Reliable Message Delivery in a Web Services World:

A Proposed Architecture and Roadmap

A Joint White Paper from IBM Corporation and Microsoft Corporation

March 13, 2003

Version 1.0

Executive Summary

This white paper provides an overview and roadmap for message formats, protocols and service interfaces that provide reliable message delivery for Web services. The paper describes the basic architectural approach and provides a scenario-based introduction to the core WS-Reliable Messaging protocol and related specifications. Some of the related specifications are already public, while other topics represent areas in which additional, open industry activity is necessary.

Introduction

The information technology industry has been using Web services for over three years. Numerous documents describe the business and technical benefits of Web services. Many companies are using Web services in production environments. These customer scenarios demonstrate the practical achievement of Web services objectives.

Customers, industry analysts, systems vendors, and the trade press all identify a key area that remains to be addressed: reliable message delivery. The reliable delivery of messages is seen as crucial for Web services to become the primary infrastructure for heterogeneous interconnection of business processes, systems and products.

Reliable messaging is critical for Web services. It is not possible to solve many business problems if the participants cannot be sure of the completion of message exchanges. Without a Web services standard for providing reliable message delivery, applications will implement the necessary function in their business logic. This requirement places a burden on developers of business logic, but more importantly impedes interoperability because of inconsistent, differing solutions to a common problem.

Finally, a reliable message delivery standard will improve the effectiveness of other Web services standards, like security, transactions and business processes. These improvements are only possible if reliable message delivery is a standard, and not embedded in business logic. Reliable message delivery ensures that Web services architecture, protocols and interfaces deliver secure, interoperable, transactional and robust solutions.

IBM, Microsoft and our partners are responding to the industry's need by defining an architecture, which contains protocols and interfaces, for reliable message delivery. The architecture is simple, effective and supports composition with other standards to provide the functions that customers, software vendors and our industry need. The architecture is also extensible, and we describe an industry roadmap for evolving and composing the

standards, which we expect to occur through our joint work and broad participation of our industry.

In this document we describe a set of Web service-based specifications that provide for reliable message delivery. These specifications build on XML, SOAP, WSDL, WS-Policy and WS-Security. The architecture also supports composition with WS-Transactions, WS-Coordination and BPEL4WS to define a robust, secure, reliable and transactional Web services infrastructure.

Our design for reliable messaging is modular and supports incremental use of its capabilities. This modularity combined with flexible composition with other Web services standards allows companies to develop flexible solutions that exactly meet their business needs.

IBM, Microsoft and our partners intend to work with customers, partners and standards bodies to evolve this body of specifications, and to define and compose additional specifications. Some examples of enhanced integration might be with policy, trust, privacy, transactions, BPEL4WS and reliable messaging. To ensure interoperability and consistent implementation of the various proposed specifications described in this paper, IBM, Microsoft, and our partners will work closely with standards organizations, the developer community, and with industry organizations such as WS-I.org to develop interoperability profiles and tests that will provide guidance to tool vendors.

Technical Overview

A cornerstone of the architecture we define in this paper is the core reliable messaging specification: the Web Services Reliable Messaging (WS-ReliableMessaging) protocol. WS-ReliableMessaging defines the facilities for ensuring efficient, asynchronous reliable message delivery. The architecture of WS-ReliableMessaging supports composition with other messaging and Web service specifications and standards. This architecture consists of the following specifications:

- WS-ReliableMessaging – A protocol that allows messages to be delivered reliably between distributed applications in the presence of software component, system, or network failures.
- WS-Addressing – A framework to identify Web service endpoints and to ensure end-to-end endpoint identification in messages.
- WSDL – A set of constructs to specify Web service interfaces and bindings for endpoints.
- WS-Policy – A base set of constructs that can be used and extended by other Web services specifications to describe a broad range of requirements, preferences, and capabilities of service interfaces.
- WS-Transactions and WS-Coordination – A set of Web service interface definitions and protocols that support participant control and agreement on the outcome of distributed, multi-party interactions.
- WS-EndpointResolution – A set of Web service mechanisms that support selecting a specific endpoint for an operation or message from a set of allowed candidates. This is particularly useful in server farms and mobile environments.
- WS-MetadataExchange – A set of Web service mechanisms to exchange policies, WSDL and potentially other metadata between two or more parties.

