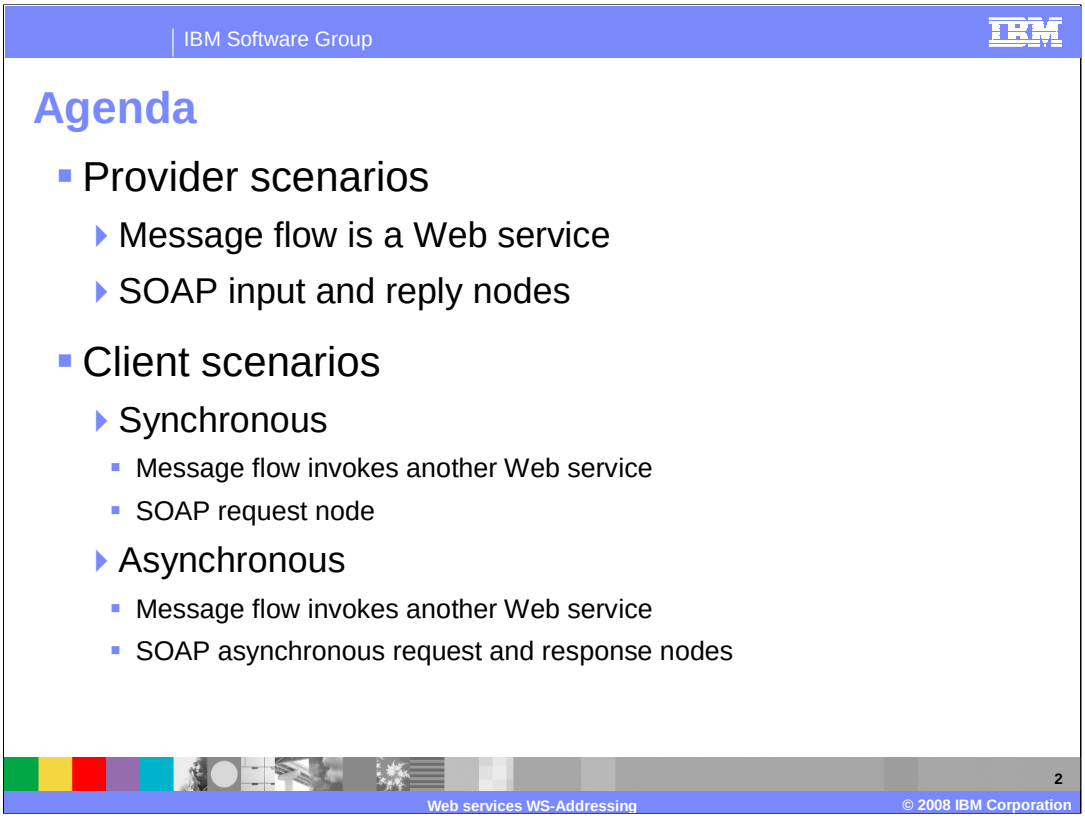
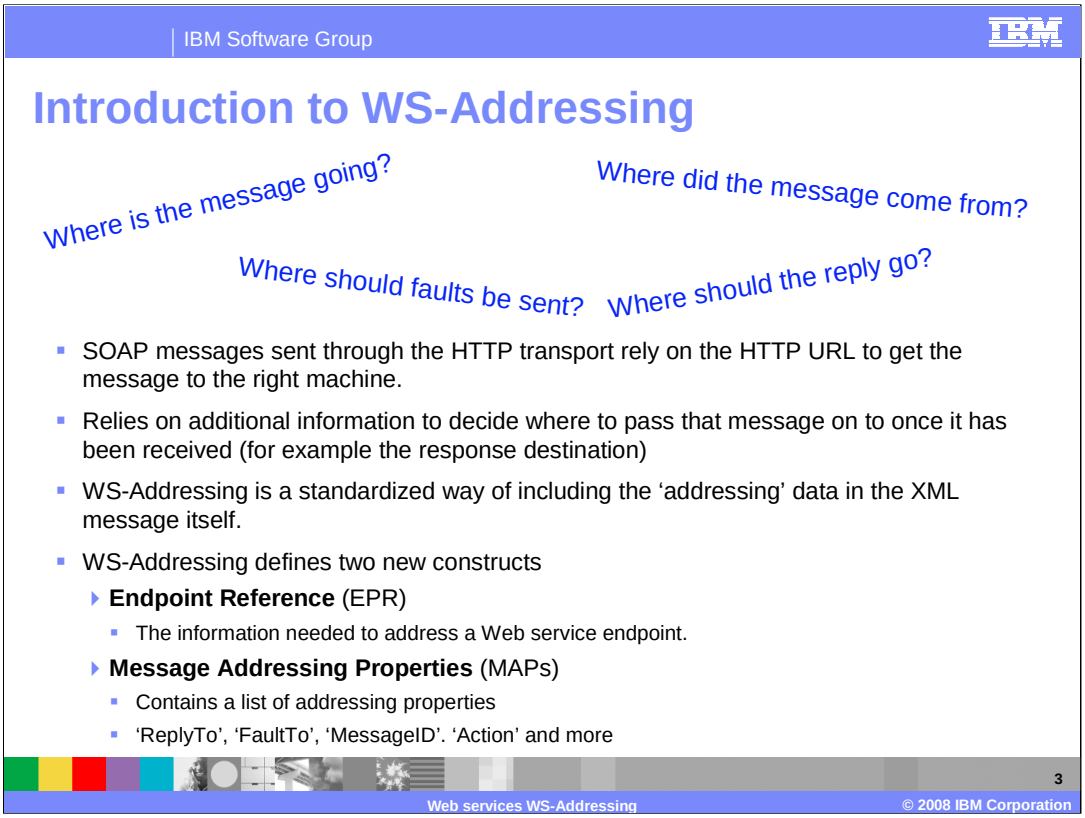
This presentation describes the Web services WS-Addressing implementation in WebSphere Message Broker Version 6.1.



