Web Services Brokered Notification 1.2 (WS-BrokeredNotification)

Working Draft 01, 21 July 2004

Document identifier:

wsn-WS-BrokeredNotification-1.2-draft-01

Location:

http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BrokeredNotification-1.2-draft-01.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 white papers and specifications that define a standard Web services approach to notification using a topic-based publish/subscribe pattern. It includes: standard message exchanges to be implemented by service providers that wish to participate in Notifications, standard message exchanges for a notification broker service provider (allowing publication of messages from entities that are not themselves service providers), operational requirements expected of service providers and requestors that participate in notifications, and an XML model that describes topics. The WS-Notification family of documents includes: a white paper: *Publish-Subscribe Notification for Web services* as well as three normative specifications: WS-BaseNotification, WS-BrokeredNotification, and WS-Topics.

This specification defines the Web services interface for the NotificationBroker. A NotificationBroker is an intermediary, which, among other things, allows publication of messages from entities that are not themselves service providers. It includes standard message exchanges to be implemented by NotificationBroker service providers along with operational requirements expected of service providers and requestors that participate in brokered notifications. This work relies upon *WS-Base Notification* and *WS-Topics*, as well as the *Publish-Subscribe Notification for Web Services* document.

Status:

On 21 July 2004, this document was approved by the OASIS WS-Notification Technical Committee for publication so that users of the specification have a stable draft version available until the TC publishes a Committee Draft. Send comments to the editor.

Committee members should send comments on this specification to the wsn@lists.oasis-open.org list. Others should subscribe to and send comments to the wsn-comment@lists.oasis-open.org list. To subscribe, send an email message to wsn-comment-request@lists.oasis-open.org with the word "subscribe" as the body of the message.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the WSN TC web page (http://www.oasis-open.org/committees/wsn/). General OASIS IPR information can be found at http://www.oasis-open.org/who/intellectualproperty.shtml.

Table of Contents

1 Introduction

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 specification defines the Web services interface for the NotificationBroker. A NotificationBroker is an intermediary, which, among other things, allows publication of messages from entities that are not themselves service providers. It includes standard message exchanges to be implemented by NotificationBroker service providers along with operational requirements expected of service providers and requestors that participate in brokered notifications. This work relies upon *WS-Base Notification* and *WS-Topics*, as well as the *Publish-Subscribe Notification for Web Services* document.

1.1 Goals and Requirements

The goal of WS-BrokeredNotification is to standardize message exchanges involved in Web services publish and subscribe of a message broker. The overall objectives of WS-Notification are presented in [WS-Notification Whitepaper]. The following section lists the specific subset of those objectives realized by WS-BrokeredNotification.

1.1.1 Requirements

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

- **Must allow for a notification broker as an intermediary.** A NotificationBroker is an intermediary Web service that decouples NotificationConsumers from Publishers. A notification broker can relieve a Publisher from having to implement message exchanges associated with NotificationProducer; the NotificationBroker takes on the duties of a SubscriptionManager (managing subscriptions) and NotificationProducer (distributing NotificationMessages) on behalf of the Publisher.
- **Must allow for federation of brokers.** It must be possible to build configurations with multiple intermediary broker services in a dynamic fashion. This specification must allow for a variety of broker topology usage patterns. Among other things, these allow for greater scalability and permit sharing of administrative workload.
- **Must provide runtime metadata:** There must be a mechanism that lets a potential Subscriber discover what elements available for a subscription are provided by a NotificationBroker, and in what formats the subscription for a notification can be made.
- **Must conform to WS-BaseNotification:** A NotificationBroker must support required message exchanges defined by the [WS-ResourceProperties] specification. It must conform to the NotificationProducer and the NotificationConsumer interfaces defined in WS-BaseNotification.

- **WS-BrokeredNotification must be independent of binding-level details:** Transport protocol details must be orthogonal to the subscription and the delivery of the notifications, so that the specification can be used over a variety of different transports.
- **Must not exclude non-service producers and subscribers:** WS-BrokeredNotification design must not exclude a non-service entity to deliver a notification message to a NotificationBroker. It must not exclude a NotificationBroker to send a notification to a non-service consumer.
- **Must provide publisher registration:** WS-BrokeredNotification must define standard message exchanges for registering a NotificationPublisher with a NotificationBroker.

1.1.2 Non-Goals

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

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

1.2 Notational Conventions

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

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

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

1.3 Namespaces

The following namespaces are used in this document:

Prefix	Namespace
s12	http://www.w3.org/2003/05/soap-envelope

xsd	http://www.w3.org/2001/XMLSchema
wsa	http://schemas.xmlsoap.org/ws/2003/02/addressing
wsbn	http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BrokeredNotification-1.2-draft-01.xsd
wsnt	http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BaseNotification-1.2-draft-01.xsd
wsrl	http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS- ResourceLifetime-1.2-draft-01.xsd
wsrp	http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceProperties-1.2-draft-01.xsd
wstop	http://docs.oasis-open.org/wsn/2004/06/wsn-WS-Topics-1.2-draft-01.xsd
wsbf	http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-BaseFaults-1.2-draft-01.xsd
wsbfw	http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-BaseFaults-1.2-draft-01.wsdl

2 Relationship to Other Specifications

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

This means that a NotificationBroker, implemented to conform to this specification, must also conform to [WS-BaseNotification]. Such a NotificationBroker can deliver notification messages to NotificationConsumers that are implemented to conform to [WS-BaseNotification], and can subscribe to Notifications distributed by NotificationProducers as defined in [WS-BaseNotification].

