

WebSphere Message Broker



Introduction

Version 6 Release 0

WebSphere Message Broker



Introduction

Version 6 Release 0

Note

Before using this information and the product it supports, read the information in the Notices appendix.

Third Edition (March 2006)

This edition applies to IBM® WebSphere® Message Broker Version 6.0 and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright International Business Machines Corporation 2000, 2006. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About this topic collection. v

Part 1. Product overview 1

Product overview. 3

WebSphere Message Brokers introduction. 3

WebSphere Message Broker Version 6.0 technical
overview 16

WebSphere Message Broker scenario 31

Accessibility 33

Part 2. Reference 35

Workbench 37

Perspectives in the Message Brokers Toolkit 37

Resource types in the Message Brokers Toolkit . . . 43

Editors in the Message Brokers Toolkit 46

Message Brokers Toolkit keyboard shortcuts 68

Glossary of terms and abbreviations 71

Part 3. Appendixes. 89

Appendix. Notices 91

Trademarks 93

Index 95

About this topic collection

This PDF has been created from the WebSphere Message Broker Version 6.0 (Tooling Version 6.0.0.1 update, March 2006) information center topics. Always refer to the WebSphere Message Broker online information center to access the most current information. The information center is periodically updated on the document update site and this PDF and others that you can download from that Web site might not contain the most current information.

The topic content included in the PDF does not include the "Related Links" sections provided in the online topics. Links within the topic content itself are included, but are active only if they link to another topic in the same PDF collection. Links to topics outside this topic collection are also shown, but these attempt to link to a PDF that is called after the topic identifier (for example, ac12340_.pdf) and therefore fail. Use the online information to navigate freely between topics.

Feedback: do not provide feedback on this PDF. Refer to the online information to ensure that you have access to the most current information, and use the Feedback link that appears at the end of each topic to report any errors or suggestions for improvement. Using the Feedback link provides precise information about the location of your comment.

The content of these topics is created for viewing online; you might find that the formatting and presentation of some figures, tables, examples, and so on are not optimized for the printed page. Text highlighting might also have a different appearance.

Part 1. Product overview

Product overview	3
WebSphere Message Brokers introduction.	3
Using WebSphere Message Brokers in your business	4
What's new in Version 6.0?	5
WebSphere Message Broker Version 6.0 technical overview	16
Create the run time.	17
Develop applications	18
Deploy applications to the run time	18
Publish/subscribe	19
Client environment.	19
Runtime environment	24
WebSphere Message Broker scenario	31
Mergers and acquisitions scenario	31
Accessibility	33

Product overview

This section provides introductory information to help you get started with WebSphere Message Broker:

- “WebSphere Message Brokers introduction”
 - Quick Tour
 - “What’s new in Version 6.0?” on page 5
- “WebSphere Message Broker Version 6.0 technical overview” on page 16
- “WebSphere Message Broker scenario” on page 31
- “Accessibility” on page 33
- Legal information

WebSphere Message Brokers introduction

WebSphere Message Broker and WebSphere Event Broker belong to a family of business integration products that is available from IBM.

Business integration is the coordination and cooperation of all your business processes and applications. It involves bringing together the data and process intelligence in your enterprise, and harnessing these so that your applications and your users can achieve their business goals.

Business integration means that:

- You can connect customers, suppliers, partners, and service providers, with continuing security and control, to enable newly built and re-engineered applications for more effective business processes (for example, Supply Chain Management).
- You can make mergers and acquisitions a success by integrating dissimilar IT infrastructures from more than one company so that they can work together as a single entity.
- You can react more quickly to market trends and opportunities because your IT systems are flexible and dependable, and no longer constraining.
- You can overcome the barriers of diverse computer systems, geographic boundaries, time differences, language and format differences, and different methods of working.

WebSphere MQ messaging provides a secure and far-reaching communications infrastructure that you can expand with WebSphere Message Broker or WebSphere Event Broker to apply intelligence to your business data as it travels through your network.

Transports

The main components of WebSphere Message Broker and WebSphere Event Broker (the broker, the Configuration Manager, the User Name Server, and the Message Brokers Toolkit) communicate using WebSphere MQ’s communications protocol, WebSphere MQ Enterprise Transport, or, if you implement WebSphere MQ Everywhere, WebSphere MQ Mobile Transport.

Your business applications, which can be on any of more than thirty industry platforms including those from IBM, Microsoft, and Sun Microsystems, Inc., can connect to the broker using WebSphere MQ protocols, or using other protocols,

such as WebSphere MQ Telemetry Transport, WebSphere MQ Real-time Transport, WebSphere MQ Multicast Transport, WebSphere MQ Web Services Transport, or WebSphere Broker JMS Transport.

The benefit of using WebSphere MQ protocols (WebSphere MQ Enterprise Transport or WebSphere MQ Mobile Transport) is that they provide assured, once-only delivery of messages between the components.

WebSphere MQ protocols provide rich support for applications:

- The Message Queue Interface (MQI) and Application Messaging Interface (AMI) are supported in several programming languages.
- The point-to-point (including request/reply and client/server) and publish/subscribe application communication models are supported.
- The complexities of communications programming are handled by the messaging services and are therefore removed from the application logic.
- The applications can access other systems and interfaces through adapters and gateways to products such as Lotus[®] Domino[®], Microsoft Exchange/Outlook, SAP/R3, and CICS[®] and IMS/ESA[®] products.

WebSphere Event Broker

WebSphere Event Broker provides an optimized publish and subscribe function within a limited WebSphere Message Broker framework. It provides high-performance nonpersistent publish and subscribe functionality to clients that are connected using various transports.

WebSphere Message Broker

WebSphere Message Broker incorporates WebSphere Event Broker and extends its function to provide a powerful message broker solution driven by business rules. Messages are formed, routed, and transformed according to the rules defined by an easy-to-use graphical user interface (GUI).

Diverse applications can exchange information in dissimilar forms, with brokers handling the processing required for the information to arrive in the right place in the correct format, according to the rules that you have defined. The applications do not need to know anything except their own conventions and requirements.

Applications also have much greater flexibility in selecting which messages they want to receive, because they can specify a topic filter, or a content-based filter, or both, to control the messages that are made available to them.

WebSphere Message Broker provides a framework that supports supplied, basic, functions along with user-defined enhancements, to enable rapid construction and modification of business processing rules that are applied to messages in the system.

Using WebSphere Message Brokers in your business

WebSphere Event Broker addresses the needs of business and application integration by distributing real-time information from disparate sources of information through a network of access points or a centralized broker. It allows you to:

- Publish a message to make it available to other applications. Other applications can choose to receive publications that relate to specific topics, or that have specific content, or both.
- Create structured topic names, topic-based access control functions, content-based subscriptions, and subscription points.

WebSphere Message Broker addresses the needs of business and application integration by managing the flow of information. In addition to the function of WebSphere Event Broker, it provides services, based on message brokers, to allow you to:

- Route a message to several destinations, using rules that act on the contents of one or more of the fields in the message or message header.
- Transform a message, so that applications using different formats can exchange messages in their own formats.
- Store a message, or part of a message, in a database.
- Retrieve a message, or part of a message, from a database.
- Modify the contents of a message; for example, by adding data extracted from a database.
- Exploit a public interface to develop message processing node types that can be incorporated into the broker framework to complement or replace the supplied nodes, or to incorporate node types developed by Independent Software Vendors (ISVs).
- Enable instrumentation by products such as those developed by Tivoli, using system management hooks.

The benefits of WebSphere Message Broker can be realized both within and outside your enterprise:

- Your processes and applications can be integrated by providing message and data transformations in a single place, the broker. This helps reduce the cost of application upgrades and modifications.
- You can extend your systems to reach your suppliers and customers, by meeting their interface requirements within your brokers. This can help you improve the quality of your interactions, and allow you to respond more quickly to changing or additional requirements.

Further information

For more information about WebSphere business integration, see the WebSphere Business Integration Web page. For a basic introduction to WebSphere Message Brokers, see the WebSphere Message Broker Basics IBM Redbook.

What's new in Version 6.0?

This topic introduces you to the main new function in WebSphere Message Broker Version 6.0. See also "New function added in Version 6.0 fix packs" on page 10.

- Extended platform support
- Simplified installation and migration
- Improved Message Brokers Toolkit
- Flexible transformation
- Improved performance and scalability
- Coexistence
- Enhanced support for Web services

- Driving message flows
- Enhanced message parsing
- Improved management
- Improved security
- License management
- Code page conversion
- WebSphere MQ as a transaction manager

Extended platform support

The Configuration Manager is supported on all the broker platforms including z/OS, UNIX, Linux, and Windows, see “Configuration Manager” on page 25.

The Message Brokers Toolkit can be installed on Linux (x86 platform) in addition to Windows, see “Client environment” on page 19.

Database support on Linux (x86 platform) is extended to include Oracle in addition to UDB DB2, see Supported databases.

Simplified installation and migration

The installation of WebSphere Message Broker Version 6.0 is easier than that of WebSphere Business Integration Message Broker Version 5.0 because the number of prerequisite products has been reduced. For more information on installation, see the Installation Guide.

Configuration Managers no longer require a database.

A Default Configuration wizard is provided to quickly build a full environment for development or other simple configurations on Windows and Linux (x86 platform). For more information on the Default Configuration wizard, see Using the Default Configuration wizard, and the Installation Guide.

Migration from WebSphere MQ Integrator Version 2.1 and WebSphere Business Integration Message Broker Version 5.0 are supported. Both Version 2.1 and Version 5.0 can coexist with WebSphere Message Broker Version 6.0 allowing a phased migration. In addition the command `mqsimigratecomponents` is provided to migrate individual components from one codebase to another. For more information on migration, see Migrating and upgrading.

The installation of WebSphere Message Broker for z/OS Version 6.0 has been simplified in the following ways:

- A single command can be used to create brokers, Configuration Managers, and User Name Servers.
- All commands are JCL-based, meaning no UNIX shell is required.

For more information, see the Installation Guide.

Improved Message Brokers Toolkit

The Message Brokers Toolkit is based on the latest release of Rational Application Developer, powered by the Eclipse open source platform. The Message Brokers Toolkit includes the following new features:

- Support on Linux (x86 platform) in addition to Windows.
- XML editors.
- Visual debugging is available with the following resources, which reduces the need for Eclipse perspective switching:
 - Message flows

- ESQL
- Java
- Graphical mappings

For more information, see Flow debugger overview.

For more information about the Message Brokers Toolkit, see “Client environment” on page 19.

Flexible transformation

Enhanced graphical mapping

The mapping tools support both novice and expert users, when authoring unidirectional transformations between source and target instance data elements. In addition to requiring less ESQL coding, the tools include the following enhancements:

- A Message Mapping editor with a source pane, target pane, drag and drop, spreadsheet view, expression entry field, edit window, and marker bar with break point indication for debug purposes, see “Message Mapping editor” on page 64.
- An incremental builder that validates map content and external links, and generates ESQL object code as output for deployment to the broker.
- Support for user-defined functions written in Java and ESQL.
- The ability to split source messages into a number of output messages, convert the message type automatically, map message headers, and map repeating elements.

Backward compatibility is provided by automatic migration, where existing mfmap files are rewritten in the new msgmap format. Message flows with Mapping nodes continue to function without change.

For more information about mapping, see Message mappings overview.

Routing and transformation rules in Java

You can use the JavaCompute node to write routing and transformation logic in Java. A standard J2SE 1.4.2 environment is provided with the standard Eclipse Java editor with color highlighting and code assist to create the transformations. XPath helper methods are provided, in addition to the full plug-in methods, to give easy access to message fields. No ESQL skill or experience is required.

You can deploy Java jars to the broker using the standard deployment facilities.

For more information about the JavaCompute node, see JavaCompute node.

ESQL enhancements

Transforming and routing messages using ESQL has been made easier by the following enhancements:

- In-memory cache to reduce access to databases for read only routing or validation data.
- Improved support for creating DATETIME variables.
- Access to multiple databases from the same Compute, JavaCompute, Database, or Filter node.

- Dynamic database schemas.
- New-user defined properties that can pass parameters to ESQL to modify standard behavior.
- Access to environment information, for example message flow name and broker name.
- Improved support for result sets returned by database stored procedures.
- Improved error recovery using SQL handlers.
- Multiple out terminals to combine the function of Compute and Filter nodes.

For more information, see ESQL overview.

XSLT enhancements

Performance of XSLT transformations is improved by the support of compiled style sheets. The style sheets can also be deployed using the standard deployment facilities.

For more information on XSLT transformations, see XMLTransformation node.

Improved performance and scalability

Performance of the broker runtime has been significantly improved by the following enhancements:

- The path lengths of the major broker functions have been shortened.
- The cost of parsing and streaming messages has been reduced.
- ESQL and publish/subscribe functions have been improved.
- The aggregation nodes now use WebSphere MQ queues to store state information instead of a database. This improves the throughput of all requests, with the greatest improvement being gained with non-persistent requests.
- A new in-memory cache allows more efficient message flows to be developed.
- The storage requirements of the MRM and XML parsers have been reduced so that larger messages can be accommodated.

Coexistence

WebSphere Message Broker Version 6.0 can coexist with either a Version 2.1 or a Version 5.0 product on the same computer. You can install WebSphere Message Broker Version 6.0 in a different location on the same computer, migrate your components and resources to WebSphere Message Broker Version 6.0, and uninstall the Version 2.1 or Version 5.0 product later when you are sure that you no longer need it.

With some restrictions, all Version 5.0 components can participate in a Version 6.0 broker domain, and all Version 6.0 components can participate in a Version 5.0 broker domain. A Version 2.1 broker is the only Version 2.1 component that can take part in a Version 6.0 broker domain.

For more information about coexistence, see Coexistence with previous versions and other products.

Enhanced support for Web services

Web services support has been extended in the following ways:

- SOAP 1.2 is supported.
- HTTP 1.1 is supported.

- SOAP schemas are provided, which can help simplify the modeling and transformation of SOAP messages; see Web services applications - SOAP.
- WSDL definitions can be imported using the new WSDL importer; see Importing from WSDL files to create message definitions.
- The WSDL generator has been improved; see Generating a Web Service Definition from a message set.
- Imported and generated WSDL is validated for compliance with the Web Services Interoperability (WS-I) Basic Profile 1.0.
- The HTTP transport has been extended to provide HTTPS support, which provides added privacy and security.

Driving message flows

Message flows can be driven by JMS transports. Supported JMS providers include the embedded JMS provider in WebSphere Application Server Version 6.0, see WebSphere Broker JMS Transport.

New TimeoutControl and TimeoutNotification nodes allow message flows to be driven periodically rather than by an external event, see TimeoutControl node and TimeoutNotification node.

A new MQGET node allows messages to be retrieved midway through a message flow, in addition to the beginning of the message flow. This allows groups of messages to be dealt with together, or queues to be used to save temporary state information, see MQGet node.

Enhanced message parsing

The performance of both parsing and writing messages has been significantly improved to allow greater throughput, especially of large messages.

A new WSDL importer is provided to create message models from WSDL files, simplifying the integration of Web Services.

A new MIME parser is provided to allow the parsing of multipart MIME messages such as SOAP with Attachments and RosettaNet, see MIME parser and domain.

Message models for industry standard message definitions such as SOAP envelope, MIME headers and SAP IDoc segments are supplied for inclusion in your own message sets.

There is now support for XML Schema list and union simple types, xsi:type attributes, and XML version 1.1.

Unbounded repeats supported for all kinds of message including binary and formatted text.

Enhanced support for COBOL messages that use OCCURS DEPENDING ON.

The following runtime validation improvements:

- Validation options available on more nodes.
- A new Validate node to validate a message in the middle of a message flow.
- The ability to detect all validation failures in a message before throwing an exception.
- The ability to force a complete parse of a message independently of runtime validation.

Improved management

Runtime versioning

The enhanced versioning capabilities introduced to the development environment in the last release have been extended to the runtime environment. All resources deployed can be tagged with version, author, and other useful information in addition to the standard compiled time and deployment time attributes. The new information is displayed in the administration interface making it easy to see which resources have been deployed to production systems. For more information on runtime versioning, see Message flow version and keywords.

More automation

New and extended command line utilities are provided to allow full automation of deployment of new resources to production environments.

The Configuration Manager Proxy (CMP) Java API is a new systems management application interface which is provided to allow WebSphere Message Broker to be fully managed by products and utilities besides the Message Brokers Toolkit and the command-line interface. For more information on the CMP, see Developing applications that use the Configuration Manager Proxy Java API.

Improved security

The HTTP transport has been extended to provide HTTPS support, which provides added privacy and security.

Communication between the Message Brokers Toolkit and a Configuration Manager has been extended to allow the use of SSL for added security.

License management

WebSphere Message Broker supports a new form of license management using IBM Tivoli License Manager (ITLM), Version 2.1, across all the WebSphere Message Broker platforms, with the exception of z/OS. Implementing ITLM allows sub-capacity pricing for eBusiness On Demand.

For more information on ITLM, see Installing Tivoli License Manager.

Code page conversion

The set of code page converters used by WebSphere Message Broker has been updated and is now comprised of code page converters from the International Components for Unicode (ICU) libraries for Unicode, Version 3.2. For the list of code page converters included in the set, see Supported code pages.