This presentation will discuss the topic of Web services addressing, and will illustrate this using three scenarios.

These are a Provider scenario and two Client scenario.

In the Client scenarios, a message flow can invoke a Web service synchronously or asynchronously. These scenarios can both make use of WS-Addressing.



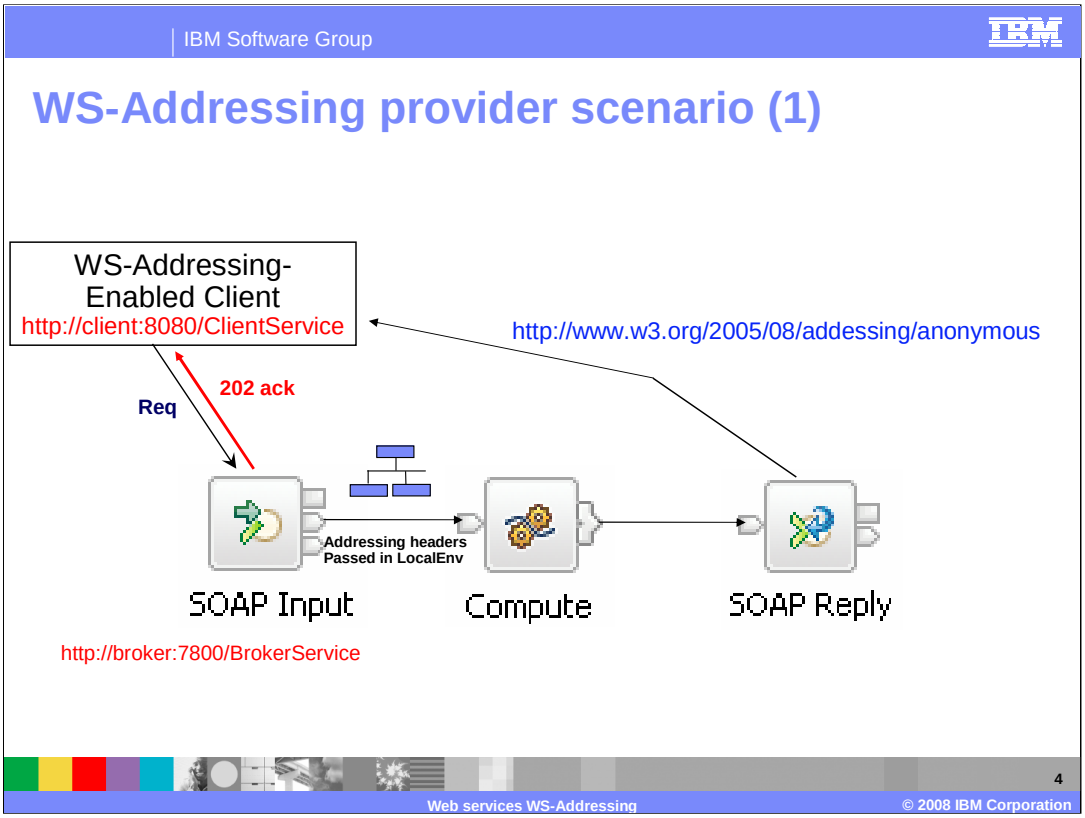
SOAP messages are generally transmitted over HTTP, and reply on the HTTP URL to ensure that the message reaches the intended destination. The HTTP address is also used to ensure that the response is sent back to the originator of the message. HTTP does not contain any mechanism to allow the originator of the message to indicate that the reply should be sent to a different destination, rather than back to the originator.