In addition a NotificationBroker MUST support hierarchical topics, and the ConcreteTopicPath topic expression dialects defined in [WS-Topics].

Please refer to [WS-Notification Whitepaper] for a description of how WS-Notification relates to other specifications, in particular to the WS-Resource Framework family of specifications.

WS-BrokeredNotification must be composable with other Web services specifications, in particular WS-Security, WS-Policy, WS-Federation, WS-Addressing, WS-Coordination, WS-ResourceProperties, WS-ResourceLifetime, WS-ReliableMessaging [WS-ReliableMessaging] and the WS-Resource framework [State Paper].

3 Terminology and Concepts

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

4 Publishing

There are three distinct stages in the Notification process

- Observation of the Situation and its noteworthy characteristics;
- Creation of the NotificationMessage artifact that captures the noteworthy characteristics of the Situation; and
- Distribution of copies of the NotificationMessage to zero or more interested parties.

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

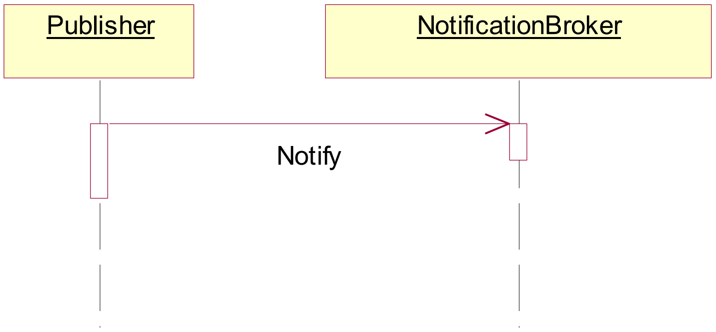
However, the WS-Notification family of specifications does specify how dissemination of messages SHOULD occur. There are two dominant patterns by which NotificationMessages are disseminated in WS-Notification: direct and brokered.

In the direct case, the publishing Web service implements message exchanges associated with the NotificationProducer interface; it is responsible for accepting Subscribe messages and sending NotificationMessages to interested parties. The implementer of this Web service can choose to program this behavior or delegate to specialized implementations of the Subscribe and NotificationMessage delivery behavior. This case is addressed by the WS-BaseNotification specification [WS-BaseNotification].

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

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

The following figure illustrates simple publishing:



In the simple publishing scenario, the Publisher entity is responsible only for the core Publisher functions - observing the Situation and formatting the NotificationMessage artifact that describes the Situation. The dissemination step occurs when the Publisher sends the Notify message to the NotificationBroker.

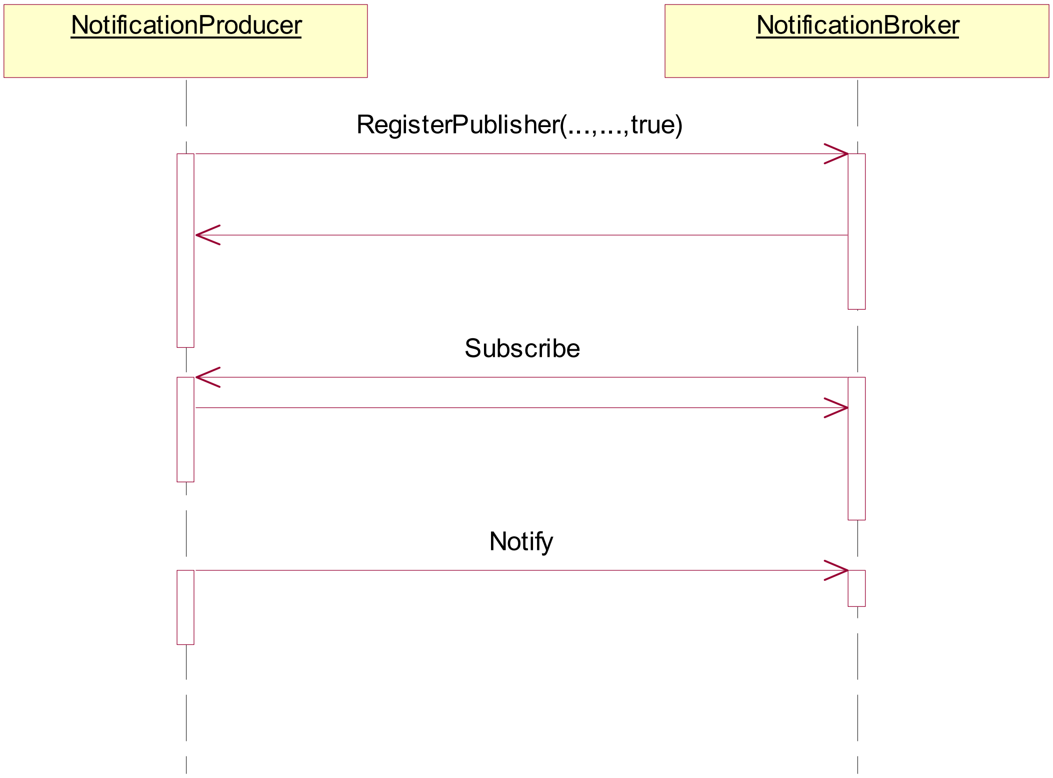
In the composable publishing pattern, the role of the Publisher is played by a Web service that implements NotificationProducer. The act of observing the Situation and formatting the NotificationMessage happens within the implementation logic of the NotificationProducer itself.

The NotificationMessage is disseminated by the NotificationProducer sending the Notify message to a NotificationBroker. The NotificationMessage may also be disseminated by sending the Notify message to any NotificationConsumer that had been previously subscribed to the NotificationProducer.