For information on how to add additional code page converters to the set, see Generating a new code page converter.

WebSphere MQ as a transaction manager

When using WebSphere MQ Version 6.0 as a transaction manager, data sources in coordinated message flows cannot connect to 32-bit DB2 instances. If data sources in your coordinated message flows connect to DB2, ensure that they only connect to 64-bit DB2 instances.

New function added in Version 6.0 fix packs

While the "What's new in Version 6.0?" on page 5 topic introduces you to the main new function in WebSphere Message Broker Version 6.0, this topic introduces you to additional function that has been added in fix packs.

For detailed information about the content of fix packs, go to the WebSphere Message Broker support Web page and select your product to display the recommended fix pack or to list all available fix packs. Each fix pack includes a memo, called memo.ptf, that includes details of its content.

Fix Pack 1 includes the following enhancements:

Updated Rational support for Message Brokers Toolkit

Message Brokers Toolkit is now based on Rational Application Developer (RAD) Version 6.0.1.1 and is compatible with that release and later Version 6.0.1 updates. It cannot coexist with RAD products that are based on different versions, so such products cannot be below Version 6.0.1.1, and must be below Version 6.0.2. Other Rational products must also be at the latest available level. For detailed guidance about ensuring you have a compatible Rational framework, see the Installation Guide.

Support for 64-bit execution groups

The addition of support for 64-bit execution groups allows the use of very large messages, and also enables WebSphere Message Broker Version 6.0 to run applications in fastpath (trusted) mode when using the 64-bit Queue Manager provided in WebSphere MQ Version 6.0.

Extended platform and operating system support for runtime environments

Support is now provided for the following runtime environments:

- Solaris (x86-64 platform), running Solaris 10.
- Linux (POWER platform) on iSeries and pSeries hardware, running either of the following operating systems:
 - Linux PowerPC Red Hat Enterprise Advanced Server V4
 - Linux PowerPC SUSE Linux Enterprise Server (SLES) 9
- Linux Red Hat Enterprise Advanced Server V4 to Linux (x86 platform) and Linux (zSeries platform).
- Solaris 10 on Solaris (SPARC platform) and Solaris (x86-64 platform).

See Operating system requirements for more detailed information about supported environments.

Extended database support

Support is added for the following databases:

- Microsoft SQL Server 2000, on the following platforms:
 - AIX
 - HP-UX
 - Linux (x86 platform)
 - Solaris (SPARC platform)
- Informix Dynamic Server V9.4, on the following platforms:
 - AIX
 - HP-UX
 - Linux (x86 platform)
 - Solaris (SPARC platform)
 - Windows.
- IBM DB2 V8 on Solaris (x86-64 platform) and Linux (POWER platform).

See Supported databases for details of available database support by operating environment.

Backward recovery extended to Version 2.1

It is now possible to restore components and resources that you have migrated to Version 6.0 from Version 2.1 products back to their original state. For details of how to do this, see Restoring components and resources to Version 2.1

User exit for tracing message flows

Support is provided for user exits, which enable user-provided custom software to track data passing through message flows in WebSphere Message Brokers. The user-provided functions can be invoked at specific points during the lifecycle of a message as it passes through the message flow, and can invoke utility functions to query information about the point in the flow, and the contents of the message assembly. For more information about user exits see Developing user exits

New in Version 5.0

This topic introduces you to the main new function in WebSphere Business Integration Message Broker Version 5.0. See also “New function added in Version 5.0 fix packs” on page 15.

- New graphical development environment based on Eclipse
- Enhanced initial user experience
- Web Services support
- Enhanced message modeling
- XSLT transformation enhancements
- Extended database user ID and password support
- Message flow accounting and statistics
- Publish/subscribe enhancements
- SSL authentication, QoS, and HTTP tunneling
- New object level security model
- Increased platform support

New graphical development environment based on Eclipse

The Message Brokers Toolkit is an integrated development environment and graphical user interface based on the Eclipse platform. The Message Brokers Toolkit consists of the Eclipse platform and a set of Java plug-ins that enable the creation, maintenance, and deployment of WebSphere Message Broker message flows (including publish/subscribe applications) and message models. See the “Client environment” on page 19 topic, and the Eclipse web site.

The Message Brokers Toolkit includes, among other things, the following new features:

- You can control multiple broker domains using a single Message Brokers Toolkit. See “Client environment” on page 19.
- You can use Message Brokers Toolkit with external source repositories, for example with repositories that permit version control of message flow and message set source data. See “Development repository” on page 24.
- Improved editors, such as Message Flow Mapping editor, Message Set editor, and ESQL editor, make it easier for you to build your solution. See “Editors in the Message Brokers Toolkit” on page 46.
- The Mapping node allows you to create message transformations using a drag-and-drop method, without the need to write ESQL code. See Mapping node.

- You can now reuse ESQL subroutines and functions. See Broker schemas.
- The flow debugger allows you to add breakpoints to the connections of a message flow, and step through ESQL code statement by statement. See Flow debugger.
- You can configure some message flow properties at deployment time rather than at development time. See Broker archive - configurable properties.

Enhanced initial user experience

The **pre-install LaunchPad** is launched from the Windows CD prior to installation. It guides you through the process of checking for, and installing, any necessary prerequisite software before launching the product installation. The installation is now implemented using InstallShield for Multi-Platforms.

The **Welcome** page for the Message Brokers Toolkit, which appears after you launch the Message Brokers Toolkit, provides links to a number of basic tasks, enabling you to configure and exercise a working system as quickly as possible. Supported tasks include:

- Take the Quick Tour.
- Create the Default Configuration.
- Verify your installation with the Pager samples.
- Explore the Samples Gallery.

Web Services support

Several enhancements provide support for Web Services:

- HTTPInput, HTTPReply, and HTTPRequest nodes. For more information, see Built-in nodes.
- Web Service Description Language (WSDL) generation from a message set. For more information, see Generate WSDL.
- Full support for the construction and parsing of SOAP messages, and the addition of XML namespace support through the extension of the MRM XML Wire Format and the addition of the XMLNS domain. For more information, see WSDL generation, Namespaces in the message model, and The XML domains.
- See also WebSphere Business Integration Message Broker Version 5.0 and Web Services SupportPac (IA81).

Enhanced message modeling

The following enhancements have been made to the message model:

- New logical message model based on the industry standard XML Schema 1.0. See The message model.
- Introduction of message definition files to make it easier to manage large message sets. See Message definition files.
- Full support for XML namespaces in the MRM domain and in the new XMLNS domain. See Namespaces in the message model and The XML domains.
- Introduction of user-defined simple types. See Message model objects: simple types.
- New XML Schema, XML DTD, C, and COBOL importers. See Importing data structures.
- Introduction of an XML Schema generator. See Generate XML Schema.

XSLT transformation enhancements

The XML Transformation node allows you to integrate existing XSLT transformations into the broker environment. The style sheet used to define the transformation can be specified as an attribute of the node instance, or can be determined within the context of the flow and specified using a defined area of the Local Environment. This capability is delivered through repackaging and enhancement of the XALAN-based XSLT style sheet transformation engine derived from the WebSphere Transcoding Publisher product.

For more information, see XMLTransformation node.

Extended database user ID and password support

External database access from Compute, Database and Warehouse nodes has been extended so that you can associate a specific user ID and password with a given ODBC DSN in the broker run time. This user ID and password combination is then used for all connections made by the broker to the particular DSN, overriding the default use of the broker's user ID and password.

This addresses a key requirement to be able to specify different user ID and password combinations for different databases and to be able to set this on a broker-by-broker basis. The mqsisetdbparms command is provided for setting this information in the broker.

For more information, see Accessing databases from message flows.

Message flow accounting and statistics

You can now collect statistics on the behavior of message flows. For example, you can collect data about how many messages are processed and how large those messages are, or about CPU usage and elapsed processing times. For more information, see Message flow accounting and statistics data, and the developerWorks article on message flow performance.

Publish/subscribe enhancements

The following publish/subscribe changes have been made:

- The set of protocols and distribution patterns supported by the broker has been extended to provide reliable, real-time IP multicast distribution of subscriptions over a Local Area Network. This is provided as an extension to the existing JMS IP support in WebSphere Event Broker. For more information, see WebSphere MQ Multicast Transport and Multicast publish/subscribe, and WebSphere MQ Real-time Transport.
- Publish/subscribe capabilities have been consolidated within WebSphere Message Brokers. This rationalizes the product offerings and provides a clear upgrade path from WebSphere Event Broker to WebSphere Message Broker.

SSL authentication, QoS, and HTTP tunneling

The following Internet-related items have been added:

- SSL authentication provides additional security. See SSL authentication.
- WebSphere MQ Telemetry Transport offers a range of message delivery options. See WebSphere MQ Telemetry Transport Quality of Service levels and flows.
- HTTP tunneling allows applet access through firewalls. See Tunneling.

New object level security model

You can now control access to runtime resources by object as opposed to by group. See Security for runtime resources.

Increased platform support

Supported operating environments now include Linux (x86 platform), Linux (zSeries platform), and Windows XP, giving you a wider choice of platforms on which to deploy your solution. For more information, see Operating system requirements.

New function added in Version 5.0 fix packs: While the “New in Version 5.0” on page 12 topic introduces you to the main new function in WebSphere Business Integration Message Broker Version 5.0, this topic introduces you to some of the other new function added in fix packs.

For detailed information about the content of fix packs, go to the WebSphere Message Broker support Web page and select your product to display the recommended fix pack or to list all available fix packs. Each fix pack includes a memo, called memo.ptf, that includes details of its content.

Some features added in Fix Pack 5:

- On Linux (x86 platform), you now use the DataDirect Driver Manager to connect to DB2 databases. For more information, see Configuring a 32-bit ODBC data source on UNIX systems.
- The MRM domain TDS physical format can now recognize and interpret HL7 service strings, and a new TDS messaging standard called ‘HL7’ has been added. For more information, see HL7 messaging standard.
- The broker now supports the ‘Z’ character to mean a zero time difference from Coordinated Universal Time (UTC) when it appears as a time zone in a dateTime field in a message. It is possible to accept dateTimes containing ‘Z’ on input, and to generate them on output. For more information, see DateTime as string data.
- The new mqsimigratequeues command makes it easier to migrate a Version 2.1 broker to Version 5.0 by creating any new queues that are needed by the updated broker.
- The mqsimigratemsgsets command includes a new option -pl that you can use to specify that elements that have prefixed identifiers and are referenced more than once are to be created as local rather than global. For more information, see Prefixed identifiers.

Some features added in Fix Pack 4:

- If you have existing Java code, you can now invoke it directly from ESQL. For more information, see Java routines.
- The MRM Tagged/Delimited String format can now parse and write raw TLOG messages. For more information, see TLOG messaging standard.
- The MRM Encoding Null mechanism has been extended for the COBOL initialization of 01 structures. For more information, see Importing from COBOL: supported features.

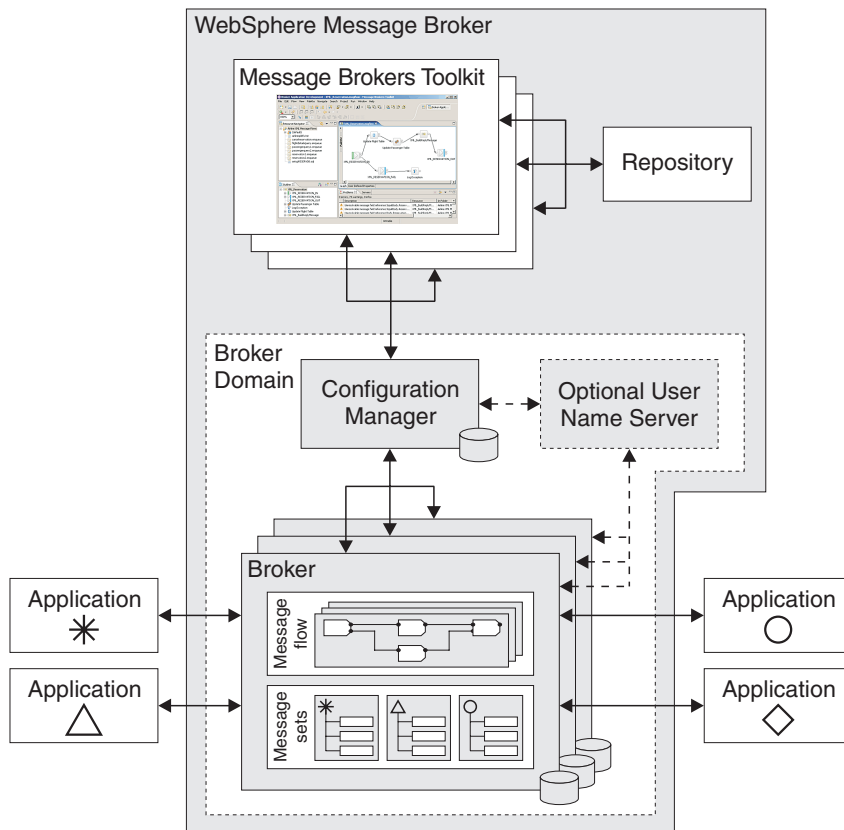
Some features added in Fix Pack 3:

- When an XML Schema is imported, the following constructs are now accepted: Key, Keyref, and Unique. However, XML Schemas with such constructs are not supported and are ignored by the broker. For more information, see Importing from XML Schema: unsupported features.

- The COBOL importer now supports the ASCII Sign EBCDIC Custom External Decimal representation within the CWF component. For more information, see Importing from COBOL: supported features.

WebSphere Message Broker Version 6.0 technical overview

WebSphere Message Broker enables information packaged as messages to flow between different business applications, ranging from large legacy systems through to unmanned devices such as sensors on pipelines.



There are two ways in which WebSphere Message Broker can act on messages.

Message routing

Messages can be routed from sender to recipient based on the content of the message.

The message flows that you design control message routing. A message flow describes the operations to be performed on the incoming message, and the sequence in which they are carried out.

Each message flow consists of:

- A series of steps used to process a message. The steps are defined in message flow nodes.
- Connections between the nodes, defining routes through the processing. Connections are made using message flow node connections.

IBM supplies built-in nodes and samples for many common functions. If you require additional functions, you can write your own user-defined nodes.

You create message flows in the Message Brokers Toolkit; an integrated development environment and broker domain administration console.

Message transformation

Messages can be transformed before being delivered:

- They can be transformed from one format to another, perhaps to accommodate the different requirements of the sender and the recipient.
- They can be transformed by modifying, combining, adding or removing data fields, perhaps involving the use of information stored in a database. Information can be mapped between messages and databases. More complex manipulation of message data can be achieved by using Extended SQL (ESQL).

Transformations can be made by various nodes in a message flow. But before a message flow node can operate on an incoming message, it must understand the structure of that message.

- Some messages contain a definition of their own structure and format. These are known as self-defining messages. Self-defining messages can be handled without the need for additional information about structure and format.
- Other messages do not contain information about their structure and format. To process them, a message definition of their structure must be created and made available.

The message definitions that you design are created within a message set, which contains one or more message definitions. Message sets also categorize message definitions. The category facility, which you can extend using XSLT scripts, is used for generating Web Services Description Language (WSDL) and documentation.

Like message flows, you create message definitions in the workbench. They can contain two types of information:

- The logical structure - the abstract arrangement and characteristics of the data, represented as a tree structure
- One or more physical formats - the way the data is represented and delimited in the physical bit stream

Create the run time

The work of routing and transforming messages takes place in a broker. Brokers contain a number of execution groups; processes in which message flows are run.

Brokers are grouped into broker domains. Each domain is coordinated by a Configuration Manager. There can be many brokers, and each can be running on a different system. This provides protection against failure, and can separate work across different divisions in a business.

The system administrator creates the Configuration Manager with a command line instruction. The Configuration Manager uses an internal repository to store information relating to its broker domain.

The system administrator similarly creates one or more brokers, linking each to a particular Configuration Manager, thus making them part of the domain controlled

by that Configuration Manager. Each broker uses a database to store the information it needs to process messages at run time.

The Configuration Manager also displays the users and groups in the Access control lists that you use to set user permissions, see Publish/subscribe below.

Develop applications

After the system administrator has created and connected the components of the broker domain, an application developer creates and modifies message flows and message definitions using the workbench.

Different perspectives in the workbench are used to develop message flows and message sets as well as to administer one or more broker domains.

A repository can be used to provide access control and version control. A repository also allows multiple developers to work on the same resources in parallel.

You can use WebSphere MQ for communication between application and brokers. Other communication protocols you can use are:

- WebSphere MQ Enterprise Transport
- WebSphere MQ Mobile Transport
- WebSphere MQ Multicast Transport
- WebSphere MQ Real-time Transport
- WebSphere MQ Telemetry Transport
- WebSphere MQ Web Services Transport
- WebSphere Broker JMS Transport

Deploy applications to the run time

When message flows and message sets have been created using the workbench, executable data can be deployed (transferred) to one or more brokers.

You can deploy data in one of two ways:

- From the workbench
- Using a shell command

When you deploy message flows and message sets, they are compiled and enveloped in a broker archive (bar) file, and sent to the Configuration Manager. The bar file has configurable system properties. You can override properties such as queue and database names, without the need to change source files or redevelop the message flow. This makes it easier to move definitions between systems.