The Web services addressing standard allows you to do this, instead of using a bespoke technique. This is done by defining WS-Addressing headers, which contain information describing where the reply to the message should be sent.

To do this, WS-Addressing has defined two constructs. These are the Endpoint Reference and the Message Addressing Properties.

The Endpoint reference contains information that is needed to address a Web service endpoint.

The Message Addressing Properties contains a variety of information which allows different parts of the message to be addressed properly. This could be items such as the "ReplyTo" address and the "FaultTo" address, which would specify where faults should be directed.



This slide shows a message flow which is a Web service provider. The message flow will start with a SOAP Input node, and end with a SOAP Reply node.

In a scenario that does not use WS-Addressing, the message is sent to the SOAP Input node, and propagated to the message flow. The SOAP Reply node sends the response back to the client using the same socket.

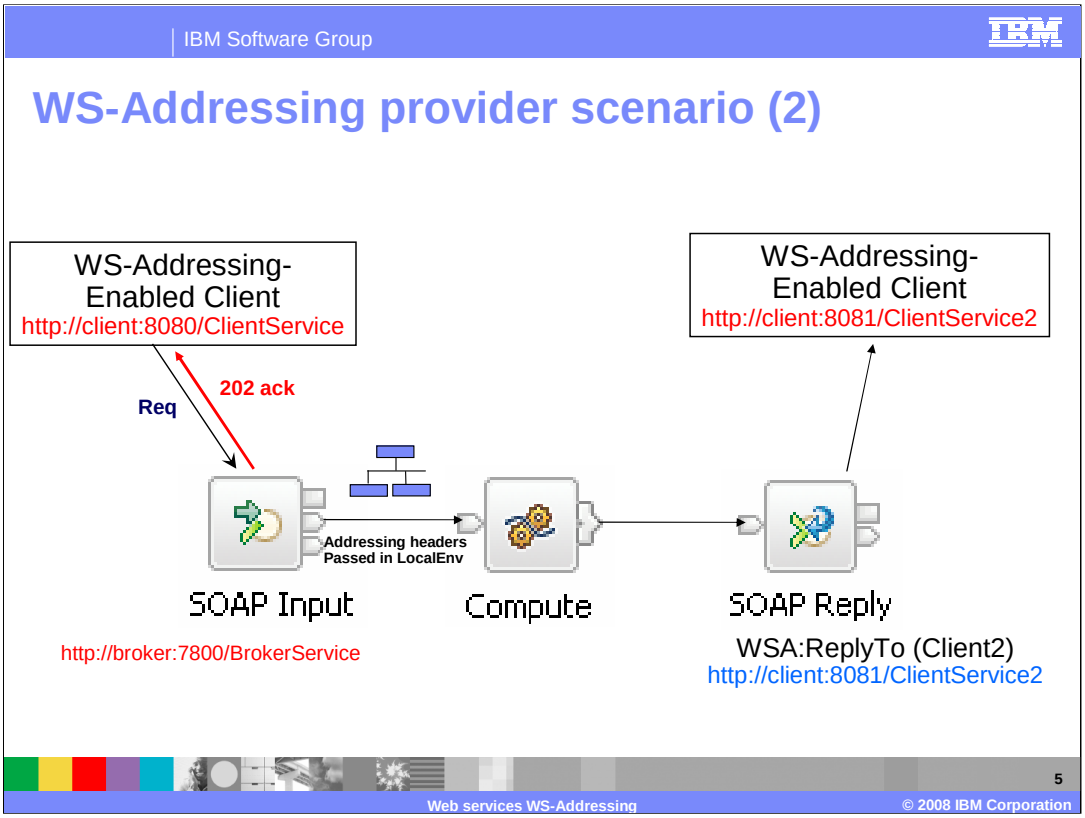
This slide shows the steps taken by the node when processing a SOAP message using WS-Addressing. The Web service client, which is enabled for WS-Addressing, sends its Web service request into the Message Broker. This is received by the appropriate SOAP Input node.