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

The last pattern, the *demand-based* pattern, requires the Publisher to be a NotificationProducer, and thereby accept the Subscribe message. Demand-based publication is intended for use in cases where the act of observing the Situation or the act of formatting the NotificationMessage artifact might be expensive to perform, and therefore should be avoided if there are no interested parties for that NotificationMessage. To use this pattern, the Publisher must register with the NotificationBroker, using the registration to express the intent to provide demand-based publishing only. Based upon this style of registration, the NotificationBroker sends the Subscribe message to the Publisher (recall: in this situation the Publisher must implement the message exchanges associated with the NotificationProducer interface).

Furthermore, the NotificationBroker is expected to pause its Subscription whenever it has no active Subscribers for the information provided by the Publisher. When the NotificationBroker does have active Subscribers, it is obliged to resume its Subscription to the Publisher.



5 NotificationBroker Interface

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

A NotificationBroker MUST support the required message exchanges defined by the WS-ResourceProperties specification [WS-ResourceProperties] and MAY support the optional message exchanges defined by WS-ResourceProperties.

A NotificationBroker MUST also support message exchanges and Resource Property elements defined by the following interfaces:

- NotificationProducer
- NotificationConsumer

5.1 NotificationBroker Resource Properties

In addition to the Resource Property elements associated with the interfaces the Notification Broker extends, the Notification Broker must also include the following reference property element:

```
<xsd:element name=”RequiresRegistration” type=”xsd:boolean”/>
```

This resource property element is further constrained as follows:

/wsbn:RequiresRegistration

The value is “true” if the NotificationBroker requires a publisher to register (see 5.4) before sending it a Notify (i.e. publish) message on this Topic. The default is “false”.

5.2 Notify

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

A Publisher sends a Notify message to a NotificationBroker in order to publish a NotificationMessage on a given Topic. As a result of the Publisher sending this message, NotificationMessages are delivered to all NotificationConsumers subscribed on the given Topic. For some Topics (those that require a Publisher to pre-register), the requestor must be a registered Publisher in order to successfully publish a NotificationMessage to the given Topic (see 5.4).

5.3 Subscribe

The NotificationBroker MUST support the Subscribe message exchange from the NotificationProducer interface [WS-BaseNotification]. Although a NotificationProducer MAY support any TopicExpression dialect, a NotificationBroker is further constrained in that it MUST accept use of the ConcreteTopicPath dialect (defined in [WS-Topics]) in the Subscribe message’s TopicExpression. It MAY also accept use of the FullTopicPath dialect or any other TopicExpression dialects.

5.4 RegisterPublisher

The RegisterPublisher message is used by the Publisher to confirm its ability to publish on a given Topic or set of Topics.

If an entity wishes to register a publisher, it MUST send a RegisterPublisher request message to the NotificationBroker. The format of the RegisterPublisher request message is:

```
…
<wsbn:RegisterPublisher>
   <wsbn:PublisherReference>
      wsa:EndpointReference
   </wsbn:PublisherReference>?
   <wsbn:Topic dialect = “xsd:anyURI”>
      {any}
   </wsbn:Topic>*
   <wsbn:Demand>
         xsd:Boolean
   </wsbn:Demand>?
   <wsbn:InitialTerminationTime>
      xsd:dateTime
   </wsbn:InitialTerminationTime>?
</wsbn:RegisterPublisher>
…
```

This request message MUST follow the implied resource pattern as outlined in [State Paper]. The components of the RegisterPublisher request message are further described as follows:

/wsbn:PublisherReference

An OPTIONAL EndpointReference to an entity that wishes to become a Publisher on one or more Topics supported by the Notification Broker. This component MUST appear if the /wsbn:Demand component has value “true”. If this component is missing, the Publisher is either not a Web service, or does not wish to receive messages from the NotificationBroker.

/wsbn:Topic

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

/wsbn:Demand

A boolean, default value is “false”. If its value is “true”, then the intent of the Publisher is to use a demand-based model from the NotificationBroker (see 4). In this case, the NotificationBroker must observe the rules associated with demand-based publishing, including establishing a Subscription with the Publisher on those Topics and pausing/resuming those Subscriptions as the NotificationBroker receives Subscriptions for those Topics.

/wsbn:InitialTerminationTime

This component contains the service requestor’s suggestion for the initial termination time of the PublisherRegistration resource being created. This time is relative to the time source used by the NotificationBroker. If the NotificationBroker is unable or unwilling to set the TerminationTime to the given value or greater, then the RegisterPublisher request MUST fault. If the value is not “in the future” relative to the current time as known by the NotificationBroker, the RegisterPublisher request MUST fault. The use of the xsi:nil attribute with value “true” indicates there is no scheduled termination time requested for the RegisterPublisher. If the element does not include the time zone designation, the value of the element MUST be interpreted as universal time (UTC) time.

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

If a /wsbn:Topic component is included in the message, the NotificationBroker MUST register the Web service specified by the /wsbn:PublisherReference component as a Publisher on the set of Topics identified by the /wsbn:Topic component.

As part of the processing of a RegisterPublisher request, the NotificationBroker creates a PublisherRegistration resource representing the registration. A new resource is created regardless of whether the same Publisher has previously registered with the NotificationBroker. The NotificationBroker returns an EndpointReference in the response to the RegisterPublisher request. This EndpointReference is a WS-Resource-Qualified EndpointReference as defined in [State Paper] and includes the address of a PublisherRegistrationManager service and a reference property identifying the PublisherRegistration resource.