The Configuration Manager opens the envelope, removes the contents, makes a record of the information that it has received, and routes the information to the appropriate brokers. (The envelope is discarded when the information it contains has been retrieved.) Each broker stores the information in its own local database. This means that, when a broker has sufficient information, it can continue processing messages even if it is no longer connected to its Configuration Manager.

The Configuration Manager coordinates all activity (for example, changes to a message set) between the workbench and brokers within its domain. WebSphere MQ messaging is used between the workbench, the Configuration Manager, and the brokers.

Publish/subscribe

The simplest way of directing messages is to use point-to-point messaging, by sending messages directly from one application to another. Publish/subscribe provides an alternative style of messaging.

A publishing application sends a message about a named topic to a broker. The broker passes the published message to those applications that have registered an interest in that topic. The publisher and the subscriber are unaware of the other's existence.

The broker handles the distribution of messages between publishing applications and subscribing applications. Applications can publish on, or subscribe to, many topics as well as apply more sophisticated filtering mechanisms.

An optional User Name Server in the broker domain controls who is authorized to publish or subscribe to topics. You set up and administer topic-based security from the workbench.

You set user permissions at individual or group level using Access control lists.

Client environment

The Message Brokers Toolkit is an integrated development environment and graphical user interface based on the Eclipse platform.

Application developers work in separate instances of the Message Brokers Toolkit to develop message sets and message flows. The Message Brokers Toolkit also communicates with one or more Configuration Managers, and is used to manage broker domains.

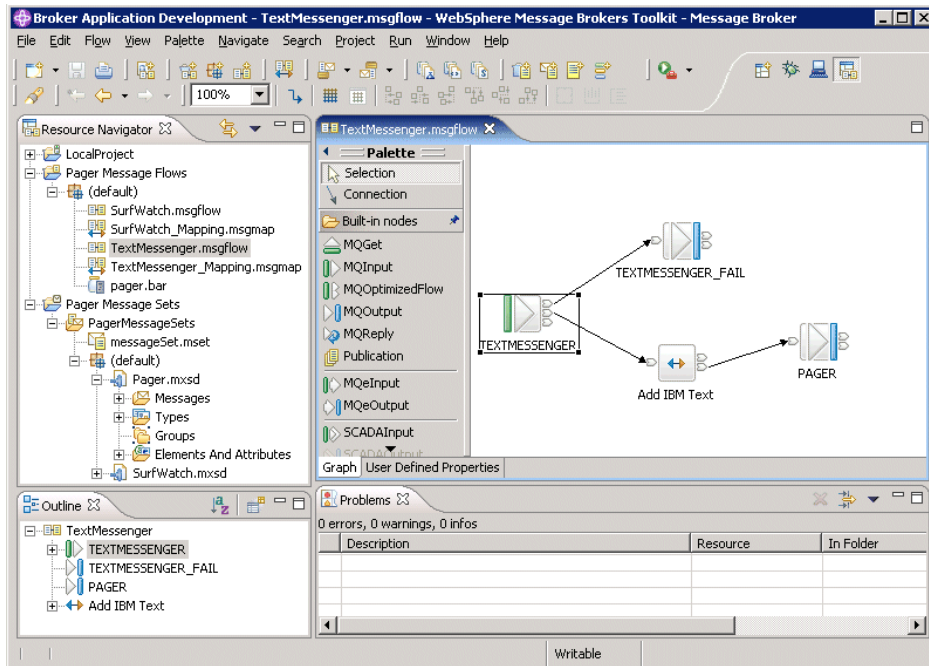
The Message Brokers Toolkit can be installed on Windows and Linux (x86 platform).

The workbench

When you start the Message Brokers Toolkit, a single window is displayed. This is the workbench, which contains one or more perspectives.

A perspective is a collection of views and editors that help you complete a specific task, or work with specific types of resource. The first time that you start the Message Brokers Toolkit, the Broker Application Development perspective is displayed. While you work in the workbench, you will switch perspectives frequently.

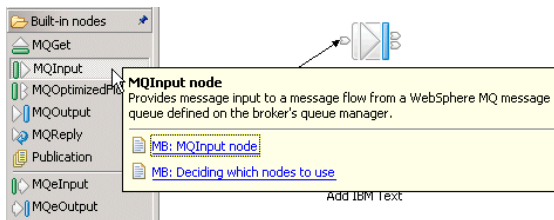
The following figure shows the Broker Application Development perspective with a message flow open in the Message Flow editor.



Accessing context-sensitive help (infopops)

Infopops are provided throughout the Message Brokers Toolkit. An infopop is a small pop-up window that displays context-sensitive help and provides links to more information in the information center. You can launch an infopop for most aspects of the user interface (for example, on the Resource Navigator, the Message Flow editor, or a properties page) by bringing focus to the object and pressing F1 (on Windows) or SHIFT+F1 (on Linux).

The following figure shows the infopop that is displayed when you press F1 on the MQInput node in the node palette of the Message Flow editor.

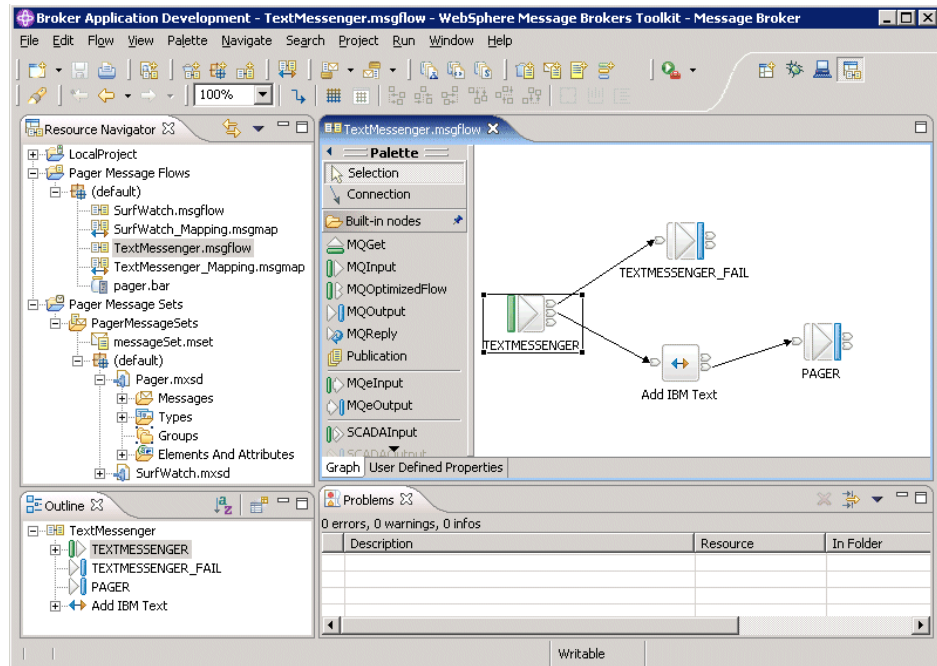


Perspectives

A perspective is a group of views and editors that shows various aspects of the resources in the workbench, and helps you to complete specific tasks.

You can switch perspectives, depending on the task at hand, and customize the layout of views and editors. Switch between perspectives by clicking **Window** → **Open Perspective** → **Other** and clicking the name of the perspective to which you want to switch.

The following figure shows the Broker Application Development perspective in the Message Brokers Toolkit.



The Message Brokers Toolkit contains the following perspectives:

Broker Application Development perspective

This is the default perspective that is displayed the first time that you start the Message Brokers Toolkit. Application developers work in this perspective to develop and modify message sets and message flows. The figure below shows the Broker Application Development perspective with a message flow open in the Message Flow editor.

Broker Administration perspective

This is a broker administration console that communicates with one or more Configuration Managers. Administrators work in this perspective to manage the resources (also referred to as domain objects) in the broker domain that are defined on one or more Configuration Managers.

Debug perspective

This is where application developers test and debug message flows.

Plug-in Development perspective

This is where application developers develop plug-ins for user-defined extensions.

Data perspective

This is where application developers import relational database schemas for ESQL content assist and validation.

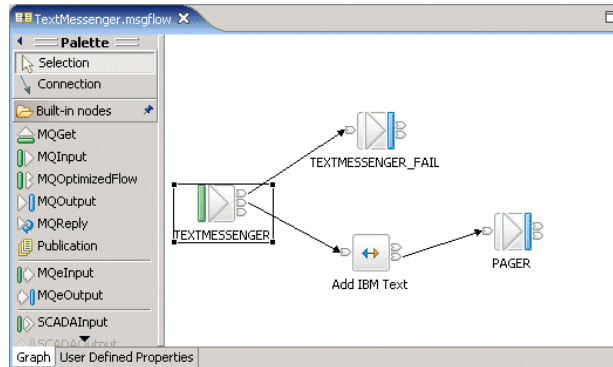
Editors

An editor is a component of the workbench. It is typically used to edit or browse a resource.

When you open a file for editing, for example by double-clicking it in the Resource Navigator view, the default editor associated with that file opens in the editor area of the current perspective. By default, the editor area is in the top-right hand side of the workbench window.

You are advised to open resources with the default editor only, because other editors might not correctly validate the changes you make.

The following figure shows the TextMessenger.msgflow file from the Pager samples opened in the Message Flow editor, which is part of the Broker Application Development perspective.



You can open any number of editors at once, but only one editor is active at any one time. The main menu bar and main toolbar display the operations that apply to the active editor. By default, editors are stacked in the editor area, but you can choose to tile them to view source files simultaneously. Tabs in the editor area indicate the names of the resources that are open for editing. An asterisk (*) indicates that an editor has unsaved changes. If you attempt to close the editor or exit the workbench with unsaved changes you are prompted to save the changes.

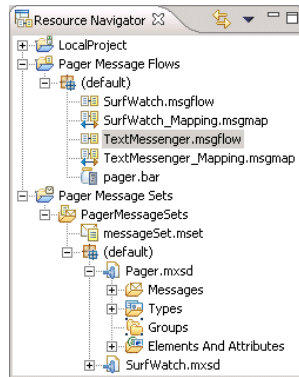
Resources

The projects, folders and files that exist in the workbench are called *resources*. They are collectively referred to as the *workspace* and they reside in your local file system.

By default, resources reside (with their metadata) in the workspace directory. The workspace directory is created the first time that you start the Message Brokers Toolkit. On Windows, the default workspace directory is created at C:\Documents and Settings\user\IBM\wmbt6.0\workspace, and on Linux, the default workspace directory is created at /home/user/IBM/wmqi6.0/workspace, where *user* is the user name with which you are logged on. You can create projects in other directories as well as in the workspace directory. You can maintain multiple workspaces by specifying a new location when prompted while the Message Brokers Toolkit is starting.

Typically, you open and view for editing any workbench resource in the Resource Navigator view. The exception is a broker domain resource (also referred to as a *domain object*) which you view and open for editing in the Broker Administration Navigator view in the Broker Administration perspective.

The following figure shows a server project, a message flow project, and a message set project in the Resource Navigator view of the Broker Application Development perspective.



Types of resource: There are three basic types of resource:

Files These are comparable to files in a file system. Different types of resource are maintained in different file types, for example, message flow files, message set definition files, ESQL files, mapping files, XML Schema files, and broker archive files. File types are listed in “Resource types in the Message Brokers Toolkit” on page 43.

Folders

These are comparable to directories in a file system. Folders are contained within projects or other folders. Folders can contain files and other folders.

Projects

Projects contain folders and files. Projects are used for building, version management, sharing, and resource organization. Like folders, projects map to directories in a file system. When you create a project, you specify a location for it in the file system. By default, projects are created in the workspace directory..

A project is either open or closed. You can view and modify an open project in the workbench. You cannot view or modify a closed project. The files and folders of a closed project are not displayed in the workbench, but they still reside on the local file system.

A closed project requires less memory than an open project.

You can:

- Add a project to the workbench.
- Save and version a project in an external repository.
- Copy a project from another user’s workbench.

Project types are listed in “Resource types in the Message Brokers Toolkit” on page 43.

By name linking: Previously, objects such as nodes in a message flow, or an execution group, were identified by a system-generated identifier called a *universally unique identifier (uuid)*. When an object was removed, renamed or deleted, the reference to the object was lost.

In WebSphere Message Broker, users identify objects using a combination of a *namespace* and a name, referred to as a *fully qualified name*. The use of fully qualified names makes it easy to identify and locate objects, and to correct broken references. For example, if you rename an object, all references to that object are broken. If you substitute another object with the same name, all the broken references are corrected. This concept, called *by name linking*, is particularly

important when you are working in a team environment. It means that you can share files in a repository and concurrently modify, add, and delete objects in your message flow application. When you integrate the various parts of the application, you can detect and resolve broken references to objects that have been moved, renamed or deleted.

Project references: A *project reference* is a property that you can set for a project so that it can be referenced by another project. When one project references another, the files in the referenced project are available for reuse.

For example, you might want to create a library of reusable ESQL subroutines in a project. Or you might want to create a similar library of message flows to reuse in other message flows. By adding a project reference to the library project, its subroutines or subflows are available for you to use.

Another use of project references is to enable Content Assist in the ESQL or Mapping editor. (“Content Assist” is context-sensitive help that displays valid ways in which a code statement can be completed.) For example, if you set up a project reference from a project containing ESQL code to a project containing a message set, the ESQL editor is able to display a list of valid message references.

If you subsequently close or delete a referenced project, or delete an object within it, it is no longer available to the referencing project and an error is generated. You can correct the error by opening the closed project or adding the missing object, with the correct name, and saving.

Development repository

You can use any repository supported by Eclipse with WebSphere Message Broker. Many repositories provide features such as version control and access control of files, and make it easier for teams to work on shared resources.

You can take the version numbers of these artefacts out of the repository, and associate that version number with the message flows and message sets when they are deployed. This allows you to display which version of the flow is deployed in the workbench. An alternative method of assigning a version number to a message flow is to specify one in the workbench when the message flow is created, see Message flow version and keywords.

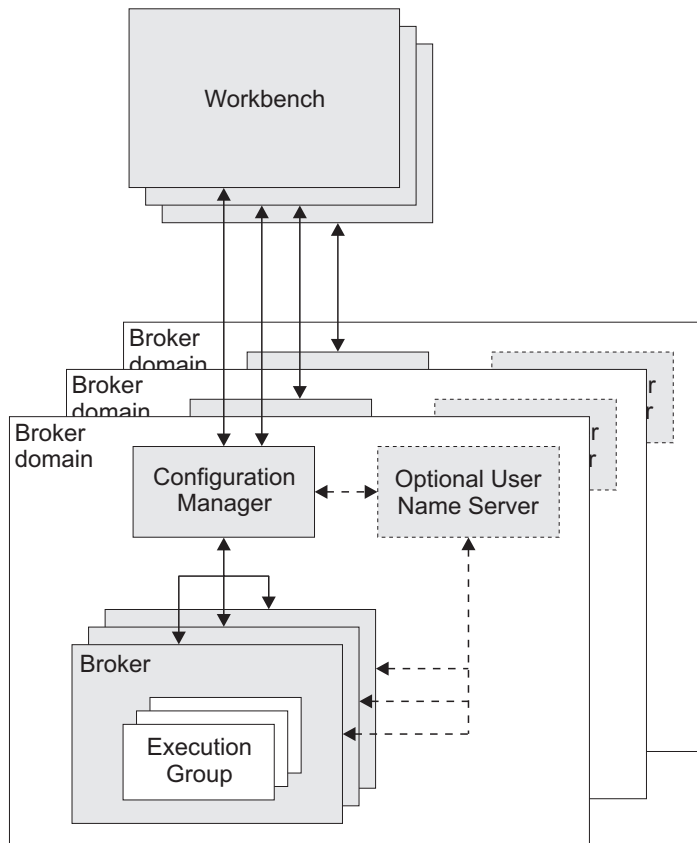
Because the Message Brokers Toolkit is based on the Eclipse platform, you can access Eclipse-supported repositories directly from the Message Brokers Toolkit.

Configuring CVS to run with the Message Brokers Toolkit describes how you set up the Message Brokers Toolkit to run with CVS. For more information about other repositories, refer to the section “Repository Providers” in the Eclipse developer community.

Runtime environment

The runtime environment is the set of resources that exist at run time, including the Configuration Manager, brokers, and execution groups.

The following figure shows the relationship between the resources that exist at run time, and how they interact with the workbench.



Follow the links below for more information about each of the runtime components:

- “Brokers” on page 28
- “Execution groups” on page 30
- “Broker domains”
- “Configuration Manager”
- “User Name Server” on page 30

Broker domains

A broker domain is one or more brokers that share a common configuration, together with the single Configuration Manager that controls them.

You install, create, and start one or more brokers, and an optional User Name Server, in a broker domain. You can configure more than one broker domain, each managed by its own Configuration Manager.

The Configuration Manager uses an internal repository to store and share information about the resources and components in the domain.

You create the physical components of a broker domain using command line instructions, and then configure and administer the broker domain using the Broker Administration perspective in the workbench.