The SOAP Input node will send an acknowledgement that the request has been received by the Web service provider.

The Web service request will carry the “WSA” headers, which contain the “Reply To” header. The SOAP Input node processes this message, generates a response address header, and places this into the Local Environment. The WS-Addressing headers themselves are then removed from the incoming SOAP message. The message is then propagated to the rest of the message flow to the SOAP Reply node, along with the Local Environment headers.

The SOAP Reply node uses these headers to send the reply message to the Endpoint Reference specified in the original WS-Addressing header.

In the example shown on this slide, the “Reply To” address is specified by the client as “anonymous”, using the “W3C” standard format for this. This value has a special meaning, and the SOAP Reply node will send the reply back to the original requester of the Web service.



In this case, the Web service client has now specified the “Reply To” address as “Client 2”, with an associated URL address. The SOAP Reply node uses this address to send the reply, and in this case the original client will not receive any response from the SOAP Reply node. It will however, as in the first case, receive an acknowledgement that the request was received by the Web service provider.

WS-Addressing provider scenario

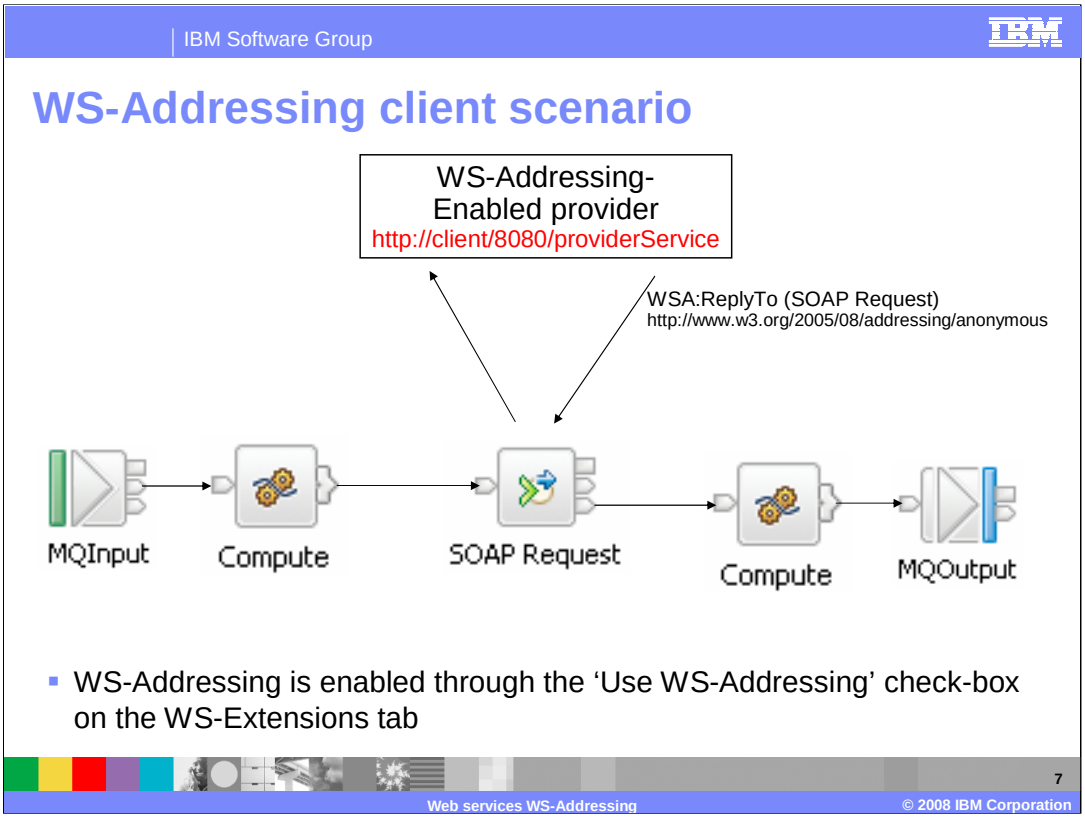
- WS-Addressing is enabled through the ‘Use WS-Addressing’ property. If not enabled, WSA headers are ignored
- SOAP Reply node sends message to the URI within the To field of the reply message
- If a Fault is detected, the FaultTo field is used to return the fault to the client
- WSA:ReplyTo can be a different EPR from the original requester

WS-Addressing is enabled using the properties of the SOAP Input node. This is done using the WS-Extensions tab of the properties. All that is required is for you to tick the WS-Addressing check-box, re-deploy the message flow, and WS-Addressing will be enabled. A Web service client that is enabled for WS-Addressing will then be able to use WS-Addressing with this message flow.

