This session discusses the new MQGet node in Message Broker Version 6.

Agenda

- Overview
- Configuration
- Debugging
- Summary and References

The agenda for this presentation includes an overview of the MQGet node, as well as configuration and debugging information.

Section

Overview

This section will provide an overview of the MQGet node.

Overview

- The MQGet node provides the ability to retrieve messages from an WebSphere MQ queue in the middle of a flow
- All MQGMO options are available, to allow for maximum flexibility in usage
- MQGet node handles messages in the following message domains:
  - MRM
  - XML
  - XMLNS
  - XMLNSC
  - JMSMap
  - JMSStream
  - MIME
  - BLOB
  - IDoc

The MQGet node reads a message from a specified queue, and establishes the processing environment for the message. If appropriate, you can define the input queue as a WebSphere MQ clustered queue or shared queue.

The MQGet Node allows a message flow to retrieve WebSphere MQ messages from a queue at any point from within the flow, unlike an MQInput node which can only be used as the first node in a message flow. Additionally, multiple messages may be processed in a flow using the MQGet node.

When you use the MQGet node, you are able to specify any of the MQGMO options, without restriction.

The output message tree from an MQGet node is constructed by combining the input tree with the result tree from the MQGET call. You can set the properties of the MQGet node to control the way that messages are received. For example, you can indicate that a message is to be processed under transaction control, or you can request that, when the result tree is being created, data conversion is performed on receipt of every input message.

The MQGet node handles messages in the following message domains:
MRM
XML
XMLNS
XMLNSC
JMSMap
JMSStream
MIME
BLOB
IDoc

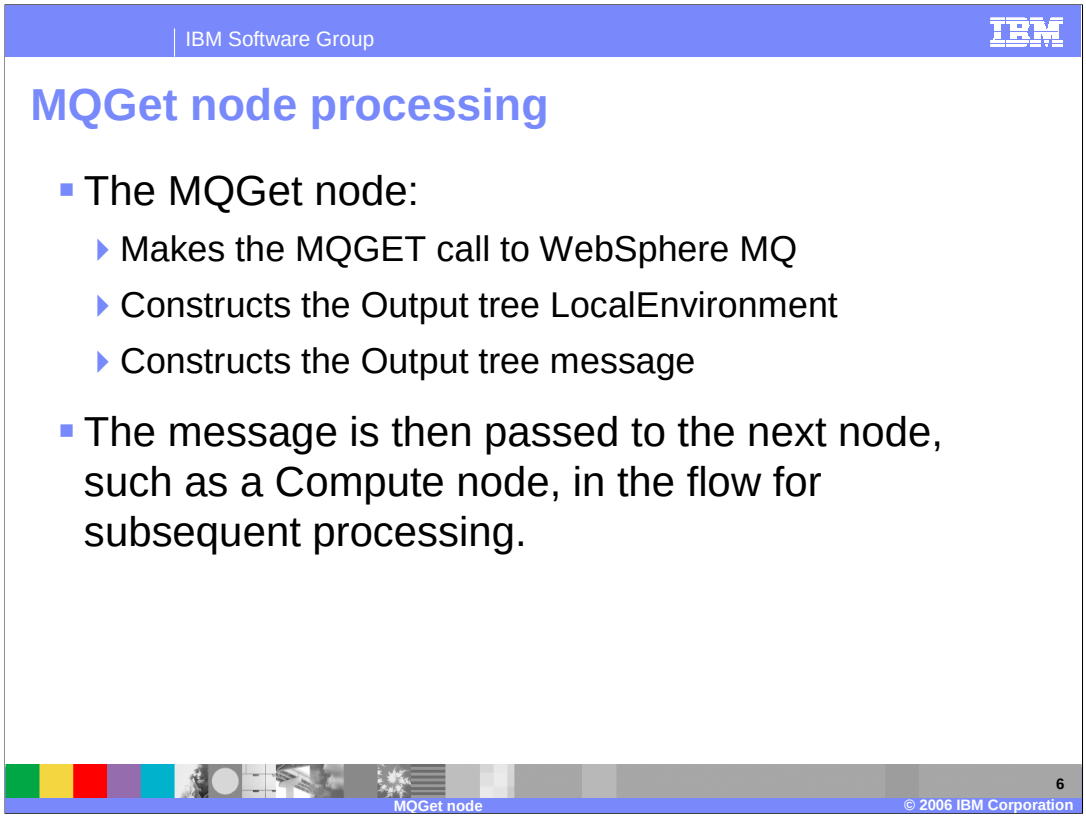
MQGet node – When is it useful?