Configuration Manager

The Configuration Manager is the interface between the workbench and an executing set of brokers. It provides brokers with their initial configuration, and

updates them with any subsequent changes. It maintains the broker domain configuration. The Configuration Manager is supported on all the broker platforms.

The Configuration Manager is the central runtime component that manages the components and resources that constitute the broker domain.

The Configuration Manager has four main functions:

- Maintains configuration details in an internal repository. This internal repository provides a central record of the broker domain components.
- Deploys the broker topology and message processing operations in response to actions initiated through the workbench. Broker archive (.bar) files are deployed through the Configuration Manager to the execution groups within a broker.
- Reports on the results of deployment and the status of the broker.
- Communicates with other components in the broker domain using WebSphere MQ transport services.

You must install, create, and start a Configuration Manager for each broker domain. You can install, create, and start more than one Configuration Manager on one or more machines.

When you create a Configuration Manager, you must give it a name that is unique within the broker domain; names cannot be shared between Configuration Managers, or between Configuration Managers and brokers.

You can install and configure the Configuration Manager runtime component on AIX, HP-UX, Linux, Solaris, Windows, and z/OS. For more information about supported platforms see Operating system requirements.

You administer one or more Configuration Managers in the Broker Administration perspective in the workbench. More than one instance of the workbench can access a Configuration Manager concurrently.

The Configuration Manager can share a host queue manager with one broker in the broker domain. To communicate with other brokers in the broker domain, the Configuration Manager requires sender and receiver channels.

Resources associated with a Configuration Manager: When you create the Configuration Manager runtime component, the following resources are also created:

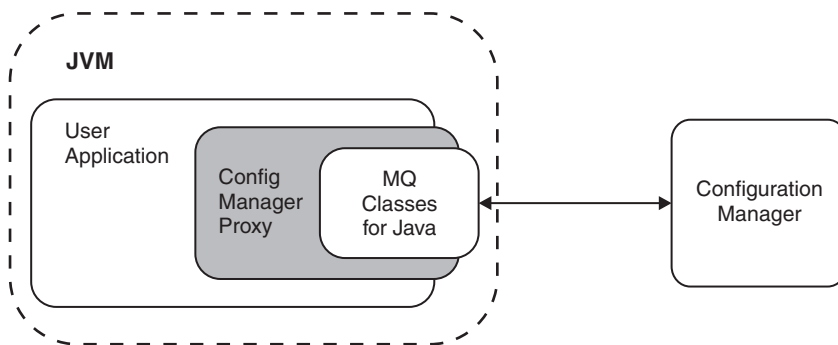
- An internal repository.
- A WebSphere MQ queue manager (unless one exists already) on the same physical system as the Configuration Manager. A queue manager can be shared with a single broker.
- A set of fixed-name queues, defined to the queue manager that hosts the Configuration Manager.

Configuration Manager Proxy: The Configuration Manager Proxy (CMP) is an application programming interface that your applications can use to control broker domains through a remote interface to the Configuration Manager.

Your applications have complete access to the Configuration Manager functions and resources through the set of Java classes that constitute the CMP. For example, you can use the CMP to interact with the Configuration Manager to:

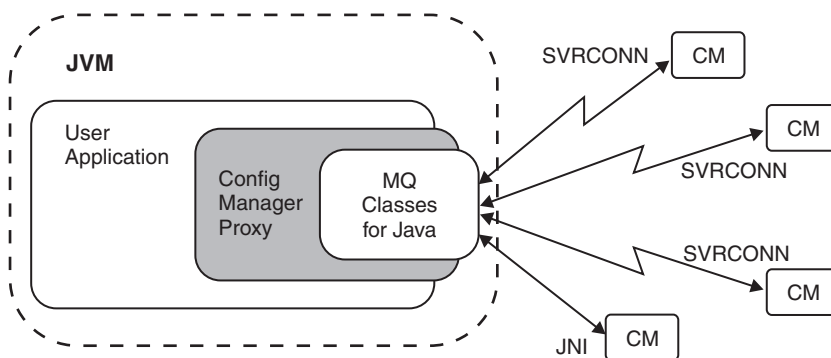
- Deploy BAR files, Publish/Subscribe topology, topic trees and broker configuration.
- Modify the Publish/Subscribe topology; add and remove brokers, broker connections and collectives.
- Create, modify, and delete execution groups
- Enquire and set status of objects in the domain, for example, run state, and be informed if status changes.
- Manipulate the topics hierarchy.
- View the broker event log and active subscriptions table.
- Modify domain Access Control Lists, when connected to Version 6.0 Configuration Managers only.

The CMP is a lightweight set of Java classes that sit logically between the user application and the Configuration Manager, inside the Java Virtual Machine (JVM) of the user application. It requires the WebSphere MQ Classes for Java in order to function, as shown below.



The CMP application can be on the same physical machine as the Configuration Manager (JNI to the queue manager using the WebSphere MQ Java Bindings transport) or distributed over a TCP/IP network (a WebSphere MQ SVRCONN channel using the WebSphere MQ Java Client transport).

It is possible for the CMP to communicate with multiple Configuration Managers from within the same application:



Using the API it is possible to connect to and manipulate Configuration Managers of the following products:

- IBM WebSphere Business Integration Event Broker Version 5.0
- IBM WebSphere Business Integration Message Broker Version 5.0
- IBM WebSphere Business Integration Message Broker Version 5.0 with Rules and Formatter Extension
- WebSphere Event Broker Version 6.0
- WebSphere Message Broker Version 6.0

A domain controlled by a Version 5.0 Configuration Manager can consist of Version 2.1 and Version 5.0 brokers, to which either version can be deployed by the CMP. Note also, that although it is only possible to run one Version 5.0 Configuration Manager on each physical machine, a single CMP application can still connect to multiple Version 5.0 Configuration Managers.

Brokers

A broker is a set of execution processes that hosts one or more message flows to route, transform, and enrich inflight messages.

Applications send messages to the broker using WebSphere MQ queues and connections. The broker routes each message using the rules defined in message flows and message sets, and transforms the data into the structure required by the receiving application.

The broker uses sender and receiver channels to communicate with the Configuration Manager and other brokers in the broker domain.

You install the broker component on the platforms described in Operating system requirements. If you have installed the broker component, you can also install the Message Transformation Services, which are a set of message processing services that extend the capabilities of the broker. For example, the Message Transformation Services provide additional message processing nodes, additional input and output nodes, and message flow debug capabilities. You create the broker using command line instructions on the machine where the component is installed. You can install and create one or more brokers, on one or more machines.

The broker depends on a broker database to hold broker information. This includes control data for resources defined to the broker, for example deployed message flows. You need to define a database and authorize access for specific users before you create the broker because creating the broker creates tables within the database. The database is also known as the broker's local persistent store.

The broker connects to the database using an ODBC connection.

When you create a broker, you must give it a name that is unique within the broker domain. Broker names are case-sensitive on all supported platforms, except Windows platforms. You must use the same name when you create a reference to the broker in the broker domain topology in the workbench. The reference to the broker is a representation of the physical broker in the configuration repository.

When you have created the broker reference, you deploy the changes to your broker domain. Deployment starts communications between the broker and the Configuration Manager. The broker receives configuration information from the Configuration Manager, and stores it in the configuration repository. Deployment also initializes the broker to make it ready to execute message flows.

Resources associated with a broker: When you create a broker, the following resources are also defined and created:

- A set of database tables for storing the information used by the broker to process messages at run time.

You can use one of a number of database products to create the database tables, depending on which platform the product is installed. See Supported databases for more information.

You must create a DB2 database if you are using WebSphere Message Broker for z/OS.

- A WebSphere MQ queue manager, if one does not already exist.
- A set of fixed-name queues that are defined to the WebSphere MQ queue manager.

System management interfaces:

The brokers provide a service for independent system management agents.

This enables a central management facility to access information about any network that includes a WebSphere Message Broker broker domain.

This support ensures that existing system management agents can be extended to include WebSphere Message Broker resources.

WebSphere Message Broker brokers publish event messages, using fixed topics, in response to configuration changes, state changes, and user actions such as subscription registrations. WebSphere Message Broker also uses architected messages to publish events related to the status, and change in status, of the brokers. These messages are published using the reserved topic root \$SYS in code page 1208.

An example of a fixed topic is:

```
$SYS/Broker/<brokerName>/Status/ExecutionGroup/<executionGroupName>
```

So the topic structure is fixed in this case, but obviously <brokerName> and <executionGroupName> are replaced with the appropriate values.

An example of the actual message data for this publication is:

```
<Broker uuid="12345678-1234-1234-1234-123456789012">  
  <ExecutionGroup uuid="12345678-1234-1234-1234-123456789012">  
    <Stop>  
      <AllMessageFlows/>  
    </Stop>  
  </ExecutionGroup>  
</Broker>
```

A system management agent can subscribe to these topics, or to a subset of these topics, to receive the detailed information about activity and state changes in the WebSphere Message Broker broker domain.

The event messages have a fixed structure, defined in *XML (Extensible Markup Language)*. The format of these messages, constructed in XML, is detailed in The XML message body. The messages cover configuration changes, state changes, error notifications, and detailed subscription and topic information (for example, a subscription registration).

You can develop or purchase system management adapters or customized administrative applications. These subscribe to the system management topics generated by WebSphere Message Broker to receive information on the broker domain activity.

Execution groups: An execution group is a named grouping of message flows that have been assigned to a broker. The broker enforces a degree of isolation between message flows in distinct execution groups by ensuring that they execute in separate address spaces, or as unique processes.

Each execution group is started as a separate operating system process, providing an isolated runtime environment for a set of deployed message flows. A single default execution group is set up ready for use when you create a reference to a broker in the workbench. By setting up additional execution groups, you can isolate message flows that handle sensitive data such as payroll records, or security information, or unannounced product information, from other non-sensitive message flows.

Within an execution group, the assigned message flows run in different thread pools. You can specify the size of the thread pool (that is, the number of threads) that are assigned for each message flow by specifying the number of additional instances of each message flow.

If you create additional execution groups, you must give each group a name that is unique within the broker, and assign and deploy one or more message flows to each one.

Execution groups are created and deployed in the workbench.

User Name Server

The User Name Server is an optional runtime component that provides authentication of users and groups performing publish/subscribe operations.

If you have applications that use the publish/subscribe services of a broker, you can apply an additional level of security to the topics on which messages are published and subscribed. This additional security, known as topic-based security, is managed by the User Name Server. It provides administrative control over who can publish and who can subscribe. For example, if a client application publishes messages containing sensitive company finance information, or personnel details, the User Name Server can be used to restrict access to those messages.

The User Name Server interfaces with operating system facilities to provide information about valid users and groups in a broker domain.

You install, create, and start a User Name Server in any supported operating environment. These are listed in Supported processors and Operating system requirements.

The User Name Server can share a host queue manager with the Configuration Manager and one broker in the broker domain. To communicate with other brokers in the broker domain, the User Name Server requires sender and receiver channels.

Resources associated with a User Name Server: When you create a User Name Server, the following resources are also created:

- A WebSphere MQ queue manager unless one exists already. The queue manager is created on the same physical system as the User Name Server. A queue manager can be shared with the Configuration Manager, a single broker, or both.
- A set of fixed-name queues, defined to the WebSphere MQ queue manager that hosts the User Name Server.

WebSphere Message Broker scenario

WebSphere Message Broker manages the flow of information in your business, applying business rules to route, store, retrieve, and transform the information as required by the different systems in your business and the systems of your customers, suppliers, partners, and service providers.

The following scenario outlines an example of how WebSphere Message Broker can be used to solve IT infrastructure problems in a business.

Mergers and acquisitions scenario

This scenario describes how WebSphere Message Broker is used by a fictitious insurance company to manage two disparate IT infrastructures after a small, Internet-based insurance company is acquired by a large, more traditional insurance company. The description focuses on what happens when a potential customer requests a motor insurance quotation using the merged company's Web site. This scenario is based on a larger, more complex scenario that was published on developerWorks. To read the full scenario see the links at the end of the scenario description.

Background

Company A is a motor and general insurance company that has been in business for about 50 years and currently has about 5 million policyholders. The company uses agents and a call center to communicate with customers. The company has a large legacy IT infrastructure, which includes CICS Transaction Server on z/OS and IBM DB2 Universal Database on z/OS.

Company B is small Internet-based motor insurance company, which currently has less than one million policyholders but is expanding. The company's IT infrastructure includes WebSphere Application Server on Windows 2000, and Oracle Enterprise and IBM DB2 Universal Database on Windows 2000.

Problems

Company A acquired Company B to gain access to the Internet-based insurance market and to use Company B's Internet-based skills and IT infrastructure. The two companies have customer and policy data of different formats but, for legal reasons, the data from the separate companies cannot be merged. However, the administration costs of managing the separate IT infrastructures are high. Also, customers, agents, and call center staff need to have a single administration process to interact with the company's data.

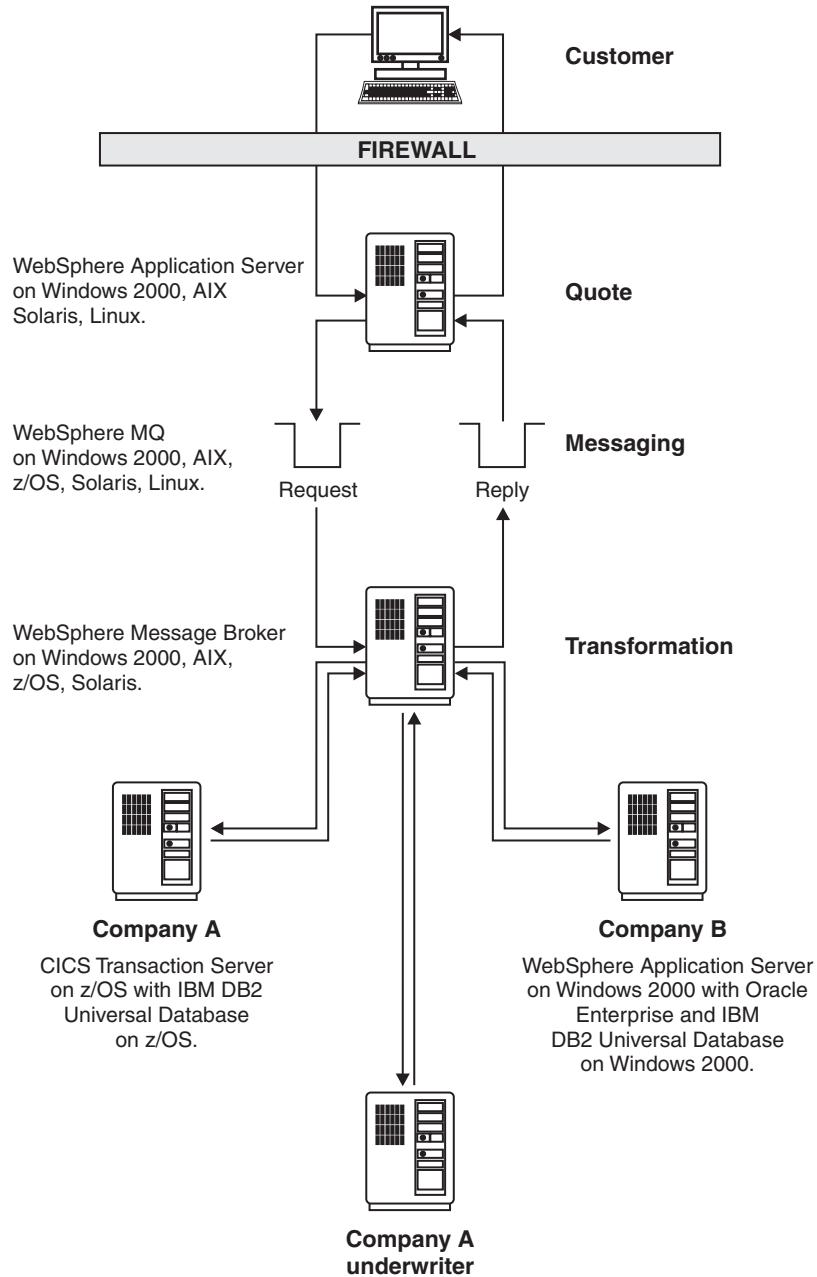
Solution

Now that the two companies have merged, users can request an insurance quotation by giving some basic personal information in a form on the new company's Web site. WebSphere Application Server, on which the Web site runs, forwards the request in XML format to WebSphere Message Broker using the

request queue in a WebSphere MQ cluster. WebSphere Message Broker transforms the XML request to the legacy COMMAREA format that is used by Company A's systems, then routes the request to Company A's systems. WebSphere Message Broker also routes the request, in XML format, to Company B's systems. Both systems return a quotation to WebSphere Message Broker.

Logic within WebSphere Message Broker also requests a risk assessment from the internal underwriter and applies the returned risk to the quotations from Company A's and Company B's systems. The broker detects that, in this instance, the best or lowest quotation for the customer has been generated by Company A's systems. So the broker transforms Company A's quotation from COMMAREA to XML and routes the quotation back to WebSphere Application Server using a reply queue in the WebSphere MQ cluster where the quotation is stored for up to 14 days. WebSphere Application Server returns the quotation to the customer.

The following diagram illustrates the flow of information in this scenario.



Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. When you use the administrative facilities provided by WebSphere Message Broker, you can use your operating system's accessibility features to modify the behavior of the user interface.

On Windows systems you can change the key behavior, provide a high-contrast display, or control the pointer with keys instead of a mouse. You can enable accessibility features from Accessibility Options within the Control Panel.

On Linux (x86 platform) systems the way in which you control accessibility features depends on the gtk toolkit which underlies the Gnome windowing system. Under Gnome, you can change the key behavior, or control the pointer with keys instead of a mouse. From the main menu click **Applications** → **Desktop Preferences** → **Control Panel** → **Accessibility**. In addition, there are Screen Reader and Magnifier options available if Assistive Technologies is enabled under Gnome. Alternatively, you can use the tooling under other windowing systems such as KDE, in which case the appropriate look and feel will be seen, however the accessibility features of KDE (primarily key behavior) do not apply.

Part 2. Reference

Workbench	37
Perspectives in the Message Brokers Toolkit	37
Broker Administration perspective.	37
Broker Application Development perspective	39
Debug perspective	42
Plug-in Development perspective	43
Resource types in the Message Brokers Toolkit	43
Message flow projects and files.	44
Message set projects and files	44
Plug-in Development projects and files	45
Rules for naming workspace objects	46
Editors in the Message Brokers Toolkit	46
Editor preferences and localized settings.	47
Broker Archive editor	47
Broker Topology editor	48
ESQL editor	49
Event Log editor	52
Message Category editor	53
Message Definition editor	54
Message Flow editor	58
Message Mapping editor	64
Message Node editor	65
Message set editor	65
Subscriptions Query editor	66
Topics Hierarchy editor	67
Message Brokers Toolkit keyboard shortcuts	68
Glossary of terms and abbreviations	71

Workbench

Follow the links below for reference information about the Message Brokers Toolkit integrated development environment:

- Perspectives
- Resource types
- Editors
- Keyboard shortcuts

A minimum display resolution of at least 1024 x 768 is required for some dialogs (for example, the Preferences dialog).

Perspectives in the Message Brokers Toolkit

Follow the links below for a description of each of the perspectives in the Message Brokers Toolkit:

- “Broker Administration perspective”
- “Broker Application Development perspective” on page 39
- “Debug perspective” on page 42
- “Plug-in Development perspective” on page 43

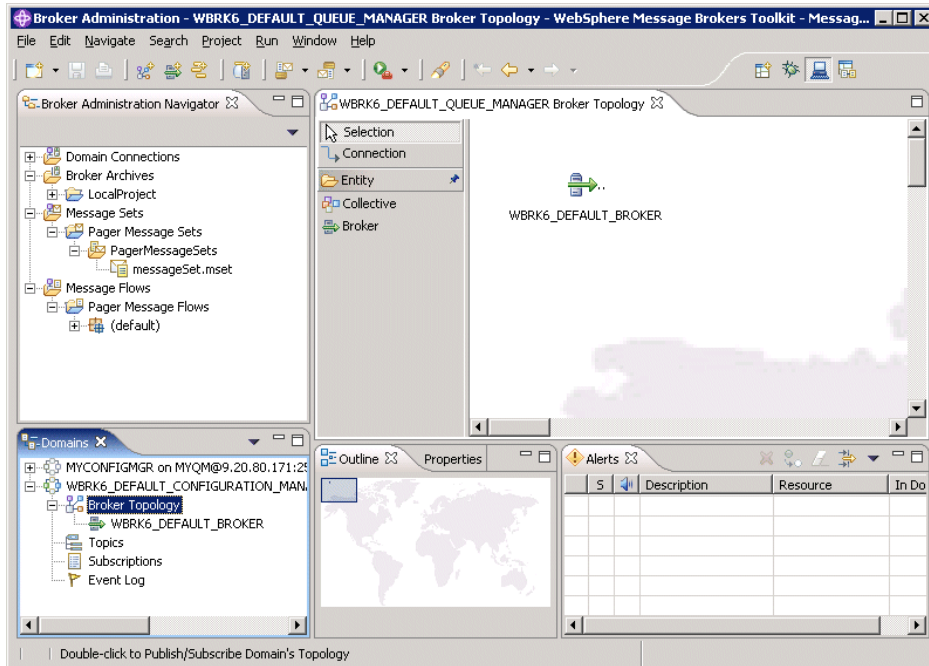
Broker Administration perspective

This is where the broker domain resources (also referred to as *domain objects*) that are defined on one or more Configuration Managers are managed.

Typically, you carry out the following tasks in this perspective:

- Setting up the broker domain, for example creating collectives
 - Creating and removing domain connections.
 - Connecting and removing brokers.
- Adding and removing execution groups.
- Creating and deploying broker archive (bar) files.
- Defining and managing publish/subscribe topic hierarchies.
- Querying, viewing and deleting subscriptions.
- Clearing and filtering deploy event log entries.
- Activating, deactivating, filtering, and clearing alerts.
- Adding, removing, and updating entries in access control lists (ACLs).

The following figure shows the Broker Administration perspective of the Message Brokers Toolkit. The Domains view in the figure shows that the Message Brokers Toolkit is connected to the Default Configuration.



The Broker Administration perspective provides several views that allow you to navigate and update your broker domain resources.

Broker Administration Navigator view

This is where you view and work with any of the broker domain resource files. You can also deploy and delete any resource file that has been created in the Broker Application Development perspective.

The broker domain resource files are contained in the following folders:

Domain Connections

Contains the domain connections that are defined to multiple local and remote Configuration Managers.

Broker Archives

Contains the broker archive (bar) files.

Message Sets

Contains the message set (mset) files.

Message Flows

Contains the message flow (msgflow) files.

Domains view

This is where you view and work with any of the broker domain objects listed below, when a broker domain is connected to a Configuration Manager:

- Broker Topology
- Topics
- Subscriptions
- Event Log entries

You can manage deployed configurations in this view. Double-clicking an object launches the appropriate editor in the editor view. You can also deploy bar files on execution groups, and start and stop message sets and message flows.

By default, an alert icon is displayed for a domain object that is connected to a Configuration Manager but is not in a normal running state. You can choose not to display the alert icon using an option that is available in the Alerts view.

Editor view

You can open any broker domain resource to launch its default editor in the editor view. The following default editors are available:

Broker Topology editor

For administering the brokers and collectives within a broker domain.

Topics Hierarchy editor

For managing topics within a publish/subscribe system.

Subscriptions Query editor

For viewing and deleting subscriptions within a publish/subscribe system. Subscriptions that use the WebSphere MQ Real-time Transport are not displayed in the Subscriptions Query editor.

Event Log editor

For viewing, clearing and filtering event log entries.

Properties view

Displays the property names and values for the domain object selected in the Broker Administration Navigator view.

Alerts view

An alert is displayed in the Alerts view when a domain object is connected to a Configuration Manager but is not in a normal running state. The normal running state for a domain object is started, without a user trace or any other special processing enabled.

The Alerts view is updated continuously whilst the workbench is connected to a Configuration Manager.

You can filter, clear, and remove alerts from this view. You can also quieten alerts. This removes the alert icon from every object in the hierarchy in the Domains view, except for the alerted object and its parent.

When a new object is added to the domain it is selected in the Alerts Filter by default. Therefore, you might want to modify the selection of objects in the Alerts Filter after a new object has been added to the domain by you or anyone else.

Broker Application Development perspective

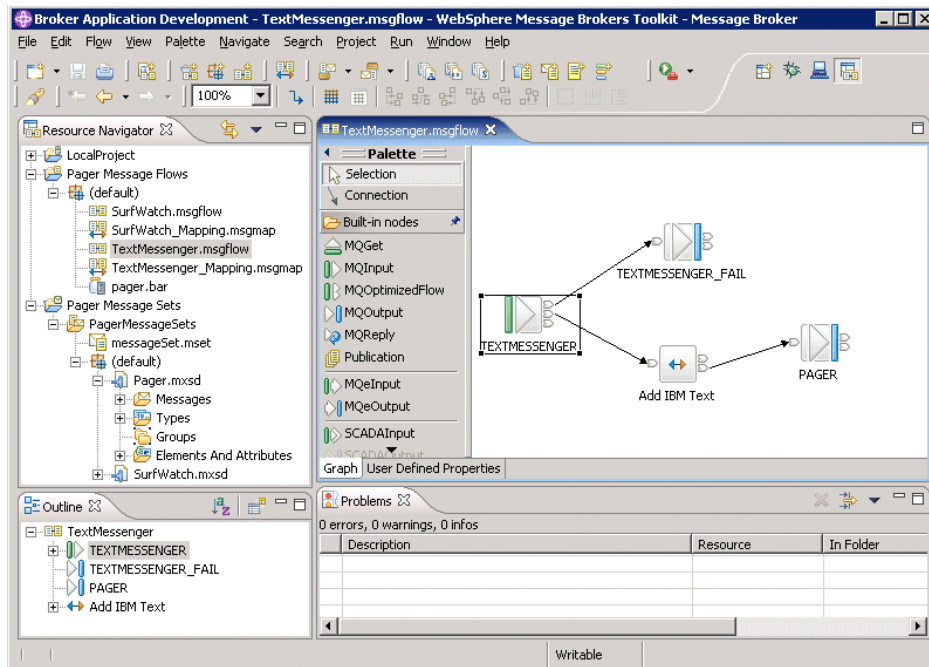
The Broker Application Development perspective is the default perspective that is displayed when you start the Message Brokers Toolkit.

Typically, you carry out the following tasks in this perspective:

- Develop message sets and message flows.

- Enqueue and dequeue test and production messages. This is particularly useful when you are debugging message flows.

The following figure shows the Broker Application Development perspective of the Message Brokers Toolkit. The TextMessenger.msgflow file in the Resource Navigator view is open in the Message Flow editor.



The Broker Application Development perspective provides several views that allow you to navigate, browse, and update your application resources.

Navigator view

This view displays a hierarchy view of the following message flow and message set resource files:

- Message flow projects
- Message flows
- ESQL files
- Mappings files
- Message set projects
- Message sets
- Message definition files

You can open any of these resources for editing using specialized editors, except project files.

Editor view

Editors are launched in the editor area when you double-click a resource in the **Resource Navigator** view.

The following specialized editors are provided by default by Message Brokers Toolkit for working with the content of message flow and message set files:

ESQL editor

For viewing and updating the ESQL statements associated with Compute, Database, and Filter nodes.

Message Category editor

For creating, viewing, and updating message categories that you define within the MRM domain.

Message Definition editor

For creating, viewing, and updating messages that you define within the MRM domain.

Message Flow editor

For creating, viewing, and updating message flows, and the nodes that define them.

Message Mapping editor

For defining transformations between data sources and targets, without the need for programming in XPath, XSLT, XQuery, Java, or ESQL.

Message Set editor

For creating, viewing, and updating message sets that you define within the MRM domain.

Outline view











The Outline view provides a summary of the content of the resource that you currently have open in the Editor view.

Tasks view

The Tasks view displays any messages (information and error) that are associated with the resource that you currently have open in the Editor view. For example, if you save a message flow that has an error (such as a mandatory property not set), you can check the content of the Tasks view to determine any corrections that you need to make. When you double-click a Tasks view entry, the appropriate editor opens the resource in error, and positions the cursor at the point of the error (where possible).

Broker Application Development perspective toolbar

The icons in the toolbar and their actions are shown in the table below.

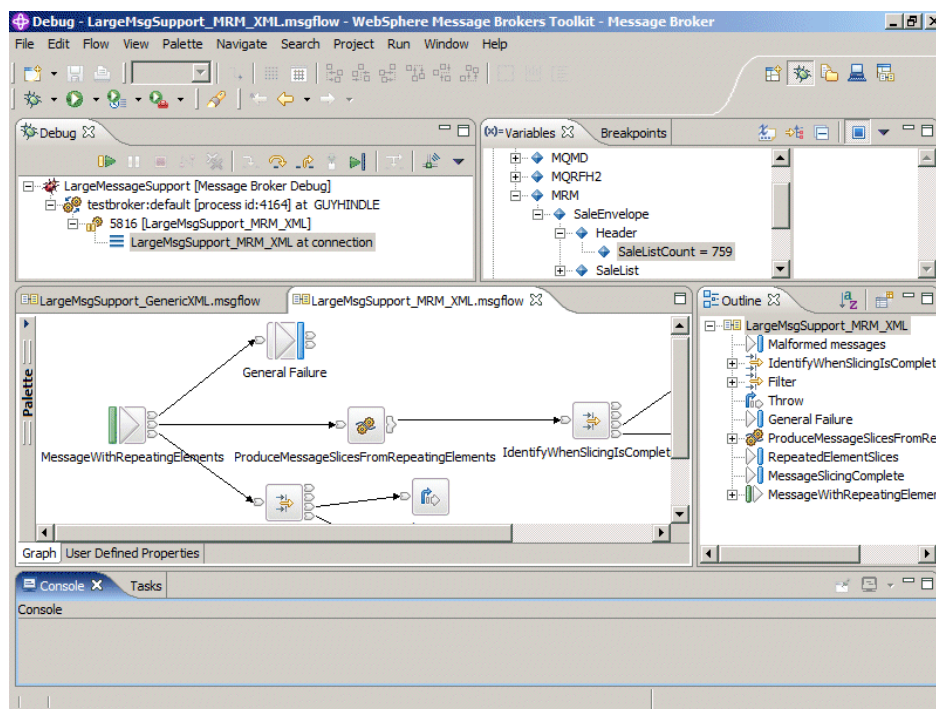
Icon	Label and action
	Get a message from a Queue
	Put a message to a Queue
	Create a New Message Flow Project
	Create a New Broker Schema
	Create a New Message Flow
	Create a New Message Flow ESQL File
	Create a New Message Flow Mapping File
	Generate a Web Services Definition from a Message Set
	Generate HTML Documentation for a Message Set
	Generate an XML Schema from a Message Definition File

Icon	Label and action
	Create a New Message Set Project
	Create a New Message Set
	Create a New Message Definition File
	Create a New Message Category File

Debug perspective

This is where you test and debug a graphical representation of your message flows using the message flow debugger.

The following figure shows the Debug perspective of the Message Brokers Toolkit. In the figure, the LargeMsgSupport_MRM_XML message flow is being debugged.



Debug perspective views

The Debug perspective contains the following views:

Debug view

Displays the deployed message flow types for a selected host to help you manage flow debugging. Toolbar buttons are provided for controlling the execution of the flow. You can start, stop, and resume a flow, step into, and out of a subflow, and step into the source code.

Breakpoints view

Lists the breakpoints that have been set on connections in your message flow. In this view you can add, disable, enable or remove breakpoints. You can also restrict a breakpoint to one or more specific instances of a message flow using the Properties view.

Variables view

When a message flow is interrupted by a breakpoint, you can view the

message content to check whether the message flow is executing as expected, and to make any changes required.

Message Flow editor view

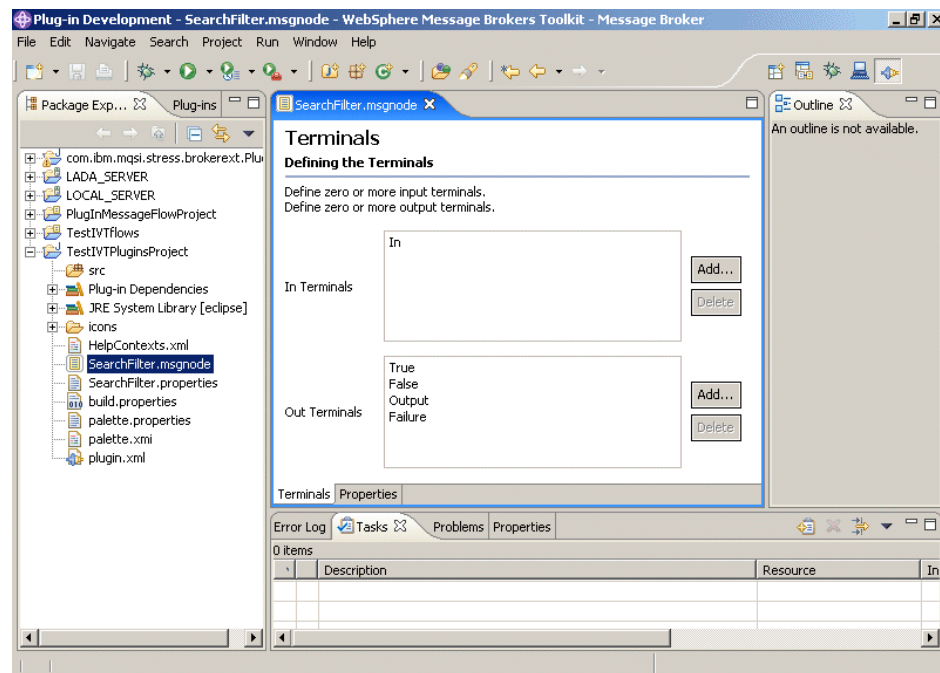
Here you can add breakpoints to the connections of a message flow that is open in the Message Flow editor.

Plug-in Development perspective

This perspective helps you develop user-defined nodes in the Message Brokers Toolkit. A user-defined node is an Eclipse plug-in that adds a category of nodes to the Message Flow editor palette.