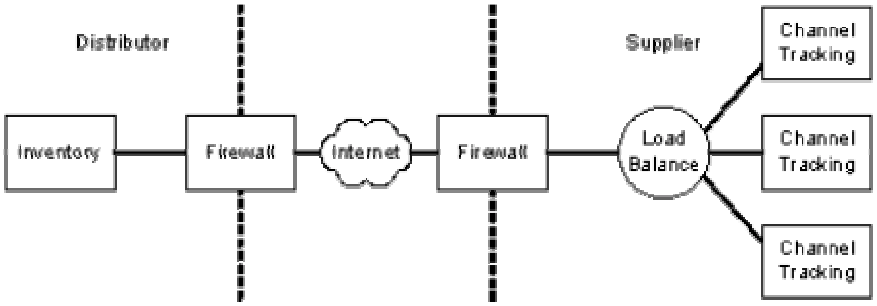
- The WS-Security Roadmap protocols – In April 2002, IBM and Microsoft published a roadmap for Web service security [WS-Security Roadmap] that supports, integrates and unifies several popular security models, mechanisms, and technologies.
- WS-TransmissionControl – A set of constructs for controlling the exchange of messages between services to improve reliability by preventing message loss due to service unavailability, overloading queues and other causes.

WSDL, WS-Security, WS-Policy, WS-Transactions, and related documents have been published by IBM, Microsoft and others and are submitted to or on track to standards bodies. We are publishing WS-Addressing and WS-ReliableMessaging with this summary. WS-MetadataExchange, WS-EndpointResolution and WS-TransmissionControl represent new areas in which IBM and Microsoft, working together with our customers, partners, and standards organizations, are investigating possible and practical solutions.

A Typical Scenario

To make our approach to reliable messaging as concrete as possible, we present the architecture through a common scenario that reflects the current and anticipated application of Web services. This scenario illustrates some of the common obstacles and design tensions that are inherent in today's Web service world.

The scenario is based on a distributor who wants to transfer inventory and account information to a supplier every night using Web services over the Internet.



The distributor has a basic inventory control system that runs the business processes for inventory tracking and management. The supplier has an array of three channel tracking systems. The supplier uses a network load-balancing product to distribute messages to the three servers. Both organizations use various networking products (proxies, firewalls, etc.) to connect their internal networks to the public Internet.

The information the parties exchange is critical to their business operations. They both require reliable and secure data exchange.

Making Message Exchange Between Web Services Reliable

The current Web services architecture places most of the onus for reliability on the application developer. For example, using today's Web service technologies, the supplier must use an application level convention to prevent duplicate message processing. The distributor's system needs to implement application-level mechanisms to ensure that the supplier's systems process request messages. The supplier's and distributor's business problems may also require their applications to implement a convention to ensure that a receiving system processes messages in the same order in which they were sent. Ad hoc,

application-level solutions to reliable message delivery may work. Their use, however, introduces two future problems:

- The *ad hoc*, application-level mechanisms may increase costs to the supplier or distributor as they expand their business to work with other partners. These other partners may have embedded different, incompatible reliable messaging functions in their applications.
- The distributor and supplier may not be able to easily improve the robustness of their solutions by adding standard Web service support for transactions or security or to expand the set of supported security mechanisms. The application-level reliable messaging solution may not support composition or addition of standard Web protocols or interfaces.

WS-ReliableMessaging provides simple, systematic solutions to the supplier's and distributor's message reliability requirements, and supports their future interoperation with other partners and use of additional Web service standards. Like previous Web service standards, WS-ReliableMessaging also allows software vendors to provide an implementation of the standard in a common reusable software component, which supports applications and frees application developers from the burden of implementing the functions in their applications. Additionally, middleware vendors can use WS-ReliableMessaging as an interoperability protocol to bridge their messaging environment to other vendor's middleware environments.