- When you need to retrieve a message in the middle of a flow
  - Removes unnecessary flows
  - Saves extra parsing and syncpointing, failure modes
  - Holds thread until MQGET timeout
- Request-Reply
  - Stores message in intermediate state
  - Allows subsequent retrieval by nodes in other flows, execution groups
- SOAP over MQ
  - WebSphere MQ transport for SOAP request/response model

The primary use for the MQGet node is to retrieve data that has been previously stored in a queue, when the data is required part-way through a message flow. With the MQInput node, the queue can only be accessed at the start of the message flow; the MQGet node allows you to access queues at any time. You can also access several messages from the same queue, or from multiple queues.

A common use of the MQGet node in a message flow is for request-reply design. The reply part of the application will normally be held in a separate message flow; hence the data that is needed for the reply to the originating application needs to be stored locally by the request flow. This can be done by putting the message on a queue that is then retrieved from WebSphere MQ by the MQGet node.

It can also be used when using Web services over WebSphere MQ. In this case, the web service request is invoked synchronously, so the message flow will wait until a reply is received. However, the reply will be sent back by way of a WebSphere MQ queue, and this response can be retrieved using the MQGet node.



The following processing is done within the MQGet node.

Propagating the message

1. If there is an MQMD in the input tree it is used, otherwise a default MQMD is used.
2. A default MQGMO is created, then if there is a GMO in the input tree it is used to modify the default one according to the node attributes.
3. The MQGET call is made to WebSphere MQ.
4. The Return Code (CC) from the call is analyzed, and the message is propagated accordingly; if the message was received without error, the output LocalEnvironment and output Message trees are created using standard message-parsing techniques.

Constructing the Output LocalEnvironment

1. If the generateMode attribute on the MQGet node is not one of the options that includes LocalEnv, then set the output local environment to be the input, and propagate the local environment. Note: In this case, no updates that go into OutputLocalEnvironment will be propagated downstream.
2. Otherwise, if the copyLocalEnv attribute is not set to none, then copy the input local environment into the output.
3. If the output data location points to the output local environment, then changes are inserted here by copying from the result tree.
4. The local environment is propagated.

Constructing the Output message

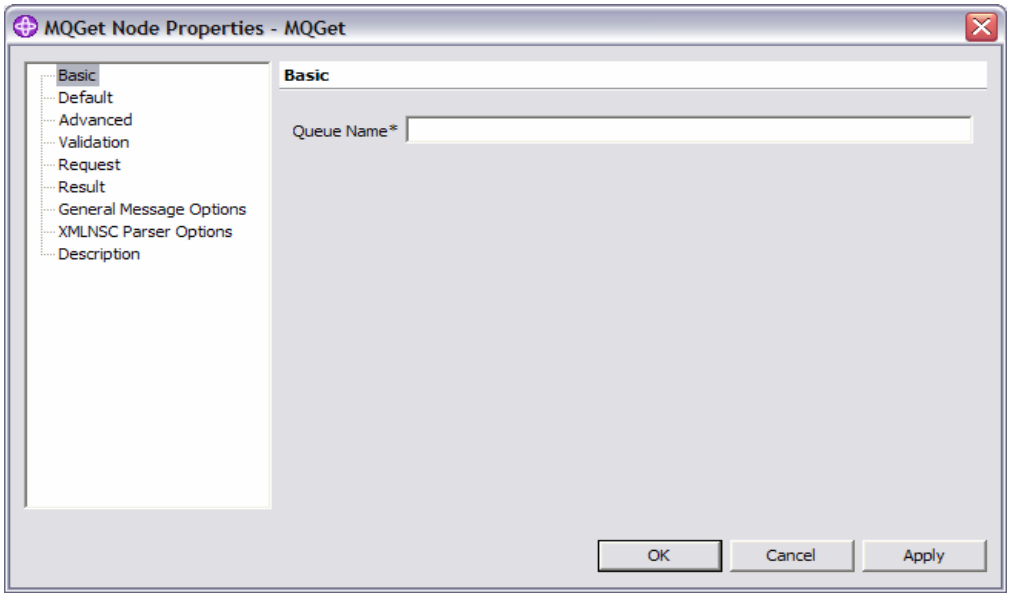
1. If generateMode does not include message, then set the output message to be the input one. Go to step 5.
2. If output Data Location is set to OutputRoot, then create the output message entirely from the result tree. Go to step 5.
3. If copyMessage is not set to none, then copy the input message into the output message tree.
4. If the output data location points to a part of the output message tree, then changes are inserted here by copying from the result tree at the point defined by result Data Location.
5. The message is propagated.