If WS-Addressing is not enabled on the message flow, and the client uses WS-Addressing, the WSA headers will not be processed. If these headers are marked “must understand”, a SOAP fault will be returned.

If a Fault is detected, the WSA header can be used to specify where the Fault should be sent.

The “Reply To” Endpoint Reference can be different to the original requester.



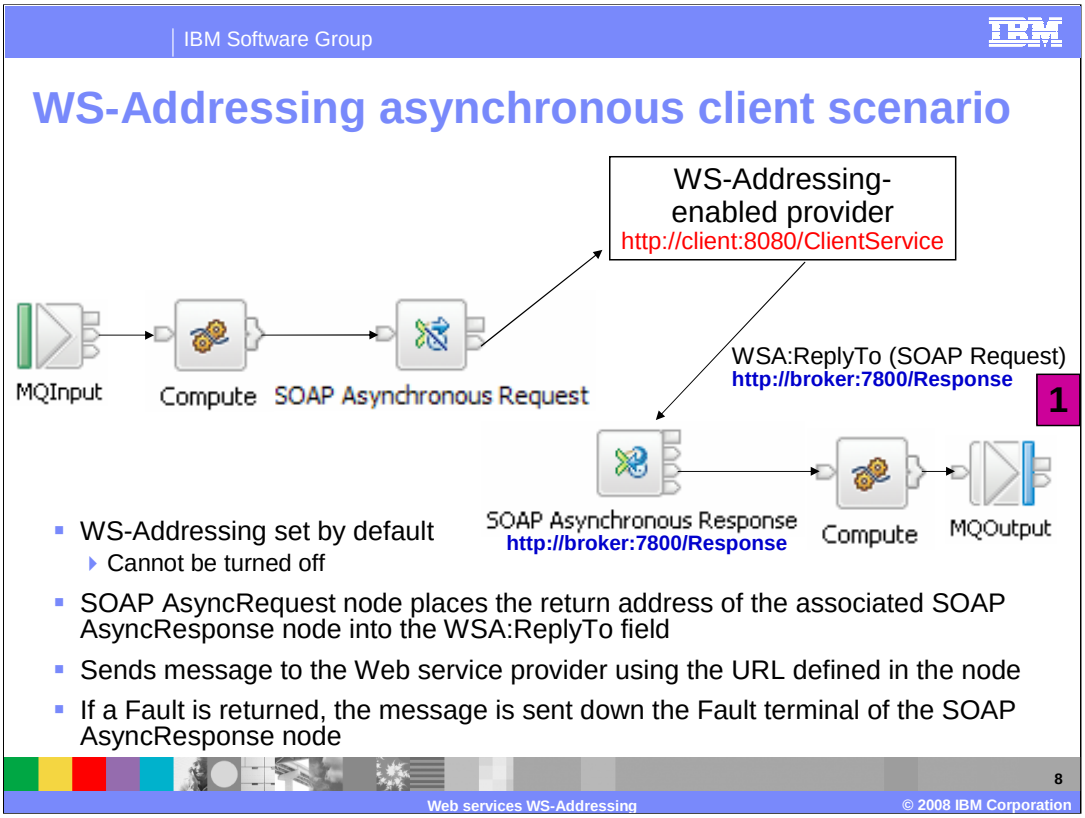
The second scenario is the Client scenario. The message flow uses the SOAP Request node to invoke a Web service. This Web service will be enabled for WS-Addressing, and will therefore expect to receive the WSA headers.

As with the SOAP Input node, WS-Addressing is enabled by using the WS-Extensions tab on the properties of the SOAP Request node.

If WS-Addressing is enabled, the SOAP request node will generate the WSA headers that will be required by the service provider. Most of the WSA header fields will be populated with default properties. For example, the "Reply To" field will be set to "anonymous", so the reply will be sent back to the original requester, which is the SOAP Request node itself.