The remainder of this section looks at specific issues in our scenario and demonstrates how WS-ReliableMessaging and related specifications will resolve them. Specifically, we examine:

- Dealing with load balancing and cross-organization messages using WS-Addressing
- Dealing with unreliable message delivery using WS-ReliableMessaging
- Dealing with disparate system capabilities and preferences using WS-MetadataExchange
- Dealing with message confidentiality and authentication using WS-Security, WS-Trust, and WS-SecureConversation
- Using WS-TransmissionControl to control message flows to prevent message loss due to service availability, fixed queuing resources or other factors.

Addressing Messages Over the Internet and Load Balancing

The distributor and supplier have an agreement to exchange information every evening after normal business hours. In order to handle the load of thousands of distributors, the supplier has deployed their channel tracking application on a server farm that sits behind a network load balancer that acts as a front-end to the farm.

To achieve the maximum flexibility, the supplier does not want to commit specific machines in the farm to specific distributors. For that reason, the supplier exposes the inventory control system as a Web service that resides at the address of the network load balancer. However, because many of the messages sent by each specific distributor will work on common data and often need a transaction and/or security context, the supplier's overall site performance would benefit greatly by dispatching these messages to the same machine in the server farm.

To address these issues, the WS-Addressing specification provides a flexible and extensible mechanism for addressing messages and Web services called an *endpoint reference*. Endpoint references allow applications to augment a traditional URI-based address with application-specific information that can be used to better dispatch the message. In our example, the distributor starts its conversation using an endpoint reference that refers only to the network load balancer.

Each evening, the distributor's inventory control system begins its conversation using this endpoint reference. As the conversation is established, the supplier's system provides a more refined endpoint reference that is unique to the host machine best suited to the particular distributor. This refined endpoint reference will contain information needed by the network load balancer to relay the message to the appropriate machine.

The use of the more specific endpoint reference is typically transparent to the supplier's and distributor's applications. Infrastructure software implementing the Web services protocols supports the transparent insertion and processing of the fully-refined endpoint reference.

IBM and Microsoft will work together with our customers, partners, and standards organizations, to investigate possible and practical solutions to WS-EndpointResolution.

Exchanging Messages Reliably

Messages exchanged between the distributor and supplier travel over multiple nodes, some on the public Internet. This means that some messages may be lost in transit. Additionally, it is possible that either the supplier's or the distributor's system may fail while a message is in transit, leaving the still-running system in a state of confusion as to whether a given message has been processed or not.

Our architecture resolves these issues with the WS-ReliableMessaging protocol. WS-ReliableMessaging provides guaranteed end-to-end delivery of messages using Web service protocols and standards.

The basic model of WS-ReliableMessaging is fairly simple to understand. Under WS-ReliableMessaging, the source node sends a normal Web service message containing a WS-ReliableMessaging header. Upon receipt of the message, the destination node sends an acknowledgement message back to the source indicating that the message was successfully delivered.

WS-ReliableMessaging takes advantage of the algorithms used in reliable messaging and transaction protocols. Specifically, WS-ReliableMessaging provides a flexible acknowledgement scheme in which receivers can efficiently convey the range of messages that have (and have not) been received. WS-ReliableMessaging also provides an efficient ordering mechanism to ensure that receivers can process messages in the same order in which they were sent, even in the face of reordering due to retransmissions or multi-path routing.

WS-ReliableMessaging was designed to compose with the existing Web service infrastructure. In particular, WS-ReliableMessaging is meant to layer on existing application protocols and interfaces described in WSDL and XML Schema. This means that the

distributor and supplier do not need to redesign their application-level message schemas or exchange patterns. The applications' implementations use the WSDL and XML Schema that define the business interfaces of the services. The infrastructure software providing the Web services protocols compose the reliable messaging functions with the Web services protocols.