Section

Configuration

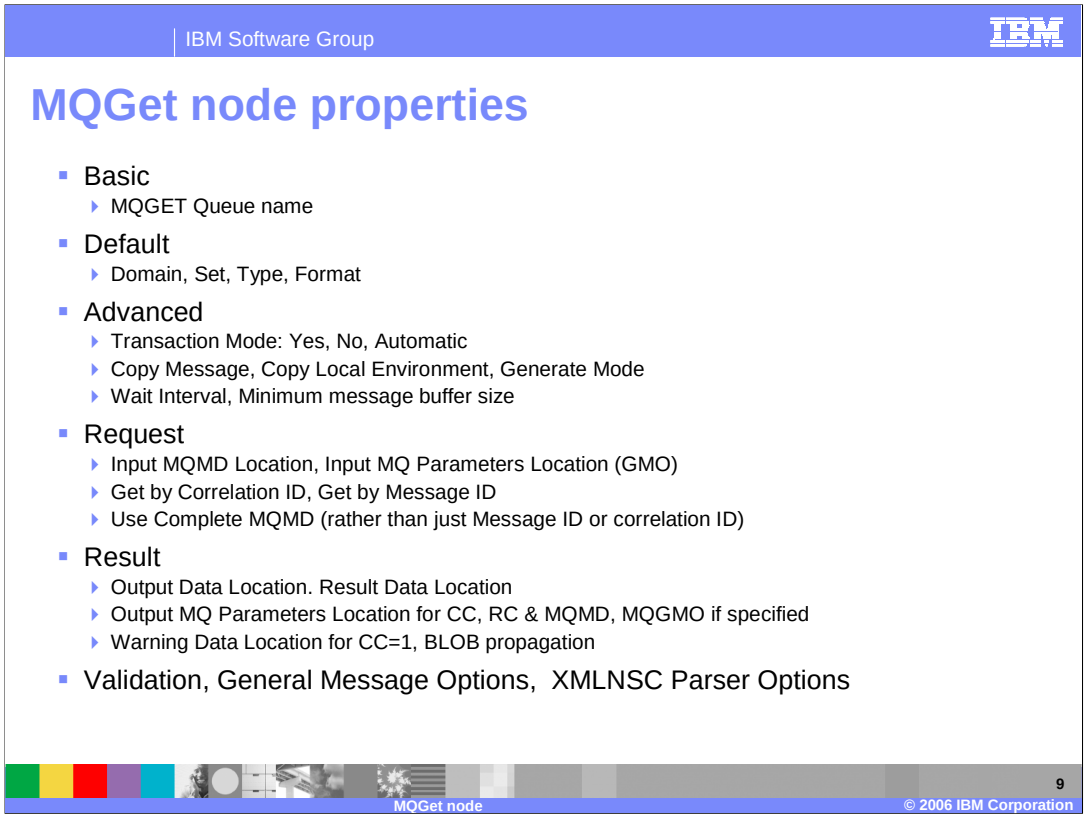
This section will discuss configuration of the MQGet node.