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

```
…
<wsbn:RegisterPublisherResponse>
   <wsbn:PublisherRegistrationReference>
      <wsa:Address>
         Address of PublisherRegistration Manager
      </wsa:Address>
      <wsa:ReferenceProperties>
          PublisherRegistration Identifier
      </wsa:ReferenceProperties>
      …
   </wsbn:PublisherRegistrationReference>
</wsbn:RegisterPublisherResponse>
…
```

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

/wsbn:PublisherRegistrationReference

A WS-Resource-Qualified EndpointReference to the PublisherRegistration WS-Resource created by the RegisterPublisher request message.

Instead of the RegisterPublisherResponse message, the NotificationBroker may also send the following faults in response to a RegisterPublisher request message:

- Invalid TopicExpression
- Given TopicExpression did not match any Topic supported by the NotificationBroker
- Publisher registration failed
- Others, TBD

5.4.1 Example SOAP Encoding of the RegisterPublisher Message Exchange

The following is a non-normative example of a RegisterPublisher response message using SOAP 1.2 [SOAP 1.2]:

```
<s12:Envelope
    xmlns:s12="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsa="http://schemas.xmlsoap.org/ws/2003/03/addressing"
    xmlns:wsbn=
      "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-
BrokeredNotification-1.2-draft-01.xsd"
    xmlns:npex="http://www.producer.org/RefProp">
  <s12:Header>
    <wsa:Action>
        http://docs.oasis-open.org/wsn/2004/06/WS-BrokeredNotification
/RegisterPublisher
    </wsa:Action>
    <wsa:To s12:mustUnderstand="1">
        http://www.producer.org/NotificationBrokerEndpoint
    </wsa:To>
  </s12:Header>
  <s12:Body>
```

```
    <wsbn:RegisterPublisher>
      <wsbn:PublisherReference>
         <wsa:Address>
            http://www.producer.org/ProducerEndpoint
         </wsa:Address>
         <wsa:ReferenceProperties>
           <npex: NPResourceDisambiguator>
               uuid:84decd55-7d3f-65ad-ac44-675d9fce5d22
           </npex: NPResourceDisambiguator>
         </wsa:ReferenceProperties>
      </wsbn:PublisherReference>
      <wsbn:Topic  dialect="http://docs.oasis-open.org/wsn/2004/06/WS-
Topics/SimpleTopicExpression">
          npex:SomeTopic
      </wsbn:Topic>
      <wsbn:Demand>
         true
      </wsbn:Demand>
      <wsbn:InitialTerminationTime>
         2003-12-25T00:00:00.00000Z
      </wsbn:InitialTerminationTime>
    </wsbn:RegisterPublisher>
  </s12:Body>
</s12:Envelope>
```

The following is a non-normative example of a RegisterPublisher response message using SOAP 1.2 [SOAP 1.2]:

```
<s12:Envelope
    xmlns:s12="http://www.w3.org/2003/06/soap-envelope"
    xmlns:wsa="http://schemas.xmlsoap.org/ws/2003/03/addressing"
    xmlns:wsbn=
      "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-
BrokeredNotification-1.2-draft-01.xsd"
    xmlns:npex="http://www.consumer.org/RefProp">
  <s12:Header>
    <wsa:Action>
        http://docs.oasis-open.org/wsn/2004/06/WS-BrokeredNotification
/RegisterPublisherResponse
    </wsa:Action>
    <wsa:To s12:mustUnderstand="1">
        http://www.publisher.org/PublisherEndpoint
    </wsa:To>
    <pubex:PublisherId xmlns:pubex="http://www.publisher.org/RP">
        Fred
    </pubex:PublisherId>
  </s12:Header>
  <s12:Body>
    <wsbn:RegisterPublisherResponse>
      <wsbn:PublisherRegistrationReference>
        <wsa:Address>
           http://www.producer.org/PublisherEndpoint
        </wsa:Address>
        <wsa:ReferenceProperties>
          <npex:NPubResourceId>
             uuid:95fefeb3-f37d-5dfe-44fe-221d9fceec99
          </npex:NPubResourceId>
        </wsa:ReferenceProperties>
      </wsbn:PublisherRegistrationReference>
    </wsbn:RegisterPublisherResponse>
  </s12:Body>
</s12:Envelope>
```

6 PublisherRegistrationManager Interface

The PublisherRegistrationManager interface defines message exchanges to manipulate PublisherRegistration resources. The PublisherRegistrationManager follows the implied resource pattern as described in [State Paper].

A PublisherRegistrationManager MUST support the required message exchanges associated with the WS-ResourceProperties specification [WS-ResourceProperties] and MAY support the optional message exchanges defined by WS-ResourceProperties.

The PublisherRegistrationManager MUST support the message exchanges defined for both forms of resource lifetime (immediate and scheduled destruction) by WS-ResourceLifetime [WS-ResourceLifetime]. These message exchanges define the means by which PublisherRegistration resources can be explicitly destroyed, or destroyed using a scheduled (time-based) mechanism.

6.1 PublisherRegistration Resource Properties

In addition to the resource properties required by WS-ResourceLifetime, the PublisherRegistration resource MUST include a reference to each of the following resource property elements in its resource properties document:

```
…
  <xsd:element name=”PublisherReference”
               type=”wsa:EndpointReference”
               minOccurs=”0” maxOccurs=”1” />
  <xsd:element name=”Topic”
               type=”wsnt:TopicExpressionType”
               minOccurs=”0” maxOccurs=”unbounded” />
  <xsd:element name=”Demand”
               type=”xsd:boolean”
               minOccurs=”1” maxOccurs=”1” />
  <xsd:element name=”CreationTime”
               type=”xsd:dateTime”
               minOccurs=”0” maxOccurs=”1” />

```

These resource property elements are further described as follows:

/wsbn:PublisherReference and /wsbn:Topic and /wsbn:Demand

> These elements are as defined in the description of the RegisterPublisher request message (see 5.4).