WS-ReliableMessaging is not bound to underlying transport protocols or sessions. This means that the lifetime of a WS-ReliableMessaging conversation can span long periods of time (days, weeks) even when one or both systems are rebooted. This allows conversations to be suspended mid-stream (for example, to allow system maintenance) and then resumed without needing to retransmit the entire conversation.

The WS-ReliableMessaging protocol has several features that benefit a large bi-directional transfer such as those typically used in partner integration scenarios such as this one. WS-Addressing allows messages to be sent asynchronously in either direction. WS-ReliableMessaging takes advantage of this capability in several ways. Rather than send an acknowledgement message for every message received, WS-ReliableMessaging allows the destination to cumulatively acknowledge every message it has received in a single, compact control element. This control element can be sent in its own message or included with a subsequent application message that is sent back to the source (for example, a response message in a request/reply conversation). WS-ReliableMessaging does not place any arbitrary restrictions on the number of outstanding messages in transit; two endpoints can have several reliable messages "in flight" in either direction over a single underlying transport connection. In this common case, the message overhead for WS-ReliableMessaging becomes very low.

Establishing Common Parameters

It is common for two autonomous partners such as the supplier and distributor to have differing sets of capabilities and requirements. For example, both the supplier and distributor may have differing requirements or capabilities for the following:

- What time of day the supplier accepts transfers and the maximum number of unacknowledged messages that it supports.
- What security credentials the supplier requires.
- What timeouts to use to control retransmission of unacknowledged messages.
- Whether the supplier accepts transactions.
- What versions of particular specifications the supplier supports.

WS-Policy establishes a framework to advertise a broad range of service requirements, preferences, and capabilities. WS-PolicyAssertions defines the means to make individual statements such as the supplier's requirement for a security token and a preference of X.509 over Kerberos. Assertions can be grouped together into documents and associated with Web services elements as defined in WS-PolicyAttachment, There are many ways that Web service partners can exchange WSDL interface and WS-Policy information. Some will be informal, out-of-band approaches using e-mail, posting documents on a Web site, or in UDDI, etc.

Sometimes, informal approaches are too limiting, and our architecture identifies the need for a WS-MetadataExchange model. We expect this model to define a WSDL interface and message model that Web service clients use to obtain WSDL and WS-Policy information for a service at a URL or endpoint reference.

In our simple scenario, the distributor would use an architected Web service associated with the supplier's endpoint reference to retrieve WSDL and WS-Policy documents defining the supplier's interfaces, and requirements and preferences for WS-ReliableMessaging, WS-Security, etc. A sophisticated supplier could also engage in a refinement of an endpoint reference and pass back to the distributor a new endpoint reference. The endpoint reference might include the policies or the distributor might need to get the policies for the new endpoint.

A distributor might also choose to request and inspect the policy of the supplier and choose alternatives that are consistent with its own policy. The distributor can then send messages with the assurance that all the requirements of the supplier's system are being met, and the correct versions of each protocol are in use.

Establishing a Security Context

One of the core requirements of both the distributor and the supplier is that their messages are appropriately secured. In some cases, each message needs to be authenticated to ensure that the expected source sent them and that only the intended recipient processes them. Since the information is exchanged over the public Internet, both the distributor and supplier are concerned about message confidentiality and integrity.

The WS-Security family of specifications was designed to handle exactly these types of issues. WS-Security provides a foundational mechanism for signing and encrypting messages using industry-standard technologies such as XML Digital Signatures and XML Encryption.

Because the information exchange between the distributor and supplier is a long-running conversation, the performance of WS-Security can be improved through the use of WS-Trust and WS-SecureConversation. WS-SecureConversation allows the distributor's inventory control system to establish a shared security context with the supplier's channel tracking system. This context is established when the conversation is first initiated. That context can then be used to derive session keys that increase the overall security while reducing the overhead of security processing cost for each message.

Advertising System Availability and Improving Message Reliability

The supplier's channel tracking systems are used throughout the business day in both OLTP and OLAP applications, both of which tax much of the available computing resources. Because of the large amounts of data involved in synchronizing with each distributor's inventory control system, the supplier is wary of allowing these large, batch-style jobs running concurrently with tasks needed for the daily operations of the business. The supplier may define a "batch window" in which the applications supporting distributor synchronization execute.