The New Plug-in Project wizard creates the supporting workbench files for your user-defined node. The New Message Flow Plug-in Node wizard creates a message node file (.msgnode) and launches the Message Node editor for developing the visual representation of your user-defined node in the Message Brokers Toolkit.

The following figure shows the Plug-in Development perspective in the Message Brokers Toolkit.



Resource types in the Message Brokers Toolkit

Follow the links below for reference information about the projects and resource files specific to Message Brokers Toolkit:

- “Message flow projects and files” on page 44
- “Message set projects and files” on page 44
- “Plug-in Development projects and files” on page 45 (for user-defined nodes and parsers)

Message flow projects and files

You develop message flows in the Broker Application Development perspective. Message flow resources are maintained within message flow projects. A message flow project compiles to one or more deployable flows.

Message flow projects contain the following resource files:

.msgflow files

A message flow file is a graphical representation of a message flow, containing message nodes and connections. It also contains property values and overrides to define, implement and control the behavior of the message flow nodes. The New Message Flow Wizard automatically creates this file.

Default editor: Message Flow editor

.esql file

(Optional) ESQL modules containing compute, filter, and database ESQL functions and procedures. If none of the message flows defined in the project include any nodes that require ESQL, you do not need to create an ESQL file.

Default editor: ESQL editor

.msgmap file

(Optional) Mapping transformation descriptions for messages and databases. If none of the message flows defined in the project include any nodes that require mappings, you do not need to create a mapping file.

Default editor: Message Mapping editor

You can define more than one message flow, each in a separate `.msgflow` file, in the same message project.

Message set projects and files

A message set project contains all the resources associated with exactly one message set. A message set is a logical grouping of messages and the objects (elements, types and groups) that comprise them. It contains the following resources, which you create and maintain in the Broker Application Development perspective.

messageSet.mset file

A message set file contains those message model properties that are common across all the content of the message set. It also contains the physical format definitions for the message set. A message set project must contain exactly one message set file.

Default editor: "Message set editor" on page 65

.mxsd files

A message definition file contains, in XML Schema form, the logical model and associated physical model for one or more messages. Each message set requires at least one message definition file to describe its properties, and you can have as many as you want within the same message set. One message definition file can, if required, reference XML Schema objects in another message definition file.

You can create a message definition file in a message set by importing any of the following data structures:

- XML Schemas
- XML DTD
- C header files
- COBOL copybooks

Importing any of these data structures automatically creates the message definition file and its content for you. Alternatively, you can create a blank message definition file and add the message definitions yourself.

Default editor: “Message Definition editor” on page 54

Previous versions of the message set model

The message definition contains the same application data that, previously, was stored in the following separate files:

- Message set (MRProject) that contains wire formats and their default properties
- Messages (MRMessage) that provides a unique name for a message and specifies its element
- Elements (MRElement) that defines a node in the message tree
- Types (MRType) that specify the structure of elements
- Declaration Qualifiers (MRDeclaration Qualifier) that specify constraints applied to elements in the message model

.category files

A message category file provides you with another way of grouping related messages, for example for documentation generation, or for convenience purposes such as specifying the messages that define a complete request/reply transaction. You can also use a message category file to assist in generating a Web Services Description Language (WSDL) document. Message category files are optional, and you can have as many as you want within the same message set.

Default editor: “Message Category editor” on page 53

Once you have created and populated a message set, you can use it to generate your message model in several different representations for use by your applications:

- A message dictionary for deployment to the WebSphere Message Broker.
- A W3C XML Schema for use by an application that is building or processing XML messages.
- A Web Services Description Language (WSDL) document that specifies the interface for a Web Service.
- An HTML document for use by business analysts and developers.

Plug-in Development projects and files

A plug-in project can contain any Message Brokers Toolkit-developed plug-in resource. In WebSphere Message Broker, it is used for user-defined message nodes.

User-defined Message Nodes

Plug-in development projects used to create Message Nodes contain these resources:

.msgnode file

Contains the definition of the node.

palette.xmi file

Lists the user-defined nodes in the project and the Message Editor palette group in which they are shown.

.html file

Contains the help files for the node.

.java file

Java source code for Property Editors and Compilers.

plugin.xml file

The plug-in manifest file that adds the node to the WebSphere Message Broker installation

.properties file

National Language properties - contains the translations for Properties and Terminals

.gif file

Icon - contains the image to associate with the node

Rules for naming workspace objects

You can uniquely identify and locate a workspace resource using a namespace and a name. An organization uses its own name to form a namespace, and all resource names within that namespace are safe from clashes with those of another organization. The combination of a user specified namespace and name is often called the "fully qualified name". The unnamed namespace (name space="") is supported.

WebSphere Message Broker defines an application symbol space using project references. No fully qualified names may be duplicated across the application symbol space. A Tasks List error is displayed if duplicate or ambiguous names are detected within an application symbol space.

Editors in the Message Brokers Toolkit

Follow the links below for more information about each of the editors provided by Message Brokers Toolkit:

- "Broker Archive editor" on page 47
- "Broker Topology editor" on page 48
- "ESQL editor" on page 49
- "Event Log editor" on page 52
- "Message Category editor" on page 53
- "Message Definition editor" on page 54
- "Message Flow editor" on page 58
- "Message Mapping editor" on page 64
- "Message Node editor" on page 65
- "Message set editor" on page 65
- "Subscriptions Query editor" on page 66
- "Topics Hierarchy editor" on page 67

Editor preferences and localized settings

If users within the broker domain want to share the files that are associated with message flow and message set development, you must ensure that the files and the content of these files is compatible and can be shared by users working on different systems and in different locales.

Users can share the following files:

- Message flow definition files (.msgflow)
- ESQL files (.esql)
- Mappings files (.mfmap)
- Message set files (.mset)
- Message files (.mxsd and .xsd)
- Message category files (.category)

You are recommended to restrict the names of these files to a specific set that all your users can understand, for example the characters in the US English code page (a to z, A to Z, 0 to 9, plus special characters underscore and hyphen). This is not a restriction, nor is it enforced, but the users must be able to use each others resources without confusion or error.

The contents of all files listed above, with the exception of ESQL files, are stored in UTF-8 and are therefore common regardless of individual user settings.

The contents of the ESQL files, which users can edit directly, are governed by the editor preference settings that you specify in the workbench:

1. Click **Window** → **Preferences**.
2. Expand the Workbench item on the left. Click **Editors**. Within "Text file encoding", select the code page that you want content to be stored in:
 - a. If you select Default, this represents the default code page for the locale in which you are currently working.
 - b. If you select Other, you can choose one from the list of available code pages.

If you expect users to share ESQL files, either through the use of shared drives across a LAN, or through a shared repository such as CVS, you must ensure that all users set the editor preference to be the same value. This allows each user to work in their current locale, but to be able to access and work with files created by any other user. You are recommended to select UTF-8 to ensure consistent ESQL file content that is accessible by all users regardless of current locale.

Broker Archive editor

The Broker Archive editor is the component you can use to edit and save bar files. It is the mechanism that you use for configuring system object names contained within a bar file without recompiling a message flow. You can also use command line tools to create bar files.

The Broker Archive editor is available in the Broker Application Development perspective and has two sections; one that allows you to add or remove message flows and message sets to or from your broker archive, and one that lists the configurable properties within your bar file, and allows you to update them.

The Broker Archive editor only allows you to add message flows that are syntactically complete, that is, it will not allow you to add any flows that do not contain an Input node, or that contain errors such as mandatory properties not set correctly.

Broker Archive editor views

The Broker Archive editor includes these views:

Content view



The Content view displays the files that are currently in the broker archive file.

Configure view

The Configure view displays the configurable properties of your broker archive file deployment descriptor. To edit a property, replace the current value with a new value.

Broker Archive editor toolbar

The icons in the toolbar and their actions are shown in the table below.

Icon	Label	Action
	Add	To add message flows or message sets to this archive.
	Remove	To remove message flows or message sets from this archive.

A column in the bar editor called *Version* displays the version tag for all objects that have a defined version. These are:

- .dictionary files
- .cmf files
- Embedded JAR files with a version defined in a META-INF/keywords.txt file

You cannot edit the *Version* column.

You can use the `mqswireadbar` command to list the keywords defined for each deployable file within a deployable archive file.

Broker Topology editor

The Broker Topology editor is the default editor that is provided by the Broker Administration perspective for managing brokers and collectives, and for saving topology changes to the Configuration Manager in a single step.

The Broker Topology editor is launched in the editor area of the Broker Administration perspective when you double-click Broker Topology in the Domains view.

If you are not connected to a domain, an attempt to do so is made before opening the editor. During this attempt, if you click **Cancel**, the connection that is in progress stops and the domain returns to its initial unconnected state.

When the editor is launched, an overview pane is displayed in the Outline view that is located in the same place as the Properties view. Use the Properties tab or the Outline tab to select which of these two views is displayed.

Use the Broker Topology editor to do any of the following:

- Create, rename, or delete brokers, or change broker properties.

In the workbench, a broker is a representation of a physical broker that exists in the configuration repository. When you create or rename a broker in the Broker Topology editor, you must use the name of the physical broker that exists in the configuration repository.

- Connect brokers to collectives, remove connections between brokers and collectives, or delete collectives.

The rules for creating a broker topology are given in the Broker Networks section of Publish/subscribe topologies.

When you save topology changes to the Configuration Manager, the topology information in the Domains view on every workbench that is connected to the Configuration Manager is refreshed automatically.

For information about changing the palette preferences in Message Brokers Toolkit, see Changing palette preferences.

By default, the Broker Topology editor has a background image of a world map. You can change the background image, and modify its scale factor, or choose not to display an image.

ESQL editor

The ESQL editor is the default editor provided by the Broker Application Development perspective for editing ESQL (.esql) files.

The editor is launched in the editor area when you select the menu item **Open ESQL** for a Compute, Database, or Filter node, or when you double click an ESQL file in the navigator view.

ESQL editor views

The ESQL editor has the following views:

Resource Navigator view

The navigator view shows all the resources in your workspace, that is all message set resources and all message flow resources, including ESQL files.

Editor view

The editor view shows the contents of the resource that is currently open. It also shows tabs for each of the resource that you have open so that you can quickly switch between them.

Outline view

The Outline view displays any schemas, defined constants, modules, and routines that you have referenced in this ESQL file.

Tasks view

The tasks view displays the warning and error messages that are generated by the editor's validation when you save the ESQL file. If you double click an error, the editor indicates where it is located by highlighting the corresponding ESQL code.

ESQL editor functions

The ESQL editor provides:

- A context sensitive Content Assist. You can invoke Content Assist from the Edit menu or, on some systems, by pressing Ctrl+Space if that has not been assigned to another function.

Content Assist helps you construct references to the content of the Properties folder. When you use the ESQL editor with predefined messages, it also helps you construct field references.

When you use the ESQL editor with the database schema definitions, Content Assist helps you construct schema, table, and column references. You can also use the ESQL editor to call user-defined maps.

When you create functions and procedures within the ESQL file, the names that you define must not start with the characters IBM_ (IBM underscore).

Note: To get Content Assist to work, you must set up a project reference from the project containing the ESQL or mappings to the project containing the message set. For information about setting up a project reference, see “Project references” on page 24.

- Automatic code formatting.

Right-click in the editor view to access the following additional functions:

- **Undo** and **Revert**. To undo a change that you have made to the ESQL file, click **Undo**. If you undo a change, you can reinstate it by clicking **Revert**.
- **Cut**, **Copy**, and **Paste**. These are standard editor functions.
- **Shift Right** and **Shift Left**. These are standard editor functions.
- **Save**. Click this to save your changes.
- **Comment** and **Uncomment**. Click **Comment** to change a line of ESQL code into a comment. Click **Uncomment** to change a comment into a line of ESQL code.
- **Format**. This function formats all selected lines of code (unless only partially selected, when they are ignored), or, if no lines are selected, formats the entire file (correcting alignments and indentation).
- **Organize Schema Paths** and **Add Schema Path**. These functions assist you with broker schema management.

Click **Organize Schema Paths** and any broker schema containing procedures or function called by the ESQL file is automatically added to the PATH statement (if you have not already added it). This function scans the ESQL file for instances of procedures or function residing in schemas not already fully qualified in the file.

Click **Add Schema Path** when you code a call to a procedure or function residing in a different broker schema to any you have included on the PATH statement, and this schema is added to the PATH statement. Ensure that the cursor is on the name of the procedure you are calling.

ESQL editor preferences

You can modify settings that affect the way the ESQL code is handled:

- Code Generation Settings (what runtime code is generated):

Schema name used to access RDB tables

Select one of the following radio buttons to specify the schema that is used when you include a reference to a database table in your ESQL:

Use default runtime schema for this data source

If you select this, the default schema for the data source is used.

Use schema name in table definition

If you select this, the schema that you specified when you created the table definition is used.

Use name specified

If you select this, the schema name that you enter in the associated text entry field is used.

Default Compatibility Level

Select either 5.0 or 2.1 from the drop down list:

5.0 If you select this value, a compiled message flow can be deployed only to Version 5.0 brokers because the runtime ESQL code that is generated is not compatible with previous versions. If you want to debug ESQL code within your message flows, this setting is required. It is also required if your message flow includes nodes that have mapping (.mfmap) files. If you set this value, and you deploy a bar file that contains a message flow that includes Version 5.0 runtime ESQL code to a Version 2.1 broker, the broker generates an error when it starts the message flow.

This is the default setting.

2.1 You must select this value if you want to deploy your message flow to Version 2.1 brokers. You can also deploy these message flows to Version 5.0 brokers, but you must ensure that you do not include any content that is specific to Version 5.0 (for example, a mapping file or a BROKER SCHEMA statement). If you have set this value, and you deploy a bar file that contains a message flow that includes Version 5.0 runtime ESQL code, the deploy fails.

If you change this setting from 5.0 to 2.1, or from 2.1 to 5.0, you must rebuild every message flow project in your workspace. If you do not do so, you might receive errors when you deploy a bar file that contains one or more of these message flows.

- Editor Settings (how code is displayed in the editor view):
 - Text font
 - Displayed tab width (default 4)
 - Background and foreground colors (for comments, statements, and so on)
- Validation Settings (what level of validation is performed when you save the file):

Validation detects four potential problems:

Unresolved identifiers

The validator attempts to resolve any identifiers that you have referenced (for example, a message field).

Message references mismatch message definition

If a message definition exists (messages in the MRM domain only), the validator checks that the use of the reference is consistent with its definition (for example, the action against a numeric field is a valid numeric action).

Database references mismatch database schema

The validator checks that the use of the reference is consistent with the database schema (for example, the action against a numeric field is a valid numeric action).

Use of deprecated keywords

The validator checks if you have used any keywords that have been deprecated in this release.

For each of these situations, select one of the following validation settings:

Ignore No validation is performed.

Warning

The validator writes warning messages to the Tasks view for each potential problem that it detects. This is the default setting.

Error The validator writes error messages to the Tasks view for each potential problem that it detects.

Validation does not check that you have specified names in the case in which you declared them. The names of modules, functions, and procedures are not case sensitive; all other names (schemas, constants, variables, and labels) are case sensitive. Check that the names that you use match the declarations for those names because the broker handles these names in a case sensitive way and generates a runtime error if they do not match.

For details of how to change these preferences, see [Changing ESQL preferences](#).

ESQL editor toolbar

The ESQL editor does not provide any additional icons and actions on the toolbar.

Event Log editor

The Event Log editor is launched in the editor area of the Broker Administration perspective when you double-click the Event Log in the Domains view.

If you are not connected to a domain, an attempt to do so is made before opening the editor. During this attempt, if you click **Cancel**, the connection that is in progress stops and the domain returns to its initial unconnected state.

The event messages are stored and managed by the Configuration Manager. You can view the messages in the Event Log editor following configuration deploys.

The Event Log editor has two panes called Logs and Details. You can maximize and minimize each pane, and toggle between them.

Event Log editor views

Logs pane

The Logs pane displays a summary of each event in the Event Log editor, comprising a message identifier, the source of the message, for example the Configuration Manager or a broker, and the date and time the message was generated.

Right-clicking in the Logs pane displays the following menu items:

- **Clear Log**

Clears all log entries from the Event Log editor and deletes the log entries from the configuration repository. You cannot retrieve a message that has been cleared.

Messages are automatically cleared after 72 hours.

- **Filter Log**

Displays the Log Filter. You can filter on the type, the source and the timestamp of the message to restrict the number of log entries that are displayed in the Event Log editor. You can also mask events, event by event, to make the other entries easier to view in the Event log editor. The filter settings you define are kept for your next session.

- **Revert**

Refreshes the Event Log editor with the most recent log entries from the Configuration Manager. You only need to use this option if the workbench is not synchronized with the Configuration Manager.

- **Save Log As**

Saves the messages to a filename and path of your choice. **Save Log As** does not remove messages from the Event Log editor or the configuration repository.

Details pane

The Details pane displays the complete message, and any corrective action, for the log entry selected in the Logs pane.

Preferences

You can change the preferences for the Event Log editor. For example, you can customize the colors used for each type of event message.

Message Category editor

The Message Category editor is the default editor provided by the Broker Application Development perspective for working with message category (.category) files in a message set.