/wsbn:CreationTime

> Indicates the date and time at which the PublisherRegistration was created. This is an optional component, supporting resource constrained devices which cannot associate a creation time with PublisherRegistration resources they create.

The following resource properties MAY be modified by the requestor, by sending a SetResourceProperties request message as defined in the WS-ResourceProperties specification:

- /wsbn:TopicPathExpression and /wsbn:Demand
    - Note: /wsbn:Demand may not take the value “true” if there is no /wsbn:PublisherReference resource property element in the resource property document.

7 Security Considerations

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

8 References

[RFC2119] S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, http://www.ietf.org/rfc/rfc2119.txt, IETF RFC 2119, March 1997.

[WS-BaseNotification] http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BaseNotification-1.2-draft-03.pdf

[WS-Notification WhitePaper] http://www.oasis-open.org/committees/download.php/6661/WSNpubsub-1-0.pdf

[WS-Security] http://www.oasis-open.org/committees/download.php/5531/oasis-200401-wss-soap-message-security-1.0.pdf

[XML] http://www.w3.org/TR/REC-xml

[XML-Infoset] http://www.w3.org/TR/xml-infoset/

[XPATH] http://www.w3.org/TR/xpath

[SOAP 1.2] http://www.w3.org/TR/soap12-part1/

[WS-Addressing] http://www-106.ibm.com/developerworks/library/specification/ws-add/

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

[State Paper] http://www.oasis-open.org/apps/org/workgroup/wsrf/download.php/6795/ws-modelingresources.pdf

[WS-ResourceLifetime] http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceLifetime-1.2-draft-03.pdf

[WS-ResourceProperties] http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceProperties-1.2-draft-04.pdf

Appendix A. Acknowledgments

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

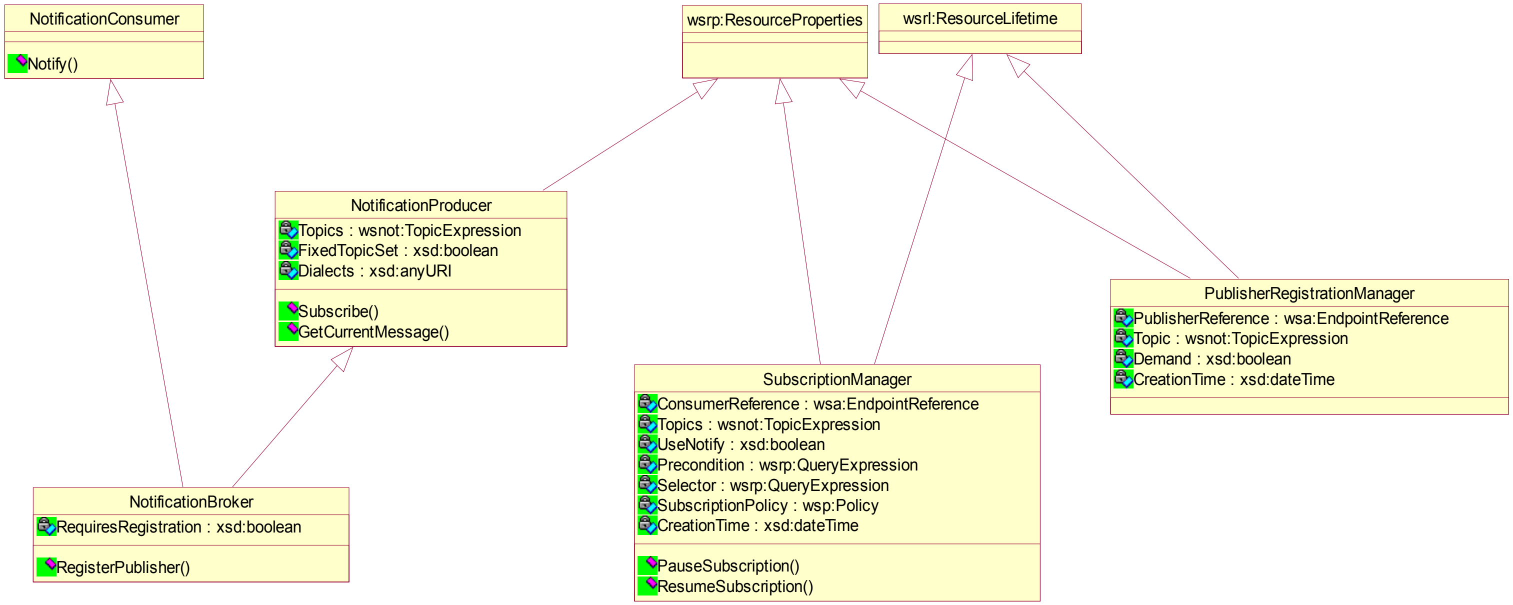
Geoff Acton, LSC Group Ltd, Sid Askary, Individual, Jeff Bohren, OpenNetwork, Fred Carter, AmberPoint, Martin Chapman, Oracle, Dave Chappell, Sonic Software, Ugo Corda, SeeBeyond Technology Corporation, Glen Daniels, Sonic Software, John Fuller, Individual, Stephen Graham, IBM, David Hull, Tibco, Anish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

