Updated the Terminology, Introduction, and Security sections. Updated sections about NotificationBroker and PublisherRegistrationManager resource properties.
1.3 01g	2005-07-01	Lily Liu	Updated the status section. Changed term NotificationMessage to Notification. Added CreatePullPoint portType to NotificationBroker. Completed issue resolutions. Replaced the Abstract section.
1.3 02d	2005-11-04	Lily Liu	Included changes to address: WSN 2.62, WSN 3.23, WSN 3.24, WSN 3.25, WSN 3.26, WSN 3.28, and WSN 3.29 Resolved AI 137, AI 138, AI 141, AI 142 AI 144, and AI 145. Updated references.
1.3 pr02	2006-02-17	Lily Liu	Updated the document with changes suggested in the errata document (http://www.oasis-open.org/apps/org/workgroup/wsn/download.php/16679/wsn-

Rev	Date	By Whom	What
			ws_brokered_notification-1.3-errata.doc .
1.3 cd03	2006- 03-23	Lily Liu	<p>Updated the document again with more changes suggested in the errata document (http://www.oasis-open.org/apps/org/workgroup/wsn/download.php/16679/wsn-ws_brokered_notification-1.3-errata.doc).</p> <p>Updated copyrights and status of the document.</p>
1.3 cd03	2006- 04-20	Peter Niblett	<p>Updated the References section to point at OASIS standard versions of WSRF specifications, and moved the WS-BaseNotification and WS-Topics references on to point to the latest committee drafts</p>

Deleted:

1293 **Appendix E. Notices**

1294 OASIS takes no position regarding the validity or scope of an intellectual property or other rights
1295 that might be claimed to pertain to the implementation or use of the technology described in this
1296 document or the extent to which any license under such rights might or might not be available;
1297 neither does it represent that it has made any effort to identify any such rights. Information on
1298 OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS
1299 website. Copies of claims of rights made available for publication and any assurances of licenses
1300 to be made available, or the result of an attempt made to obtain a general license or permission
1301 for the use of such proprietary rights by implementers or users of this specification, can be
1302 obtained from the OASIS Executive Director.

1303 OASIS invites any interested party to bring to its attention any copyrights, patents or patent
1304 applications, or other proprietary rights which may cover technology that may be required to
1305 implement this specification. Please address the information to the OASIS Executive Director.

1306 Copyright © OASIS Open 2004-2006. All Rights Reserved.

Deleted: 4

1307 This document and translations of it may be copied and furnished to others, and derivative works
1308 that comment on or otherwise explain it or assist in its implementation may be prepared, copied,
1309 published and distributed, in whole or in part, without restriction of any kind, provided that the
1310 above copyright notice and this paragraph are included on all such copies and derivative works.
1311 However, this document itself does not be modified in any way, such as by removing the
1312 copyright notice or references to OASIS, except as needed for the purpose of developing OASIS
1313 specifications, in which case the procedures for copyrights defined in the OASIS Intellectual
1314 Property Rights document must be followed, or as required to translate it into languages other
1315 than English.

1316 The limited permissions granted above are perpetual and will not be revoked by OASIS or its
1317 successors or assigns.

1318 This document and the information contained herein is provided on an "AS IS" basis and OASIS
1319 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO
1320 ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY
1321 RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A
1322 PARTICULAR PURPOSE.