MQGet node properties



In common with the normal MQInput node, there are a number of configuration parameters for the MQGet node. These are accessed using the node properties.

The screen capture here shows the MQGet node Properties dialogue options. Details of the properties are discussed on the next slide.



The Basic property specifies the WebSphere MQ queue name.

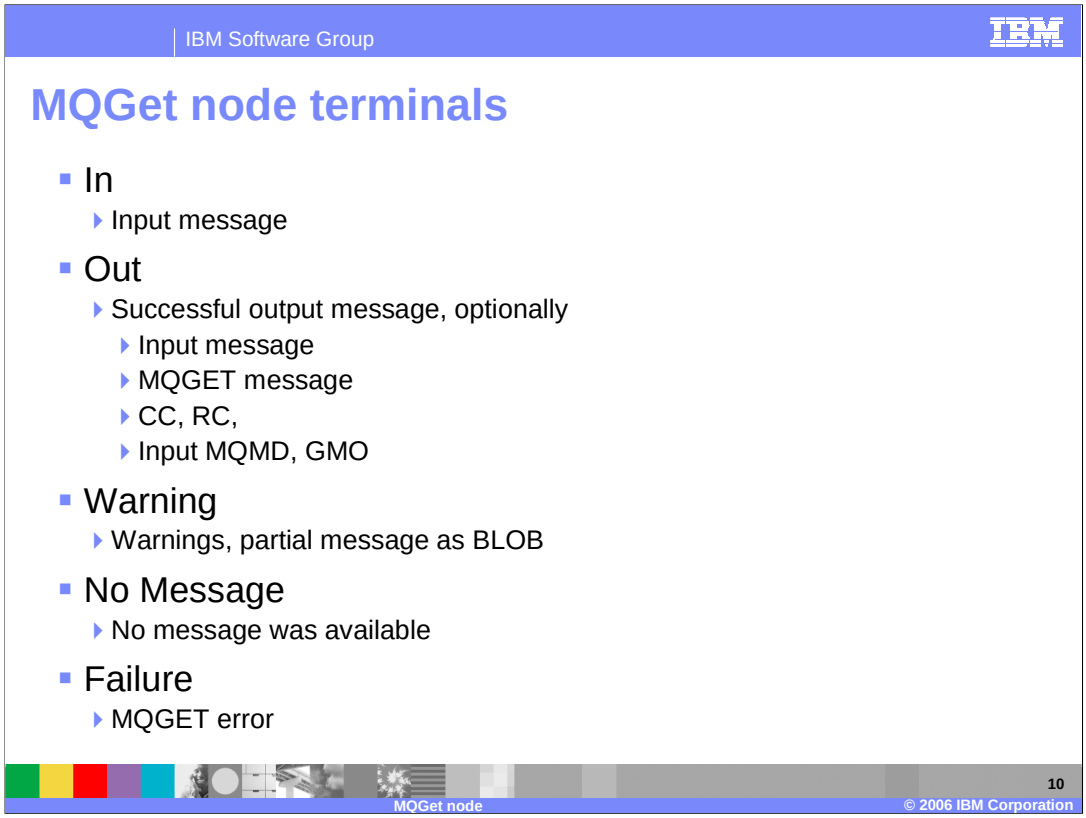
The Default properties are used to specify the parsing domain that will be used to parse the WebSphere MQ message. These parameters follow the same rules as the MQInput node. These properties can be overridden by the MQRFH2 settings.

The Advanced properties are used to specify whether the MQGET is to be included in the message flow transaction, and how the message is to be propagated through the node. It is also used to specify a Wait Interval for the MQGET. If a WebSphere MQ message is not available immediately, the flow will wait until a message arrives on the queue, or until the wait interval expires.

Select Request in the properties dialog navigator to set values for the properties that determine how the request parameters are constructed. The Request properties specify which MQMD and GMO (GetMessageOptions) are to be used. Note that it is possible to specify inconsistent options; cross-check your selections for consistency. To retrieve a particular message from a queue, use either the WebSphere MQ Correlation ID or the WebSphere MQ Message ID.