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

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

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

Appendix B. UML


NotificationConsumer
Notify()
wsrp:ResourceProperties
wsrl:ResourceLifetime
NotificationProducer
Topics : wsnot:TopicExpression
FixedTopicSet : xsd:boolean
Dialects : xsd:anyURI
Subscribe()
GetCurrentMessage()
PublisherRegistrationManager
PublisherReference : wsa:EndpointReference
Topic : wsnot:TopicExpression
Demand : xsd:boolean
CreationTime : xsd:dateTime
SubscriptionManager
ConsumerReference : wsa:EndpointReference
Topics : wsnot:TopicExpression
UseNotify : xsd:boolean
Precondition : wsrp:QueryExpression
Selector : wsrp:QueryExpression
SubscriptionPolicy : wsp:Policy
CreationTime : xsd:dateTime
PauseSubscription()
ResumeSubscription()
NotificationBroker
RequiresRegistration : xsd:boolean
RegisterPublisher()

Appendix C. XML Schema

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

```
<?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). 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.
-->

<xsd:schema
  xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:wsa="http://schemas.xmlsoap.org/ws/2003/03/addressing"
  xmlns:wsbn=
    "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-
BrokeredNotification-1.2-draft-01.xsd"
  xmlns:wsnt=
    "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BaseNotification-
1.2-draft-01.xsd"
  targetNamespace=
    "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-
BrokeredNotification-1.2-draft-01.xsd"
  elementFormDefault="qualified"  attributeFormDefault="unqualified">

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

  <xsd:import namespace=
    "http://schemas.xmlsoap.org/ws/2003/03/addressing"
              schemaLocation=
    "http://schemas.xmlsoap.org/ws/2003/03/addressing"
  />

  <xsd:import namespace=
  "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BaseNotification-1.2-
draft-01.xsd"
              schemaLocation=
  "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BaseNotification-1.2-
draft-01.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="Topic"
                      type="wsnt:TopicExpressionType" />
         <xsd:element name="Demand"
                      type="xsd:boolean" />
         <xsd:element name="CreationTime"
                      type="xsd:dateTime" />
</xsd:schema>
```