Furthermore, during the window in which the supplier supports distributor synchronization, the supplier's systems may become overloaded and overrun by incoming messages. During the batch window, the supplier systems may implement a flow (transmission) control policy that prevents distributors from overrunning the supplier systems with messages. Over time, it is expected that such policies could also be factored into the context of service level agreements.

IBM and Microsoft will work together with our customers, partners, and standards organizations, to investigate possible and practical solutions to WS-TransmissionControl.

Outcome Agreement

WS-ReliableMessaging and WS-TransmissionControl help ensure the reliable exchange of messages between Web services. In some scenarios, the reliable delivery of messages is not sufficient for successful completion of a distributed interaction. Endpoints may not be able to process a message for many reasons, which can include business semantics and system resource issues. The inability to process a message, even though reliably delivered, may invalidate the results of prior message exchanges. The WS-Coordination and WS-Transaction specifications augment WS-ReliableMessaging to provide solutions to coordinating and agreeing on the outcome of computations which may contain reliable message exchanges.

Summary

As this paper has described, IBM and Microsoft believe that a modular and extensible approach to reliable messaging and Web services offers customers the flexibility and functionality needed to adapt Web services to a wide range of enterprise scenarios. The architecture presented in the paper provides customers with the necessary building blocks to add this functionality to their existing applications and infrastructure with a minimal amount of disruption and IT costs.

Contributors

This document was jointly authored by IBM and Microsoft.

Key contributors include (alphabetically): Don Box, Microsoft; Felipe Cabrera, Microsoft; Erik Christensen, Microsoft; Francisco Curbera, IBM; Don Ferguson, IBM; Christopher Ferris, IBM; Tom Freund, IBM; Jeffrey Frey, IBM; Maryann Hondo, IBM; John Ibbotson, IBM; Chris Kaler, Microsoft; David Langworthy, Microsoft; Frank Leymann, IBM; Rodney Limprecht, Microsoft; Steve Lucco, Microsoft; Steve Millet, Microsoft; Anthony Nadalin, IBM; Henrik Frystyk Nielsen, Microsoft; John Shewchuk, Microsoft; Tony Storey, IBM; Sanjiva Weerawarana, IBM.

References

[WS-Addressing]
"Web Services Addressing", IBM/Microsoft, March 2003

[WS-ReliableMessaging]
"Web Services Reliable Messaging Protocol", IBM/Microsoft, March 2003

[WS-Policy]
"Web Services Policy Framework", IBM/Microsoft/BEA/SAP, December 2002

[WS-PolicyAssertions]
"Web Services Policy Assertions Language", IBM/Microsoft/BEA/SAP, December 2002

[WS-PolicyAttachment]
"Web Services Policy Attachment", IBM/Microsoft/BEA/SAP, December 2002

[WS-Security]
"Web Services Security Language", IBM/Microsoft/VeriSign, April 2002

[WS-Security Roadmap]
"Security In A Web Services World: A Proposed Architecture and Roadmap", IBM/Microsoft, April 2002

[WS-SecureConversation]
"Web Services Secure Conversation Language", IBM/Microsoft/VeriSign/RSA Security, December 2002

[WS-SecurityPolicy]
"Web Services Security Policy Language", IBM/Microsoft/VeriSign/RSA Security, December 2002

[WS-Trust]
"Web Services Trust Language", IBM/Microsoft/VeriSign/RSA Security, December 2002

[XML-Encrypt]
W3C Recommendation, “XML Encryption Syntax and Processing,” 10 December, 2002.

[XML-Signature]
W3C Candidate Recommendation, "XML-Signature Syntax and Processing," 31 October 2000.

[WS-Coordination]
Web Services Coordination, IBM/Microsoft/BEA, August 2002.

[WS-Transactions]
Web Services Transactions, IBM/Microsoft/BEA, August 2002.