Select Result in the properties dialog navigator to set values for the properties that determine how the results of the MQGET call are handled. If the MQGET is successful, the Output Data Location properties are used to specify where the resulting data is to be placed. Result Data Location controls which part (all or which subtree) of the retrieved message is placed in the output message. Set a value in Output MQ Parameters Location to control where the CC (completion code), the RC (return code) and any other WebSphere MQ parameters (for example the MQMD used by the MQGET call) are placed in the output tree. Set a value in Warning Data Location to control where the retrieved message is placed when the MQGET call returns a Warning code.

Select Validation in the properties dialog navigator if you want the MRM parser to validate



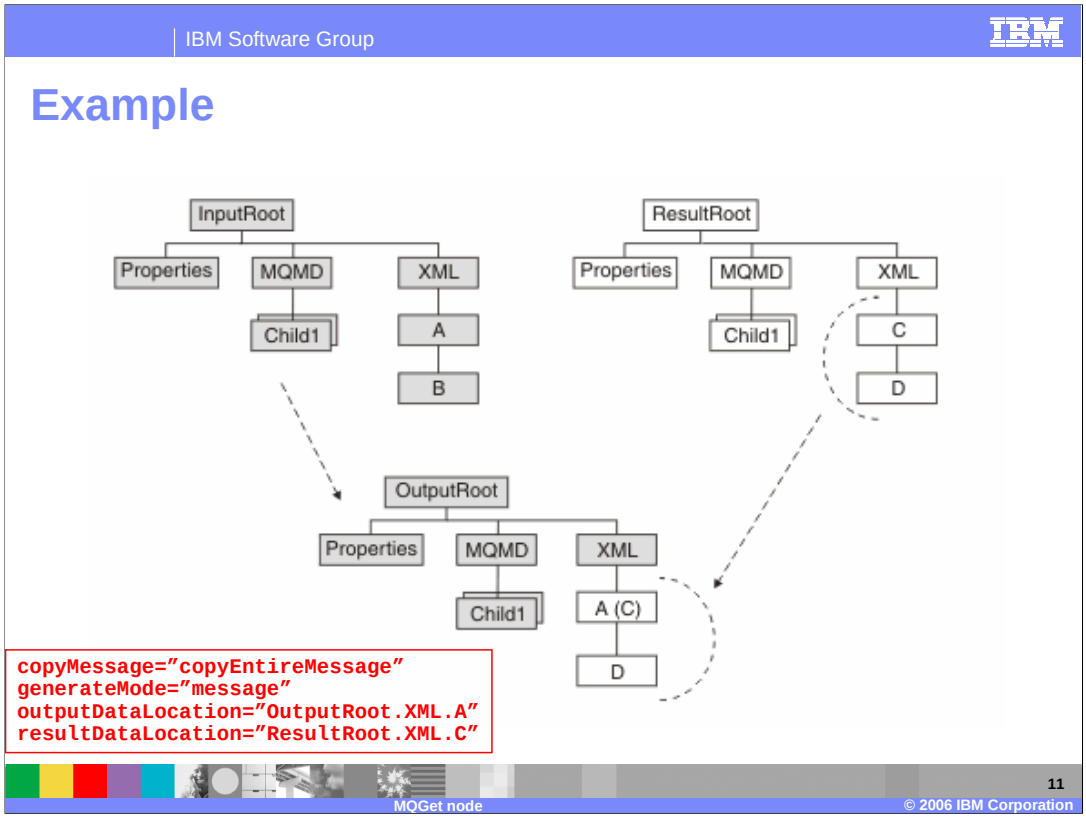
When wiring the MQGet terminals, the terminal names and functions follow the normal node conventions.

In: accepts the message that is being processed by the message flow.

Out: where the message is routed if it is successfully retrieved from the WebSphere MQ queue.