Appendix D. WSDL 1.1

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

```

<?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). 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.
-->
<wsdl:definitions name="WS-BrokeredNotification"
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:wsa="http://schemas.xmlsoap.org/ws/2003/03/addressing"
  xmlns:wsbn=
 "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-
BrokeredNotification-1.2-draft-01.xsd"
  xmlns:wsnt=
 "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-
BaseNotification-1.2-draft-01.xsd"
  xmlns:wsbf=
   "http://docs.oasis-open.org/wsn/2004/06/wsrf-WS-BaseFaults-1.2-
draft-01.xsd"
  xmlns:wsrp=
 "http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
ResourceProperties-1.2-draft-01.xsd"
  xmlns:wsrl=
 "http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
ResourceLifetime-1.2-draft-01.xsd"
  targetNamespace=
 "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-
BrokeredNotification-1.2-draft-01.wsdl">

<!-- ========================= Imports ========================== -->
   <wsdl:import
       namespace=
  "http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
ResourceProperties-1.2-draft-01.wsdl"
       location=
"http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
ResourceProperties-1.2-draft-01.wsdl" />

   <wsdl:import
       namespace=
  "http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
ResourceLifetime-1.2-draft-01.wsdl"
       location=
"http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
ResourceLifetime-1.2-draft-01.wsdl" />

   <wsdl:import
       namespace=
  "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-
BaseNotification-1.2-draft-01.wsdl"
       location=
"http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BaseNotification-
1.2-draft-01.wsdl" />

<!-- ==================== Types Definitions ====================== -->
   <wsdl:types>
      <xsd:schema
        targetNamespace=
  "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-
BrokeredNotification-1.2-draft-01.xsd">

         <xsd:include schemaLocation=
 "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-
BrokeredNotification-1.2-draft-01.xsd" />

         <xsd:import
            namespace=
            "http://schemas.xmlsoap.org/ws/2003/03/addressing"
            schemaLocation=
            "http://schemas.xmlsoap.org/ws/2003/03/addressing"
         />

         <xsd:import namespace=
  "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BaseNotification-1.2-draft-01.xsd"
              schemaLocation=
  "http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BaseNotification-1.2-draft-01.xsd"
         />

       <xsd:import
          namespace=
       "http://docs.oasis-open.org/wsn/2004/06/wsrf-WS-BaseFaults-1.2-draft-01.xsd"
          schemaLocation=
  "http://docs.oasis-open.org/wsn/2004/06/wsrf-WS-BaseFaults-1.2-draft-01.xsd"
       />

         <xsd:import namespace=
  "http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceProperties-1.2-draft-01.xsd"
              schemaLocation=
  "http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceProperties-1.2-draft-01.xsd"
         />

         <xsd:import
            namespace=
  "http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceLifetime-1.2-draft-01.xsd"
            schemaLocation=
  "http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceLifetime-1.2-draft-01.xsd"
         />

<!-- =============== Resource Property Related  =================== -->

<!-- ======== Resource Properties for NotificationBroker ========== -->
         <xsd:element name="NotificationBrokerRP" >
            <xsd:complexType>
               <xsd:sequence>
         <!-- From NotificationProducer -->
                  <xsd:element ref="wsnt:Topic"
                               minOccurs="1" maxOccurs="unbounded" />
                  <xsd:element ref="wsnt:FixedTopicSet"
                               minOccurs="1" maxOccurs="1" />
                  <xsd:element ref="wsnt:TopicExpressionDialects"
                               minOccurs="1" maxOccurs="unbounded" />
         <!-- NotificationBroker specific  -->
                  <xsd:element ref="wsbn:RequiresRegistration"
                               minOccurs="1" maxOccurs="1" />
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>

<!-- ====== Resource Properties for PublisherRegistration ========= -->
         <xsd:element name="PublisherRegistrationRP" >
            <xsd:complexType>
               <xsd:sequence>
         <!-- From WS-ResourceLifetime ScheduledResourceTermination -->
                  <xsd:element ref="wsrl:CurrentTime"
```

```
                               minOccurs="1" maxOccurs="1" />
                  <xsd:element ref="wsrl:TerminationTime"
                               minOccurs="1" maxOccurs="1" />

         <!-- PublisherRegistration specific  -->
                  <xsd:element ref="wsbn:PublisherReference"
                               minOccurs="0" maxOccurs="1" />
                  <xsd:element ref="wsbn:Topic"
                               minOccurs="0" maxOccurs="unbounded" />
                  <xsd:element ref="wsbn:Demand"
                               minOccurs="1" maxOccurs="1" />
                  <xsd:element ref="wsbn:CreationTime"
                               minOccurs="0" maxOccurs="1" />
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>

<!-- ========== Message Types for NotificationBroker  ============ -->
         <xsd:element name="RegisterPublisher">
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element name="PublisherReference"
                               type="wsa:EndpointReferenceType"
                               minOccurs="0" maxOccurs="1" />
                  <xsd:element name="Topic"
                               type="wsnt:TopicExpressionType"
                               minOccurs="0" maxOccurs="unbounded" />
                  <xsd:element name="Demand"
                               type="xsd:boolean" default="false"
                               minOccurs="0" maxOccurs="1" />
                  <xsd:element name="InitialTerminationTime"
                               type="xsd:dateTime"
                               minOccurs="0" maxOccurs="1" />
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>

         <xsd:element name="RegisterPublisherResponse">
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element name="PublisherRegistrationReference"
                               type="wsa:EndpointReferenceType"
                               minOccurs="0" maxOccurs="1" />
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>

         <xsd:complexType name="ResourceUnknownFaultType">
            <xsd:complexContent>
               <xsd:extension base="wsbf:BaseFaultType"/>
            </xsd:complexContent>
         </xsd:complexType>
         <xsd:element name="ResourceUnknownFault"
                      type="wsbn:ResourceUnknownFaultType"/>

         <xsd:complexType name="InvalidTopicExpressionFaultType">
            <xsd:complexContent>
               <xsd:extension base="wsbf:BaseFaultType"/>
            </xsd:complexContent>
         </xsd:complexType>
         <xsd:element name="InvalidTopicExpressionFault"
                      type="wsbn:InvalidTopicExpressionFaultType"/>

         <xsd:complexType name="TopicNotSupportedFaultType">
            <xsd:complexContent>
               <xsd:extension base="wsbf:BaseFaultType"/>
            </xsd:complexContent>
         </xsd:complexType>