The WSA headers can be over-ridden by using values in the Local Environment. The node will first look at the "Destination.SOAP.Request.WSA" folder in the Local Environment. If this folder is empty, the SOAP Request node will automatically generate all required message addressing properties in the outgoing message.

Once the response to the request is received, the Request node will remove all WS Addressing headers from the response message and place them in the "Destination.SOAP.Response.WSA" folder. You can query them if required, in a similar fashion to the way the SOAP Input node deals with the Input WS Addressing headers.



The third scenario is where the response to the original Web service request is handled separately from the request. In this case, the use of WS-Addressing is mandatory, and the property is automatically set on the SOAP Asynchronous Request node.

The SOAP Async Request node will generate the required WS-Addressing headers. However, the “Reply To” header is the reply address of the associated “SOAP Asynchronous response” node. In the example on this slide, this is the URL shown as “1”. Because the Web service provider is enabled for WS-Addressing, the reply will honor the “Reply To” header, and will be sent back to the SOAP Asynchronous Response node. The second part of the message flow will then complete as normal.

As before, the SOAP Fault conditions are honored, and faults will be sent to the address indicated by the “Fault To” header, if specified.

WS-Addressing versions

- There are two prominent versions of WS-Addressing that are commonly used:
  - W3C Final WS-Addressing
  - Member Submission WS-Addressing
- Both versions are supported
  - Will default to W3C Final WS-Addressing version
- Changing to 'Submission' from 'Final'
  - SOAP Reply node
    - SET OutputLocalEnvironment.Destination.SOAP.Reply.WSA.Version = 'Submission-2004/08';
  - SOAP Request node and Async Request node
    - SET OutputLocalEnvrionment.SOAP.Request.WSA.Version = 'Submission-2004/08';

WS Addressing is a core Web services technology. There are two main versions of WS Addressing, which are referred to as the Submission version and the Final version.

WebSphere Message Broker supports both versions of the standard. The default version is the Final version. If you want to use the Submission version, this can be changed by setting the appropriate field in the Local Environment, as shown on this slide.

Local environment overrides

- Large number of overrides and status information contained within the LocalEnvironment
- Allows properties to be over-ridden
  - Defined within the node dynamically
- Too many to cover in this presentation
  - Next four slides contain tables

Most of the WS-Addressing fields can be over-ridden. The next few slides show details of these. Further information is provided in the information center. The remainder of the slides in this presentation are included for reference.

SOAP request and SOAP AsyncRequest nodes

Path under LocalEnvironment.Destination	Element Type	Node using path
SOAP.Request.Transport.HTTP.WebServiceURL	String, Remote Web service	Both
SOAP.Request.Transport.HTTP.Method	String (GET, POST etc)	Both
SOAP.Request.Transport.HTTP.Timeout	Integer	Both
SOAP.Request.Transport.HTTP.ProxyConnectHeaders	String	Both
SOAP.Reply.Transport.HTTP.SSLProtocol	String, SSL, SSLv3 or TLS	Both
SOAP.Reply.Transport.HTTP.SSLCiphers	String, list of acceptable ciphers	Both
SOAP.Reply.Transport.HTTP.ProxyURL	String, Proxy URL	Both
SOAP.Request.Operation	String, Operation name	Both
SOAP.Request.UserContext	BLOB of user context information	SOAP Async Request only
SOAP.Request.WSA.*	Request EPR	Both