The editor is launched in the editor area when you open an existing message category file using the Resource Navigator or when you create a new message category file using the New Message Category File wizard.

Message Category editor views

The Message Category editor has the following views:

Resource Navigator view

The Resource Navigator shows a hierarchical view of all the resources that are currently in your workspace. By expanding the folder for a message set project, you can see the resources, including the message category file or files, that this message set project contains.

Editor view

You edit the properties of message category in the editor area of the Message Category editor, which has a single tab, the **Properties** tab. This tab provides:

- The Properties Hierarchy, which shows the hierarchy for a message category file and the messages that have been added to it.

- The Details view, which when you select a message category file, shows the properties of the message category, or of an individual message in the category, depending on what you have selected in the Properties Hierarchy.

There are tabs across the top of the editor area for each file that you have open, so that you can quickly switch between them.

Tasks view

Each time you save a change within a message set project, the content is validated to ensure that the message model follows certain rules. Any informational, warning or error messages relating to the validation appear in the task list.

Properties Hierarchy functions

Right-click in the Properties Hierarchy to display the following menu items:

- **Add Messages** (available at the **Message Category** level only) adds a new message to a message category file.
- **Undo** and **Redo** allow you to undo and redo changes to message category files. These two menu items apply to adding new messages, and also to changes that you make in the Details view. To undo a change that you have made to a message category file but not yet saved, click **Undo**. If you undo a change, you can reinstate it by clicking **Redo**.
- **Delete** (available at the individual **Message** level only) deletes the selected message from the message category file.

Details view functions

In the Details view, you use the **Message Category Kind** field to specify whether the message category will be used to generate WSDL (Web Services Description Language) files.

If the **Message Category Kind** is WSDL, you must provide definitions in the **Role Name** and **Role Type** fields for each message that you add to the message category file. These two fields appear when you select a message in the Properties Hierarchy.

You create any required documentation for a message set in the **Documentation** field. Right-click in this field to display a menu of standard text editing menu items that apply specifically to this field and allow you to undo changes, cut, copy, paste and delete text, and select all the text in the field.

Message Definition editor

The Message Definition editor is the default editor provided by the Broker Application Development perspective for editing message definition (.mxsd) files.

The editor is launched in the editor area when you open an existing message definition file using the Resource Navigator or when you create a new message definition file using the New Message Definition File wizard.

You use the Message Definition editor to:

- Edit message definitions created by importing data structures using the XML Schema, DTD, C or COBOL importers. The message definition file that the import process creates is automatically populated with the imported content, which you can then edit as required.
- Populate empty message definition files with message model objects by creating the elements, attributes, groups, types and messages needed to represent your message formats. The message model that you create can consist of both logical and physical information, if appropriate physical formats exist in the message set.

Message Definition editor views

The Message Definition editor has the following views:

Resource Navigator view

The Resource Navigator shows a hierarchical view of all the resources that are currently in your workspace. By expanding the folder for a message set project, you can see the resources, including message definition file or files, that this message set project contains.

Outline view

You use the Outline view to select message set objects to display in the editor area. When you open a message definition file, the Outline view shows a hierarchical view of the messages, types, groups and elements and attributes that the selected message definition contains.

Editor view

You edit the properties of message set objects in the editor area, which displays the message definition information for the object that you have currently selected in the Outline view. There are tabs across the top of the editor area for each file that you have open, so that you can quickly switch between them.

In the editor area, you can make any of the following types of change:

- Edit the logical structure of a message.
- Create and edit the physical structure and properties of a message.
- Create message definitions.
- Create common constructs within a message set for use with other message definition files.

The Message Definition editor comprises an Overview editor and a Properties editor. You switch between them by clicking the two tabs in the bottom left of the editor area. You can change the order in which the tabs are displayed by changing the message set preferences.

Tasks view

Each time you save a change within a message set project, the content is validated to ensure that the message model follows certain rules. Any informational, warning or error messages relating to the validation appear in the task list. Double clicking a message displays the properties of the relevant object in the editor area and shows the location of this object by highlighting the appropriate levels in the Resource Navigator, and the Outline view.

Message Definition editor functions

For information on the functions that the Message Definition editor provides, refer to the following topics:




- “Message Definition editor: Outline view”
- “Message Definition editor: Overview editor” on page 57
- “Message Definition editor: Properties editor” on page 57


Message Definition editor: Outline view

When you open a message definition file, the hierarchy for the selected file appears in the Outline view. You use the Outline view to select objects so that you can view and edit their message definition details in the editor area.

Global actions:

The Outline view toolbar icons provide a number of global actions that apply to all levels of the displayed hierarchy. The icons in the toolbar and their actions are shown in the table below.

Icon	Label	Action
	Find Element	Finds the specified message definition construct.
	Sort global constructs	If selected, global constructs are sorted alphabetically. If not selected, global constructs are displayed in the order that you create them.
	Collapse All	Collapses all the levels that are currently displayed in the Outline view.

You can also select these global actions from the toolbar drop-down menu, which appears when you click the  icon.

Common editing and navigation actions:

The Outline view provides the following common actions, which are also available in the Overview editor. Right-click in the Outline view or Overview editor to display the menu items.

- **Copy** and **Paste** provide standard editing functions.
- **Undo** and **Redo** allow you to undo and then redo changes that you have made. To undo a change, click **Undo**. If you undo a change, you can reinstate it, if you want to, by clicking **Redo**.
- **Go To Declaration** takes you to the appropriate point in the hierarchy (global group from a group reference, global attribute from an attribute reference, global element from an element reference, global element from a message) either in the same or in a different message definition file.
- **Go to Type Definition** takes you to the appropriate point in the hierarchy.
- **Delete** deletes the selected message set object.

Options for adding message model objects:

The Outline view provides options for adding the message model objects listed below. These options are also available in the Overview editor.

- Messages
- Complex types
- Simple types

- Attribute groups
- Groups
- Elements
- Attributes

Right click the appropriate level of the displayed hierarchy to display a pop-up menu with the menu items available for the selected level.

Message Definition editor: Overview editor

The Overview editor displays the contents of the message set object that you select in the Outline view so that you can view and change this object's most common properties. The Overview editor appears when you click the Overview tab in the editor area. The navigation buttons and links on the toolbar allow you to move back and forth between different objects. In the Overview editor you can make the types of change listed below:

- Edit the displayed values of any of the following:
 - Name
 - Type
 - Min Occurs
 - Max Occurs

To edit any of these values, select then click the appropriate field in the Overview editor. Depending on the type of field, this either displays a drop-down list or enables the field for direct editing.

- Add message model objects as in the Outline view. To add a new message model object, right-click **Structure** then click the appropriate menu item from the displayed pop-up menu. For example, the menu for **Messages** provides **Add Message** and **Add Message from Global Element**. When you add a new message model object, this appears in both the Outline view and the Overview editor.

Message Definition editor: Properties editor

The Properties editor allows you to change the properties for the selected message set object. The information that appears depends on what is currently selected in the Outline view and Overview editor. The Properties editor appears when you click the Properties tab in the editor area. The navigation buttons and links on the toolbar allow you to move back and forth between different objects. The Properties editor comprises the Properties Hierarchy and Details view.

Properties Hierarchy functions:

When you select a object in the Outline view, the Properties Hierarchy shows the following hierarchy for the message definition file that is currently open in the Message Definition editor:

- Logical properties
- Physical properties
- Documentation

Right-click in the Properties Hierarchy to display a number of menu items. The displayed menu items depend on which level of the hierarchy you select:

- **Undo** and **Redo** are available for all levels in the Properties Hierarchy and apply to changes that you make in the Details view for the currently selected object. To

undo a change that you have made to a file but not yet saved, click **Undo**. If you undo a change, you can reinstate it by clicking **Redo**.

- **Add Include** and **Add Import** are available when the top level of the message definition (.mxd) file's hierarchy is displayed in the **Outline** view. They allow you to link message definition files together.
- **Apply default physical format settings** is available for **Physical formats** only and applies the default wire format settings for the selected object.

If you add a new physical format in the message set (messageSet.mset) file, this new format is added under **Physical Properties** in the Properties Hierarchy. When you click **Physical Properties**, a link to the Message Set editor appears in the Details view.

Details view functions:

The Details view shows the properties for the object that is currently selected in the Properties Hierarchy. You can save any changes to the properties as you make them.

You create any required documentation for a message set in the **Documentation** field. This field appears when you select an object under **Documentation** in the Properties Hierarchy. Right-click in this field to display a menu of standard text editing menu items that apply specifically to this field and allow you to undo changes, cut, copy, paste and delete text, and select all the text in the field.

Message Flow editor

The Message Flow editor is the default editor provided by the Broker Application Development perspective for defining a graphical representation of a message flow in the workbench, and for setting properties for individual message flow nodes.

The editor is launched in the editor area of the workbench window when you open a message flow (.msgflow) file in the Resource Navigator view. The editor area is where you select built-in and user-defined nodes, and the connections between them, to define a message flow.

For information about changing the palette preferences in Message Brokers Toolkit Version 6.0, see Changing palette preferences.

Message Flow editor views

The Message Flow editor has the following views:

Resource Navigator view

The navigator view shows all the resources in your workspace, that is all message set resources and all message flow resources.

Editor view

The editor view shows the contents of the resource that is currently open. It also shows tabs for each of the resources that you have open so that you can quickly switch between them.

When the resource that is open is a message flow, at the bottom left of the editor view, there are two tabs, **Graph** and **User Defined Properties**.

When **Graph** is selected (this is the default), a graphical display of the message flow is shown in the editor view.

To the left of the editor view, there is a palette bar that contains all the available nodes that you can include in the message flow. By default, the palette bar is shown in collapsed mode when you open a message flow.

To open the palette, use one of the following methods:

- Hover the mouse over the palette bar while it is in collapsed mode. The palette bar expands. When you move the mouse away from the palette bar, it collapses again.
- Click the Show Palette icon at the top of the palette bar. The palette bar expands and it remains expanded when the mouse is moved away from the palette bar. To collapse the palette bar again, click the Hide Palette icon at the top of the palette bar while it is in expanded mode.

By default, the palette bar is docked to the left-side of the editor view. You can move it to the right-side of the editor view by clicking on the Palette bar while it is in collapsed mode, and dragging the bar to the right-side of the editor view.

When the **User Defined Properties** tab is selected, a User Defined Properties editor is opened that allows you to change the User Defined Properties of the message flow. The User Defined Properties editor comprises a User Property Hierarchy view and a Details view.

The User Property Hierarchy view provides three icons, Add Property Group, Add Property, and Delete that can be used to update the property hierarchy. When you add a property, a Details view is opened. In the Details view you can define the property Type and Default Value.

Outline view

The Outline view enables you to navigate to a particular node in a message flow and edit its properties.

Palette view

The Palette view lists the available nodes that you can select and include in the message flow. When the Palette view is not open the palette is available in the Message Flow editor as a pinnable and dockable element.

Overview view












The Overview tab provides a useful summary for large, complex message flows because it shows a small-scale version of the flow. Click the tab to show or hide the outline view.


Tasks view

The tasks view displays the warning and error messages that are generated by the editor's validation when you save the message flow file. If you double click an error, the editor indicates where it is located (for example, if you have not set a mandatory property in a node, it opens the properties dialog for that node on the right page).

Message Flow editor toolbar

The icons in the toolbar and their actions are shown in the table below.

Icon	Label	Action
	Manhattan	Displays all node connections as a series of horizontal and vertical lines
	Show grid	Displays a grid of horizontal and vertical dotted lines in the background of the editor area.
	Grid properties	Defines the horizontal and vertical spacing of the grid markers, and the gap between the borders of the editor area and the start of the grid markers.
	Align left	Lines up the left edge of the currently selected nodes. Enabled only when more than one node is selected.
	Align center	Lines up the horizontal center point (between left and right) of the currently selected nodes. Enabled only when more than one node is selected.
	Align right	Lines up the right edge of all the currently selected nodes. Enabled only when more than one node is selected.
	Align top	Lines up the top edge of the currently selected nodes. Enabled only when more than one node is selected.
	Align middle	Lines up the vertical center point (between top and bottom) of the currently selected nodes. Enabled only when more than one node is selected.
	Align bottom	Lines up the bottom edge of the currently selected nodes. Enabled only when more than one node is selected.
	Show distribute box	Displays a rectangular box around the currently selected nodes.
	Distribute horizontally	Aligns the currently selected nodes with the nearest right or left edge within the distribute box.

Icon	Label	Action
	Distribute vertically	Aligns the currently selected nodes with the nearest top or bottom edge within the distribute box.

Message Flow editor hover-feedback

In the Message flow editor you can display node and connection metadata by hovering the mouse over a node or subflow in a message flow. To view metadata information for a node, subflow or connection:

1. Open the Broker Application Development Perspective
2. Open a message flow
3. In the Editor view, hover the mouse over a node, a subflow, or a node connection in the open message flow by placing the mouse over the element.

A custom tooltip is displayed below the element.

To turn the pop-up into a focusable, scrollable window, press **F2**.

To hide the pop-up or window, choose one of the following methods:

- Press **Esc**
- Move the mouse away from the node.

Node and subflow feedback

When you hover the mouse over a node or subflow in the Editor view, the pop-up contains the following metadata information:

- The first line of text contains the the node-type and the display name of the node-instance. This line appears as **bold text**.
- The line below the node-type and display name contains the node's short description
- The final line contains the node's long description

If the node or subflow has no short or long description associated with it, only the first line of text is displayed.

If the task-list contains errors or warnings associated with the node or subflow, then that error or warning information is displayed instead of the metadata information.

Connection feedback

When you hover the mouse over a connection in the Editor view, hover-help is displayed containing the names of the output and input terminals that are connected.

Message Flow editor menus

All the actions that you can perform in the Editor view of the Message Flow editor are also available from a series of drop-down menus in WebSphere Message Broker. When the Message Flow editor is open, the following menus appear in the Workbench:

Menu	Keyboard shortcut
Flow	Alt + O
View	Alt + V
Palette	Alt + L
Run	Alt + R

The Navigate menu also contains actions specific to the Message Flow editor.

Flow menu

The flow menu contains all the actions that address editing of the flow model. Actions are statically positioned on the menu, and are enabled or disabled when nodes or connections are selected in the editor.

The following actions are available:

Menu item	Keyboard shortcut
Create Connection	C
Add subflow...	A
Rename...	N
Locate Subflow...	L
Promote Property	P
Properties...	R

View menu

The View menu contains all actions specific to the visual presentation of the flow model.

Menu item	Keyboard shortcut
Zoom In	I
Zoom Out	O
Manhattan Layout	M
Show Grid	G
Snap to Grid	G
Grid Properties...	P
Align	A
Distribute	D
Layout	L
Rotate	R

Palette menu

The Palette menu contains all actions specific to the Message Flow editor Palette.

Menu item	Keyboard shortcut
Add Node to Canvas	N
Customize	T
Settings...	S
Revert Palette to Defaults	T

Run menu

The Run menu contains the flow debug actions.

Menu item	Keyboard shortcut
Add Breakpoint	A
Remove Breakpoint	E
Add Breakpoints After Node	A
Add Breakpoints Before Node	B

Navigate menu

The Navigate menu contains the following actions specific to the Message Flow editor:

Menu item	Keyboard shortcut
Open Subflow	O
Open Source	S

Adding a node using the keyboard

Before you start

To complete this task, you must have completed the following task:

- Creating a message flow

You can use the Message Flow editor to perform tasks using the keyboard, such as adding a node to a message flow.

Complete the following steps to add a node to the canvas:

1. Open the message flow you want to add a node to by double-clicking the message flow in the Navigator view. You can also open the message flow by right-clicking it in the Navigator view and clicking **Open**. The message flow contents are displayed in the editor view.
2. Open the Palette view or the Palette bar.
3. Select a node in the Palette view or Palette bar using the up and down arrows to highlight the node you want to add to the canvas.
4. Add the Node to the canvas using one of the following methods:
 - Select **Palette** → **Add Node to Canvas** by pressing **Alt + L** and then pressing **N**.

- Press **Shift + F10** to open the context-sensitive menu for the Palette, and Press **N**.

The node that you selected in the Palette bar or Palette view is placed on the canvas in the Editor view.