         <xsd:element name="TopicNotSupportedFault"
                      type="wsbn:TopicNotSupportedFaultType"/>

         <xsd:complexType name="PublisherRegistrationFailedFaultType">
            <xsd:complexContent>
               <xsd:extension base="wsbf:BaseFaultType"/>
            </xsd:complexContent>
         </xsd:complexType>
         <xsd:element name="PublisherRegistrationFailedFault"

type="wsbn:PublisherRegistrationFailedFaultType"/>

      </xsd:schema>
   </wsdl:types>

<!-- ========== NotificationBroker::RegisterPublisher =============
   RegisterPublisher(PublisherReference, TopicExpression* ,
                     [Demand], [InitialTerminationTime])
   returns: WS-Resource qualified EPR to a PublisherRegistration -->
   <wsdl:message name="RegisterPublisherRequest">
      <wsdl:part name="RegisterPublisherRequest"
                 element="wsbn:RegisterPublisher"/>
   </wsdl:message>

   <wsdl:message name="RegisterPublisherResponse">
      <wsdl:part name="RegisterPublisherResponse"
                 element="wsbn:RegisterPublisherResponse"/>
   </wsdl:message>

   <wsdl:message name="ResourceUnknownFault">
      <part name="ResourceUnknownFault"
            element="wsbn:ResourceUnknownFault" />
   </wsdl:message>

   <wsdl:message name="InvalidTopicExpressionFault">
      <part name="InvalidTopicExpressionFault"
            element="wsbn:InvalidTopicExpressionFault" />
   </wsdl:message>

   <wsdl:message name="TopicNotSupportedFault">
      <part name="TopicNotSupportedFault"
            element="wsbn:TopicNotSupportedFault" />
   </wsdl:message>

   <wsdl:message name="PublisherRegistrationFailedFault">
      <part name="PublisherRegistrationFailedFault"
            element="wsbn:PublisherRegistrationFailedFault" />
   </wsdl:message>

<!-- ================== PortType Definitions ==================== -->
<!-- ========== NotificationBroker PortType Definition ============ -->
   <wsdl:portType name="NotificationBroker"
      wsrp:ResourceProperties ="wsbn:NotificationBrokerRP">
      <!-- ============= extends NotificationConsumer ============= -->
      <wsdl:operation name="Notify">
         <wsdl:input message="wsnt:Notify" />
      </wsdl:operation>

      <!-- ========== extends wsrp:ResourceProperties ============= -->
      <wsdl:operation name="GetResourceProperty">
         <wsdl:input  name="GetResourcePropertyRequest"
                      message="wsrp:GetResourcePropertyRequest" />
         <wsdl:output name="GetResourcePropertyResponse"
                      message="wsrp:GetResourcePropertyResponse" />
         <wsdl:fault  name="ResourceUnknownFault"
                      message="wsrp:ResourceUnknownFault" />
         <wsdl:fault  name="InvalidResourcePropertyQNameFault"
                      message="wsrp:InvalidResourcePropertyQNameFault"
/>
      </wsdl:operation>

      <!-- ============= extends NotificationProducer ============= -->
      <wsdl:operation name="Subscribe">
         <wsdl:input  message="wsnt:SubscribeRequest" />
         <wsdl:output message="wsnt:SubscribeResponse" />
         <wsdl:fault  name="ResourceUnknownFault"
                      message="wsnt:ResourceUnknownFault" />
         <wsdl:fault  name="SubscribeCreationFailedFault"
                      message="wsnt:SubscribeCreationFailedFault"/>
      </wsdl:operation>

      <wsdl:operation name="GetCurrentMessage">
         <wsdl:input  message="wsnt:GetCurrentMessageRequest"/>
         <wsdl:output message="wsnt:GetCurrentMessageResponse"/>
         <wsdl:fault  name="ResourceUnknownFault"
                      message="wsnt:ResourceUnknownFault" />
         <wsdl:fault  name="InvalidTopicExpressionFault"
                      message="wsnt:InvalidTopicExpressionFault" />
         <wsdl:fault  name="TopicNotSupportedFault"
                      message="wsnt:TopicNotSupportedFault" />
         <wsdl:fault  name="NoCurrentMessageOnTopicFault"
                      message="wsnt:NoCurrentMessageOnTopicFault" />
      </wsdl:operation>

      <!-- ========= NotificationBroker specific operations ======= -->
      <wsdl:operation name="RegisterPublisher">
         <wsdl:input  message="wsbn:RegisterPublisherRequest"/>
         <wsdl:output message="wsbn:RegisterPublisherResponse"/>
         <wsdl:fault  name="ResourceUnknownFault"
                      message="wsbn:ResourceUnknownFault" />
         <wsdl:fault  name="InvalidTopicExpressionFault"
                      message="wsbn:InvalidTopicExpressionFault"/>
         <wsdl:fault  name="TopicNotSupportedFault"
                      message="wsbn:TopicNotSupportedFault"/>
         <wsdl:fault  name="PublisherRegistrationFailedFault"
                      message="wsbn:PublisherRegistrationFailedFault"/>
       </wsdl:operation>
   </wsdl:portType>

<!-- ===== PublisherRegistrationManager PortType Definition ====== -->
   <wsdl:portType name="PublisherRegistrationManager"
      wsrp:ResourceProperties ="wsbn:PublisherRegistrationManagerRP">
      <!-- ========== extends wsrp:ResourceProperties ============= -->
      <wsdl:operation name="GetResourceProperty">
         <wsdl:input  name="GetResourcePropertyRequest"
                      message="wsrp:GetResourcePropertyRequest" />
         <wsdl:output name="GetResourcePropertyResponse"
                      message="wsrp:GetResourcePropertyResponse" />
         <wsdl:fault  name="ResourceUnknownFault"
                      message="wsrp:ResourceUnknownFault" />
         <wsdl:fault  name="InvalidResourcePropertyQNameFault"
                      message="wsrp:InvalidResourcePropertyQNameFault"
/>
      </wsdl:operation>

      <!-- === extends wsrl:ImmediateResourceTermination ========== -->
      <wsdl:operation name="Destroy">
         <wsdl:input message="wsrl:DestroyRequest" />
         <wsdl:output message="wsrl:DestroyResponse" />
         <wsdl:fault name="ResourceUnknownFault"
                     message="wsrl:ResourceUnknownFault" />
         <wsdl:fault name="ResourceNotDestroyedFault"
                     message="wsrl:ResourceNotDestroyedFault" />
      </wsdl:operation>

      <!-- === extends wsrl:ScheduledResourceTermination ========== -->
      <wsdl:operation name="SetTerminationTime">
         <wsdl:input message="wsrl:SetTerminationTimeRequest" />
         <wsdl:output message="wsrl:SetTerminationTimeResponse" />
         <wsdl:fault name="ResourceUnknownFault"
                     message="wsrl:ResourceUnknownFault" />
         <wsdl:fault name="UnableToSetTerminationTimeFault"
                     message="wsrl:UnableToSetTerminationTimeFault" />
         <wsdl:fault name="TerminationTimeChangeRejectedFault"
                     message="wsrl:TerminationTimeChangeRejectedFault"
/>
      </wsdl:operation>
      <!-- ==== PublisherRegistrationManager specific operations == -->
   </wsdl:portType>
</wsdl:definitions>
```

1028 # Appendix E. Revision History

Rev	Date	By Whom	What
Ws-bn-01	2004-05-12	Lily Liu	Initial version
Ws-bn-02	2004-06-07	Dave Chappell	Updates and consistency check w/ other WS-N specs
Ws-bn-03	2004-06-24	Lily Liu, Dave Chappell	Addition of a Goals and Requirements section and minor format changes
Ws-bn-03	2004-07-12	Lily Liu	Addition of a status paragraph
Ws-bn-01	2004-07-20	Lily Liu	Changed the version back to draft 1. Fixed the URLs and references.

Appendix F. Notices

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 implementers 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.

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.