SOAP request and SOAP AsyncResponse nodes

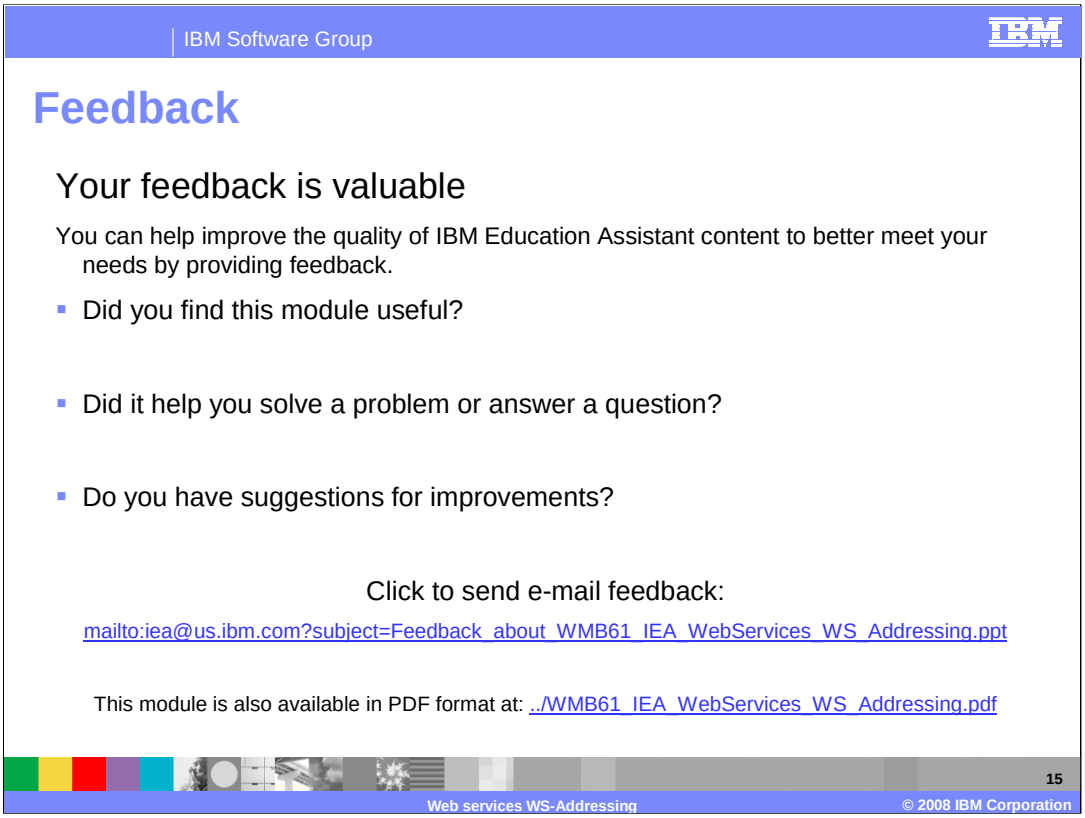
Path under LocalEnvironment.Destination	Element Type	Node using path
SOAP.Response.Transport.HTTP.AckStatusCode	Integer, HTTP Return code from the remote server (202 if success)	Populated by SOAPAsyncRequest and SOAP Reply when sending Async reply
SOAP.Response.Transport.HTTP.AckStatusLine	String, HTTP Status Line code from the remote server	Populated by SOAPAsyncRequest only and SOAP Reply when sending Async reply
SOAP.Response.Transport.HTTP.AckResponseHeaders	BLOB, HTTP Response Headers from the remote server	Populated by SOAPAsyncRequest and SOAP Reply when sending Async reply
SOAP.Response.Transport.HTTP.AckResponseBody	BLOB, HTTP Response Body from the remote server (if any)	Populated by SOAPAsyncRequest and SOAP Reply when sending Async reply
SOAP.Response.UserContext	BLOB of user context information	Populated by SOAPAsyncResponse only
SOAP.Response.WSA.*	Request EPR	Both

SOAP input node

Path under LocalEnvironment.Destination	Element Type	Node using path
SOAP.Input.WSA.*	Input EPR	Populated by SOAP Input

SOAP reply node

Path under LocalEnvironment.Destination	Element Type	Node using path
SOAP.Reply.Transport.HTTP.ReplyStatusCode	HTTP status code (500 etc)	SOAPReply
SOAP.Reply.Transport.HTTP.AsyncWebServiceURL	String, Remote Web service	SOAPReply
SOAP.Reply.Transport.HTTP.AsyncSSLProtocol	String, SSL, SSLv3 or TLS	SOAPReply
SOAP.Reply.Transport.HTTP.AsyncSSLCiphers	String, list of acceptable ciphers	SOAPReply
SOAP.Reply.Transport.HTTP.AsyncTimeout	Integer, timeout in seconds	SOAPReply
SOAP.Reply.Transport.HTTP.AsyncProxyURL	String, Proxy URL	SOAPReply
SOAP.Reply.Transport.HTTP.AsyncProxyConnectHeaders	String, list of extra headers to send to a proxy server when using SSL	SOAPReply
SOAP.Reply.Transport.HTTP.AsyncMethod	String, GET, POST etc	SOAPReply
SOAP.Reply.ReplyIdentifier	Inbound request ID (BLOB)	Populated by SOAP Input, used by SOAP Reply



You can help improve the quality of IBM Education Assistant content by providing feedback.

Trademarks, copyrights, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

WebSphere

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.