Warning: where the output tree is propagated if an error (with a CC that indicates a warning) occurs within the node while trying to get a message from the queue. The MQMD part of the message is parsed, but the rest of the message is an unparsed BLOB element. The warning is discarded if the terminal is not connected, and there is no output propagation from the node at all. If data is propagated to the Warning terminal, this normally means that the data from WebSphere MQ is not the expected data.

No message: where the input message is routed if no message was available on the queue. The output message that is propagated to the No Message terminal is constructed from the input message only, according to the values of the Generate Mode and Copy Message, or Copy Local Environment properties. In the request/reply scenario, this terminal is used when the reply message flow discovers that the message expected on the queue, in the outbound request flow, is not there when the reply message flow tries to retrieve it. This usually means that there has been a failure in the request or reply processing. Wiring this terminal provides an opportunity for the message flow to examine the cause of the failure and specify what action to take.



Here is an example, where the MQGet Node properties are configured as listed:

copyMessage = copyEntireMessage

generateMode = message

outputDataLocation = OutputRoot.XML.A

resultDataLocation = ResultRoot.XML.C

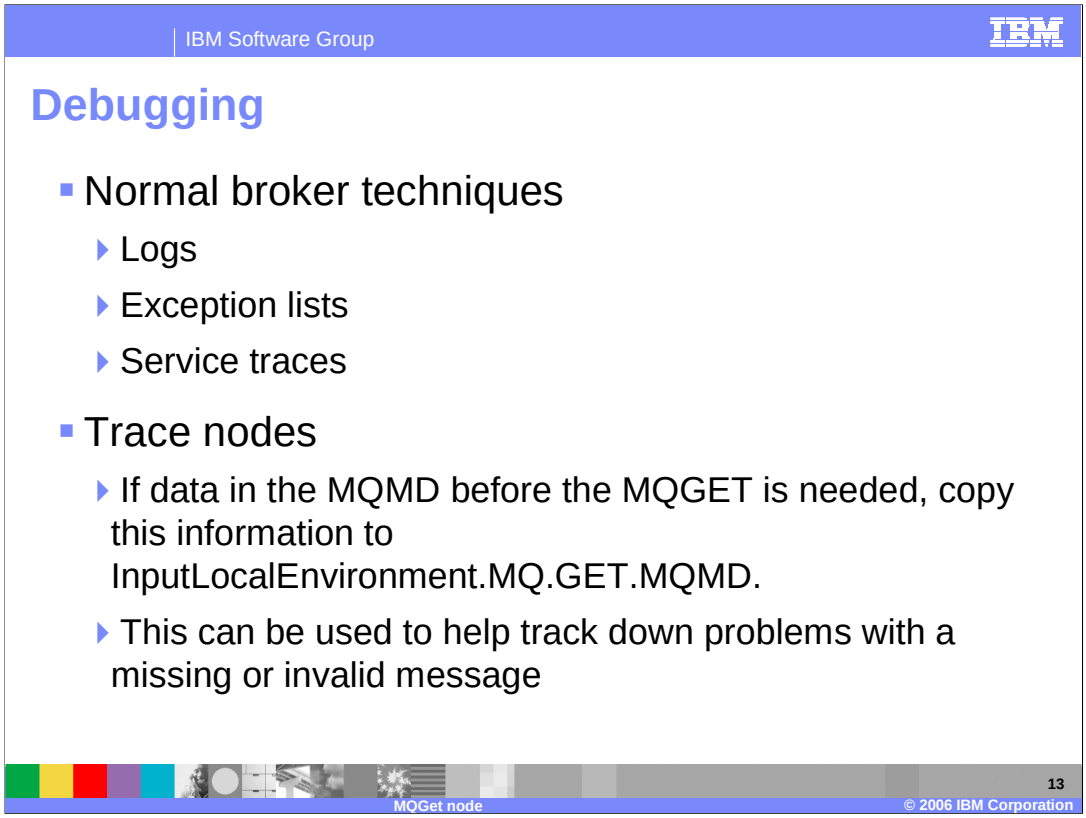
In this example, the Output tree is constructed according to the following sequence:

1. The whole of the Input tree is copied to the Output tree, including the XML branch with child A and A's child B.
2. From the Result tree, the XML branch's child C and C's child D are put into the Output tree at position OutputRoot.XML.A. Any previous content of A (values and children) is lost, and replaced with C's content, including all values and children it has, in this case child D.
3. The position in the Output tree remains named A.

