An Analysis of Notification Related Specifications for Web/Grid applications

Shrideep Pallickara and Geoffrey Fox
(spallick, gcf)@indiana.edu
Community Grids Laboratory, Indiana University

Abstract

Notification is especially important in the Service Oriented Architecture (SOA) model engendered by Web Services. where Web Services interact with each other through the exchange of messages. In this paper we compare and contrast two competing specifications in the area of notifications. The first one, WS-Notification, is part of the Web Service Resource Framework (WSRF). The second one is the WS-Eventing specification. These specifications are expected to have far reaching implications on the development of asynchronous, complex, dynamic systems.

Keywords: notifications, publish/subscribe, middleware systems, Web Services, Grid Services, WSRF

1. Introduction

Messaging is a fundamental primitive in distributed systems. Entities communicate with each other through the exchange of messages, which can encapsulate information of interest such as application data, errors and faults, system conditions, search and discovery of resources. A related concept is that of *notifications* where entities receive messages based on their registered interest in certain occurrences or situations. Messaging and notifications are especially important in the Service Oriented Architecture (SOA) model engendered by Web Services. Here, Web Services interact with each other through the exchange of messages.

In this paper we compare and contrast two competing specifications in the area of notifications. The first one, WS-Notification [1], is part of the Web Service Resource Framework (WSRF) [2]. WSRF is a realignment of the dominant Open Grid Service Infrastructure [3, 4] to be more in line with the emerging consensus [5] within the Web Services community. The second one is the WS-Eventing [6] specification from IBM and Microsoft. These specifications are expected to have far reaching implications on the development of asynchronous, loosely-coupled, dynamic systems.

2. Comparing the specifications

WS-Notification is a complex specification comprising three other specifications viz. WS-BaseNotification, WS-BrokeredNotification and WS-Topics. Furthermore several elements (such as subscriptions and topic spaces) are also resources (WS-Resource) as outlined in the WSRF suite of specifications. In their role as resources these aforementioned elements also need to support inspection and modification of the associated properties and lifetimes as outlined in the WS-ResourceProperties and the WS-ResourceLifetime specifications respectively. WS-Eventing on the other hand is a self-contained specification.

WS-Notification provides support for both a *Notify* message as well as *raw* application-specific messages. The Notify message type also encapsulates topic information within them. This is especially useful in allowing a consumer to identify the sub-processes responsible for dealing with specific topics. The WS-BrokeredNotification specification also provides support for loosely-coupled interactions since a publisher need not keep track of all its consumers. WS-Eventing on the other hand provides support only for raw application specific messages. WS-Eventing notifications do not encapsulate any topic information within them.

WS-Notification currently only outlines the push delivery mode for notifications. The push model is one in which notifications are pushed to the consumer. WS-Notification however incorporates support for delegated (or brokered) delivery of notifications. WS-Eventing also outlines the push model for notifications. A related specification from Microsoft and Intel, WS-Management [7] outlines three other modes for delivery: *batched*, *pull* and *trap*. The batched mode allows an event source to batch multiple notifications into a single SOAP envelope. In the pull mode a sink is responsible for polling the source at regular intervals and pulling notifications if any are available. Finally, the trap mode leverages the SOAP over UDP specification and indicates that the sink is interested in receiving notifications over UDP.

Both specifications provide support for delegated management of subscriptions through the Subscription Manager interface. Furthermore, the specifications also allow the specification of XPath constraints to *filter* notifications. In WS-Notification the subscription related operations include *subscribe*, *pause* and *resume*. Pause and resume relate to the ability to suppress receipt of notifications in the intervening period between them. WS-Notification also includes support for retrieving the last message that was published by a publisher on a given topic. The specification also allows consumers to modify their termination times. It should be noted that there is no operation for unsubscribe. Instead, the WS-Notification specification expects consumers to adjust the time for expiration of the subscription resource as governed by the WS-ResourceLifetime specification. This is a problematic issue since an unsubscribe operation is semantically different from the expiry of a subscription. In WS-Notification there is also no exchange which announces the end of a subscription to a consumer. In WS-Eventing the subscription related operations include *subscribe*, *renew*, *unsubscribe* and *subscription-end*. The renew operation relates to the ability to extend the lifetime of a subscription. A sink receives a Subscription End notification either as a result of the lifetime expiring or an unsubscribe operation. Though the WS-Eventing specification does not support the pause-renew set of operations, the WS-Management specification facilitates this operation. There is no separate message in WS-Eventing to retrieve the last message published by a source, though this is not really needed if one has the pause-resume feature from WS-Management.

WS-Notification includes a separate specification, WS-Topics, which deals with the management of a topic space. The topic space facilitates hierarchical organization of topics and supports two wildcard operators, ***** and //, for the selection of topics within a topic tree. In WS-Notification consumers can inspect the topics available at a producer through the Notification Producer interface. In WS-Eventing there is no formal specification regarding the management of topics.

In WS-Notification a publisher need not keep track of all the subscriptions or the routing of events to consumers. This task is performed by the broker intermediary. In WS-Eventing the source keeps track of all sinks, and is responsible for routing notifications to the sinks.

3. Federation between the specifications

We believe that it is possible that these specifications might be deployed concurrently. Federation between these specifications will allow endpoints in these specifications to interact with each other. This would involve mapping the semantics of operations involved in these specifications. These operations need to be managed by a *middleware*. Here we briefly review some of the key issues involved. First, the operations related to subscriptions need to be mapped. Here, the requests to unsubscribe and to renew subscriptions in WS-Eventing should be mapped into the appropriate calls using WS-ResourceLifetime if needed. Second, the middleware also needs to maintain a list of properties that are automatically generated. This would enable WS-Eventing components to behave as WS-Resources that facilitate inspection of properties. Delivery modes supported in either specifications need to be mapped appropriately. Issues pertaining to pausing and renewing of subscriptions need to be handled by the federation module by appropriately keeping track of issued notifications.

4. Conclusions

In this paper we have analyzed and contrasted the two dominant specifications in the area of Web Services notifications. Depending on the needs of the application deployments can choose to leverage either of these specifications.

5. References

[1] Web Services Notification (WS-Notification). IBM, Globus, Akamai et al. http://www-106.ibm.com/ developerworks/library/specification/ws-notification/

[2] The Web Services Resource Framework. http://www.globus.org/wsrf/

[3] I. Foster, C. Kesselman, J. Nick, S. Tuecke, "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration.", Global Grid Forum, June 22, 2002. Available from http://www.globus.org/research/papers/ogsa.pdf.

[4] The Open Grid Services Infrastructure (OGSI). http://www.gridforum.org/Meetings/ggf7/drafts/draft-ggf-ogsi-gridservice-23_2003-02-17.pdf

[5] D. Booth, et al, "Web Services Architecture." W3C Working Group Note 11 February 2004. http:// www.w3.org/TR/2004/NOTE-ws-arch-20040211/.

[6] Web Services Eventing. Microsoft, IBM.. http: //ftpna2.bea.com/pub/downloads/WS-Eventing.pdf

[7] Web Services Management. Microsoft, Intel et al. http://msdn.microsoft.com/library/en-us/dnglobspec/html/ws-management.pdf