Section

Debugging

This section will discuss debugging information.



Normal Message Broker debugging techniques are applicable when using MQGet. Logs and traces are integral in problem determination. However, using a Trace node with MQGet may need special handling as the trace output may not reflect the requested MQMD. If you create an MQMD and pass it to the MQGet node, the MQMD will have been updated after completion of the MQGET call. This new information will be used when writing the output tree to the trace. If you need to examine the data in the MQMD before the MQGET call was executed, you should copy this information to the InputLocalEnvironment.MQ.GET.MQMD. It can then be written to the output trace when the trace node is encountered.

If the MQGet node fails to retrieve any message at all, all data is propagated to the “No Message” terminal connection. If this happens even though it is clear that there are messages on the queue, it will be important to look at the MQMD and WebSphere MQ Get Message Options that are used by the MQGet node. Using the technique of storing these locally first, will provide the needed MQMD in the trace.

Debugging

- How to interpret the documentation
  - Look for any parsing errors in the result tree (in service trace).
  - Check unexpected WebSphere MQ return codes (normally turn up in an exception).
  - Check for incorrect data on input as a result of Get Message Options (such as MQMD.CorrelId) incorrectly spelled (in Trace node of service trace).

The MQGet node provides the normal debugging information, so you should look for any parsing errors in the results tree, or by using trace techniques.

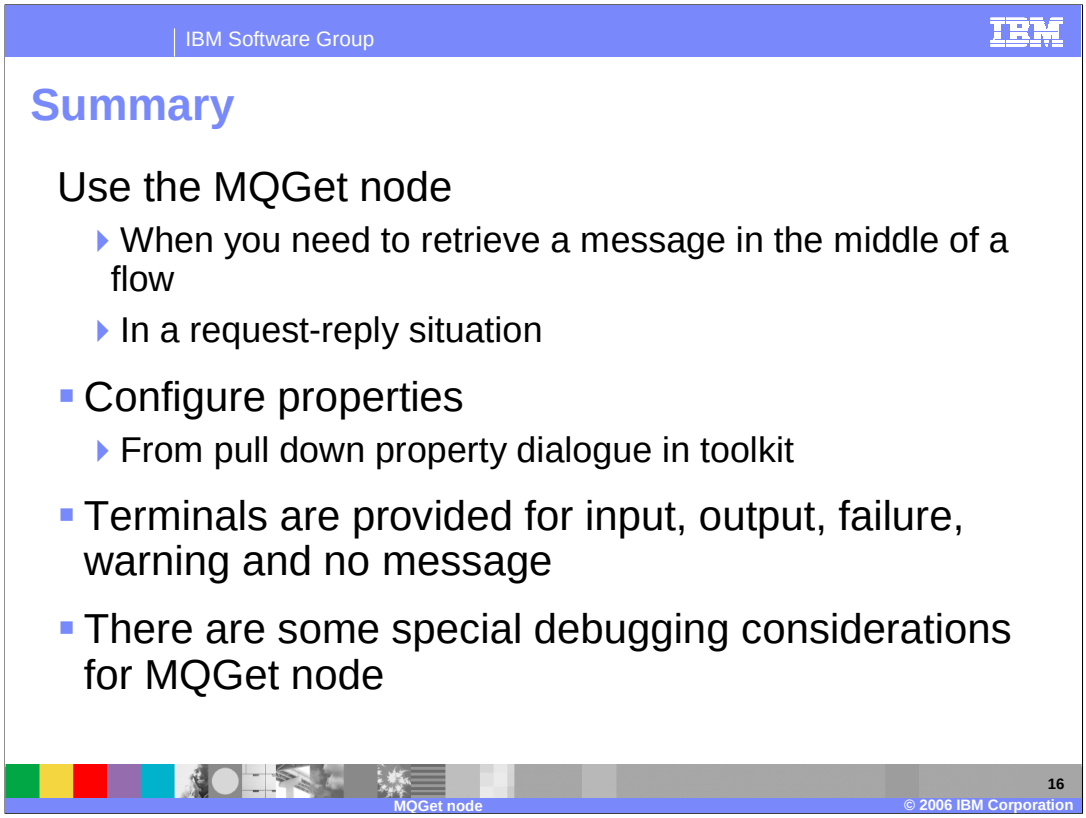
Note that the MQGet node simply passes all WebSphere MQ return codes through to the message flow. The only time that the MQGet node performs some special action is to handle the “No message” failure, which is propagated to the appropriate output terminal.

Check that the required keywords for setting “Get Message Options” are correctly spelled. This could be the cause of incorrect data on input.

Section

Summary and References

This section contains a summary and references.



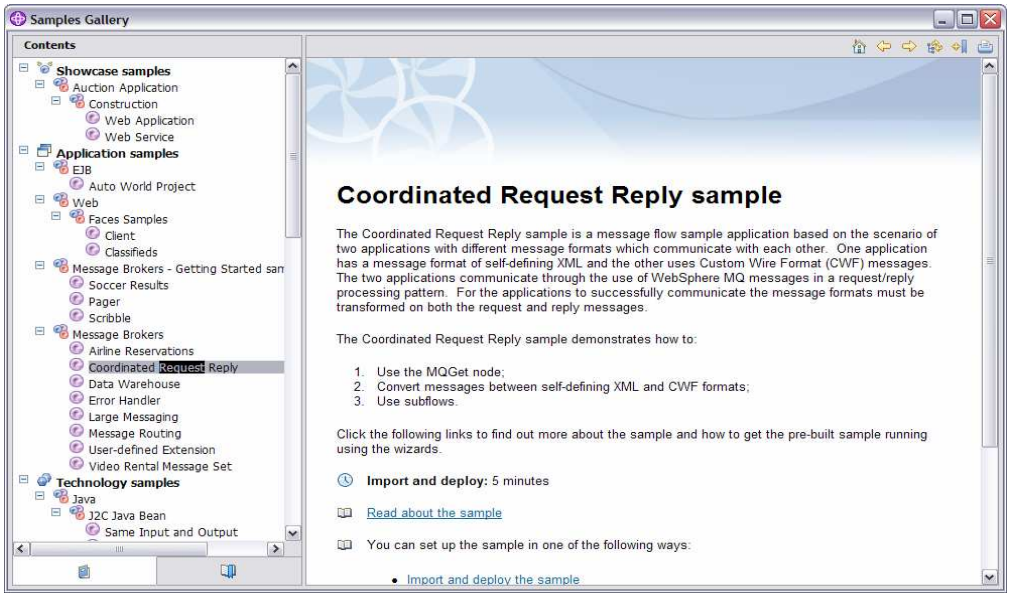
Use the MQGet node when you need to retrieve a message in the middle of a flow. A request-reply situation, whether native WebSphere MQ or SOAP over MQ, can be handled in a single message flow.

Properties determining the message content and behavior in the flow are configured in the pull-down property dialogue.

The input terminal of the MQGet node provides the WebSphere MQ message to the flow. The output terminal options are Output (for normal processing) and Failure, Warning and No Message for exception processing.

MQMD information in a trace may be misleading as this information is modified in MQGet node processing. Special handling may be needed to retain original information. The 'No Message' terminal supplies an additional option for exception handling.

Samples Gallery



There is a good example of using the MQGet node in the Samples Gallery; look for the “Coordinated Request/Response” sample provided under Help > Samples Gallery in the Message Broker Toolkit (shown in the screen print here).

References

- WebSphere Message Broker library:

http://www-306.ibm.com/software/integration/wbimessagebroker/library/

- WebSphere Message Broker Information Center:

http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r0m0/index.jsp

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.