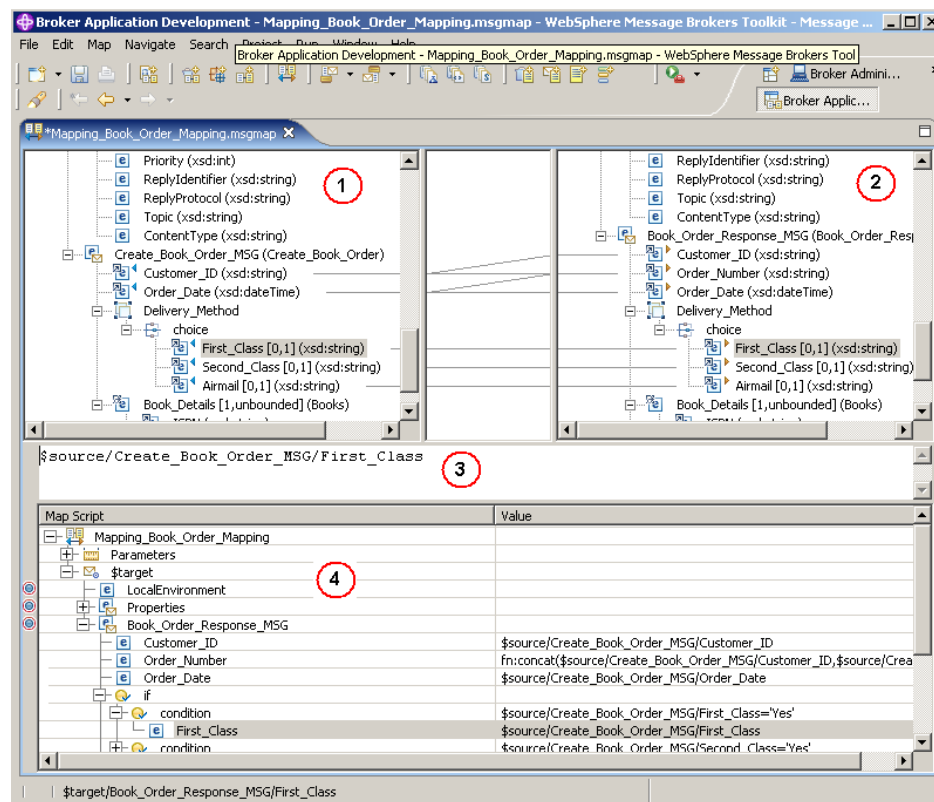
You can move the node that you have placed on the canvas using the keyboard controls described in “Message Brokers Toolkit keyboard shortcuts” on page 68

Message Mapping editor

You configure a message mapping using the Message Mapping editor, which you use to set values for:

- the message destination
- message headers
- message content

Here is an example of the Message Mapping editor. There are separate panes for working with sources, targets and expressions, as well as a spreadsheet view.



1. **Source pane:** displays a source message or database table
2. **Target pane:** displays a target message
3. **Edit pane:** displays the expression to be used to derive the target element value
4. **Spreadsheet pane:** displays a summary of the mappings in spreadsheet columns (each target field and its value)

Use the Message Mapping editor to perform various mapping tasks.

Wizards and dialog boxes are provided for tasks such as adding mappable elements, working with ESQL, and working with submaps. Mappings that are created with the Message Mapping editor are automatically validated and compiled, ready for adding to a broker archive (bar) file, and subsequent deployment to WebSphere Message Broker.

Message Node editor

The Message Node editor is the default editor provided by the Plug-in Development perspective for defining the content of user-defined nodes.

The editor is launched in the editor area of the perspective when a new message flow node file (.msgnode) is created.

Message Node editor views

The Message Node editor has two tabs:

Terminals

This tab is for adding, renaming, or deleting input and output terminals for the new node.

Properties

This tab is for adding properties and property groups to the node, and defining property type.

Message set editor

The Message set editor is the default editor provided by the Broker Application Development perspective for editing message set (messageSet.mset) files.

The editor is launched in the editor area when you open an existing message set file using the Resource Navigator or when you create a new message set file using the New Message Set File wizard.

Message set editor views

The Message set editor has the following views:

Resource Navigator view

The Resource Navigator shows a hierarchical view of all the resources that are currently in your workspace. By expanding the folder for a message set project, you can see the resources, including the message set file, that this message set project contains.

Editor view

You edit the properties of message set in the editor area of the Message set editor, which has a single tab, the **Properties** tab. This tab provides:

- The Properties Hierarchy, which provides a hierarchy at three levels: Message set, Physical properties and Documentation.
- The Details view, which displays the properties relating to the currently selected level in the Properties Hierarchy.

There are tabs across the top of the editor area for each file that you have open, so that you can quickly switch between them.

Tasks view

Each time you save a change within a message set project, the content is

validated to ensure that the message model follows certain rules. Any informational, warning or error messages relating to the validation appear in the task list.

Properties Hierarchy functions

Right-click in the Properties Hierarchy to display the menu items available for the level of the hierarchy that you have currently selected:

- **Undo** and **Redo** are available for all levels in the Properties Hierarchy and apply to changes that you make in the Details view. To undo a change that you have made to a message set file but not yet saved, click **Undo**. If you undo a change, you can reinstate it by clicking **Redo**.
- For the **Physical formats** level, the **Add Custom Wire Format**, **Add XML Wire Format**, and **Add Tagged/Delimited String Format** menu items are available for adding physical format layers to a message set file.
- Under **Physical formats**, the following menu items are available for the selected physical format layer:
 - **Apply default physical format settings** applies the default physical format settings to the selected physical format layer.
 - **Rename** renames the selected physical format layer for the message set file. Click **Finish** on the displayed window to confirm the change of name.
 - **Delete** deletes the selected physical format layer from the message set file. Click **Finish** on the displayed window to confirm the deletion.

Details view functions

You use the Details view to define and edit global properties for the message set, and create and edit properties for the physical format layers that you have added to the message set.

You create any required documentation for a message set in the **Documentation** field. Right-click in this field to display a menu of standard text editing menu items that apply specifically to this field and allow you to undo changes, cut, copy, paste and delete text, and select all the text in the field.

Subscriptions Query editor

The Subscriptions Query editor is the default editor that is provided by the Broker Administration perspective for managing subscriptions. Use this editor to view and delete subscriptions in your publish/subscribe system. The editor allows you to set query parameters, submit subscription queries, and delete subscriptions.

You can use wildcard characters in subscription names so that you can manage a specified range of subscriptions.

The Subscriptions Query editor is launched in the editor area of the Broker Administration perspective when you double-click Subscriptions in the Domains view.

If you are not connected to a domain, an attempt to do so is made before opening the editor. During this attempt, if you click **Cancel**, the connection that is in progress stops and the domain returns to its initial unconnected state.

Topics Hierarchy editor

The Topics Hierarchy editor is the default editor that is provided by the Broker Administration perspective for managing the publish/subscribe topic hierarchy. Use this editor to create, modify, and delete topics, or to define or modify your topic Access Control Lists (ACLs).

The Topics Hierarchy editor is launched in the editor area of the Broker Administration perspective when you double-click Topics in the Domains view.

If you are not connected to a domain, an attempt to do so is made before opening the editor. During this attempt, if you click **Cancel**, the connection that is in progress stops and the domain returns to its initial unconnected state.

In the editor view, use the **Topic/Users** and **User/Topics** buttons to determine whether you see a list of topics or a list of users in the left-hand part of the editor area.

If you click **Topic/Users**, the left-hand part of the editor area contains a list of the topics. Click on a topic to see in the right-hand part of the editor area the ACL for the topic that you selected.

If you click **User/Topics**, the left-hand part of the editor area contains a list of the groups and users. Click on a group or user to see in the right-hand part of the editor area the ACL for the topics that are available to the group or user that you selected.

When the editor is launched, an Outline view is also shown that lists the topics and principals (groups and users).

Either the outline view or the editor view can be used to select what is shown in the right-hand part of the editor area. The behavior varies depending on whether **Topic/Users** or **User/Topics** is selected.

If **Topic/Users** is selected, the following happens:

- If you select a topic in the Outline view, the corresponding topic is selected in the Topics tree.
- If you select a user or group in the Outline view, the corresponding group or user is selected in the Topic ACL table.
- If you select a topic in the Topics tree, the corresponding topic is selected in the Outline view.
- If you select a group or user in the Topic ACL table, the corresponding group or user is selected in the Outline view.

If **User/Topics** is selected, the following happens:

- If you select a topic in the Outline view, the corresponding topic is selected in the User ACL table.
- If you select a user or group in the Outline view, the corresponding group or user is selected in the Users tree.
- If you select a user or group in the Users tree, the corresponding user or group is selected in the Outline view.
- If you select a topic in the User ACL view, the corresponding topic is selected in the Outline view.

The Outline view reacts to changes made in the editor area; for example when a topic is created or removed.

Message Brokers Toolkit keyboard shortcuts

You can navigate the user interface using the keyboard. The following table shows the general shortcut keys provided in the workbench:

Use...	To....
Left, right, up, and down arrows	Move in the direction shown on the key.
Tab	Move the focus forward through the workbench.
Shift+Tab	Move the focus backward through the workbench.
Esc	Cancel the action.
Enter	Commit the action.
Spacebar	Select the object which currently has the focus.
Shift+F10	Open a pop-up menu.

Message Flow editor keyboard shortcuts

The following table shows keyboard shortcuts that are specific to the Message Flow editor:

Use...	When the Selection tool is active	When the Connection tool is active
Left, right, up, and down arrows	Select a node on the palette or on the canvas. Use with Shift, or Ctrl+Spacebar, to select multiple nodes.	Select a node's terminal.
/ and \	Cycle forward (press /) or backward (press \) through the selected node's connections. Use with Shift, or Ctrl+Spacebar, to select multiple connections.	
. (decimal key) and Shift + . (decimal key)	Cycle forward (press the decimal key) or backward (press Shift and the decimal key) through the move handles or the resize handles of the currently selected object on the canvas. When a node is selected, use this to activate the move handles, and then use the arrow keys to move the node. Press Enter to confirm the move. When a connection is selected, use this to activate its resize handles, and then use the arrow keys to create bend points in the connection. Press Enter to confirm the move.	
Esc		In general, pressing Esc cancels the previous action. When there is no previous action to cancel, and the Connection tool is selected, pressing Esc activates the Selection tool.

Here are some examples of how to use the Message Flow editor keyboard shortcuts:

Adding nodes from the palette to the canvas:

1. Give focus to the palette, and use the up and down arrows to highlight a node.
2. Press Alt+l (Alt+lowercase 'L') to open the Palette menu, and select **Add Node to Canvas**.

Moving nodes on the canvas:

1. Select a node on the canvas.
2. Press . (decimal key) to select one of the move handles around the node.
3. Use the left, right, up, and down arrows to move the node to a new location, and press Enter to confirm the move.

Connecting nodes:

1. Give focus to the palette, and use the arrow keys to highlight the Connection tool, and then press spacebar to activate the Connection tool.
2. Give focus to the canvas. Use the arrow keys to select a source (output) terminal, and press Enter to confirm your selection.
3. Use the arrow keys to select a target (input) terminal, and press Enter to confirm your selection.
4. Optional: Press Esc to deactivate the Connection tool and activate the Selection tool.

Selecting a node connection:

1. Give focus to the canvas, and use the arrow keys to select a node that has a connection you want to select.
2. Use the / and \ keys to cycle through the connections.

Creating a bend point:

1. Select a connection, and press . (decimal key) until the midpoint of the connection is selected.
2. Use the arrow keys to move the midpoint to a new location, and press Enter to confirm the move.

Reconnecting an existing connection to a new terminal:

1. Select a connection, and press . (decimal key) until the end of the connection that you want to modify is selected.
2. Use the arrow keys to select a new source (input) terminal, and Press Enter to confirm your selection.

Glossary of terms and abbreviations

This glossary defines WebSphere Message Broker terms and abbreviations used in this online information.

There is also a migration glossary that lists differences in terminology between WebSphere Message Broker Version 6.0 and previous versions of the product.

A

access control list (ACL)

In computer security, a list associated with an object that identifies all the subjects that can access the object and their access rights. Subjects are principals that have explicit permissions (to publish, to subscribe to, and to request persistent delivery of, a publication message) against a topic in the topic tree. The ACLs define the implementation of topic-based security.

ACL See access control list.

aggregation

See message element aggregation.

AMI See Application Messaging Interface.

Application Messaging Interface (AMI)

The programming interface provided by WebSphere MQ that defines a high level interface to message queuing services. See also Message Queue Interface (MQI) and Java Message Service (JMS). Applications that use the AMI connect to the broker using WebSphere MQ Enterprise Transport.

attribute

In XML, a name-value pair within a tagged element that modifies certain features of the element. In other message domains, a simple element within a message. An attribute requires special treatment when used with XML messages. In messages that are not XML, an attribute is treated exactly like a simple element based on the same simple type.

attribute group

A set of attributes that can appear in a complex type.

B

bar file

See broker archive file.

bend point

A point that is introduced in a connection between two message flow nodes at which the line that represents the connection changes direction. A bend point can be used to make node alignment and processing logic clearer and more effectively displayed.

binary large object (BLOB)

A block of bytes of data (for example, the body of a message) that has no discernible meaning, but is treated as one solid entity that cannot be interpreted.

BLOB See binary large object.

BLOB domain

The message domain that includes all messages that have content that cannot be interpreted and subdivided into smaller sections of information. Messages in this domain are processed by the BLOB parser. See also IDoc domain, JMS domain, MRM domain, and XML domain.

BLOB parser

A program that interprets a bit stream or message tree that represents a message that belongs to the BLOB domain, and generates the corresponding tree from the bit stream on input, or a bit stream from the tree on output.

broker

A set of execution processes that host one or more message flows. Also known as message broker.

broker archive file

The unit of deployment to the broker; also known as a bar file. It contains any number of compiled message flows (.cmf), message sets (.dictionary), and a single deployment descriptor. It can also contain any additional files you might need as long as the extension does not overlap the .cmf and .dictionary extensions.

broker domain

A collection of brokers that share a common configuration, together with the Configuration Manager that controls them.

broker schema

A symbol space that defines the scope of uniqueness of the names of resources defined within it. The resources are message flows, ESQL files, and mapping files.

built-in node

A message flow node that is supplied by the product. A number of supplied nodes provide basic processing such as input and output.

C**callback function**

See implementation function.

cardinality

See mapping cardinality.

category

An optional grouping of messages that are related in some way. For example, messages that relate to a particular application are included in a single category.

cmf See compiled message flow.

collective

A set of brokers that are fully interconnected and form part of a multi-broker network for publish/subscribe applications.

compiled message flow (cmf)

A message flow that has been compiled to prepare it for transmission to the broker. A cmf is sent to the broker within a bar file.

complex element

A named structure that contains simple elements within the message.

Complex elements can contain other complex elements, and can also contain groups. The content of a complex element is defined by a complex type. See also simple element.

complex type

A structure within a message. A complex type contains elements, attributes, and groups organized into a hierarchy. See also simple type.

component

A set of runtime processes that perform a specific set of functions. A component is a broker, a Configuration Manager, a Database Instance Manager, or a User Name Server.

component directory

In z/OS, the root directory of the component's runtime environment.

component name

The external name of a component. It is used, for example, in the workbench and in commands. Each component requires a name.

component PDSE

In a z/OS environment, a PDSE that contains jobs to define resources to DB2, WebSphere MQ, and the WebSphere Message Broker started task. See partitioned data set

configuration

In a broker domain, the brokers, execution groups, deployed message sets, and deployed message flows, and the defined topics and access control lists.

Configuration Manager

The component that provides an interface between the workbench and a set of runtime brokers. It provides brokers with their initial configuration, and updates them with any subsequent changes. It maintains the broker domain configuration.

Configuration Manager Proxy

An application programming interface that your applications can use to control broker domains through a remote interface to the Configuration Manager.

connection

See message flow node connection.

content-based filter

In publish/subscribe, an expression that is included as part of a subscription to determine whether a publication message is received based on its content. The expression can include wild cards.

Custom Wire Format

The physical representation of a message in the MRM domain that is composed of a number of fixed format data structures or elements, which are not separated by delimiter characters.

CWF See Custom Wire Format.

D

Database Instance Manager

On Windows, a network server that supports the creation, maintenance, and deletion of databases used by brokers in all installations on a single

| system. Database support is limited to Derby and DB2. The Database
| Instance Manager is associated with a Windows service.

data element separation

A delimiter sequence that defines how a TDS message is to be parsed. The following separation types are supported: data pattern separation, delimited separation, fixed length separation, and tagged separation

datagram

A form of asynchronous messaging in which an application sends a message, but does not want a response. Also known as send-and-forget. See also request/reply.

debugger

See flow debugger.

deploy

The process of transferring data to an execution group on a broker so that it can take effect in the broker domain. For deploying message flows and associated resources, the data is packaged in a broker archive (bar) file before being sent to the Configuration Manager, from where it is unpackaged and distributed appropriately.

Derby The database delivered by IBM Cloudscape Version 10.0, which is built on the Derby database from the Apache Software Foundation. Cloudscape does not modify Derby in any way, but provides additional function including installers. Derby database support is embedded in the broker component on Windows only.

destination list

See local environment.

distribution list

A list of WebSphere MQ queues to which a message can be put with a single statement.

document type definition (DTD)

The rules that specify the structure for a particular class of SGML or XML documents. The DTD defines the structure with elements, attributes, and notations, and it establishes constraints for how each element, attribute, and notation can be used within the particular class of documents. A DTD is analogous to a database schema in that the DTD completely describes the structure for a particular markup language.

DTD See document type definition.

E

editor area

The area in the workbench window where files are opened for editing.

element

A named piece of information, or a field, within a message, with a business meaning agreed by the applications that create and process the message. See also simple element and complex element.

embedded message

See multipart message.

environment

A structure within the message tree that is user-defined and can contain variable information that is associated with a message while it is being processed by a message flow.

ESM See external security manager.

ESQL See extended SQL.

ESQL data type

A characteristic of an item of data that determines how that data is processed. ESQL supports six data types (Boolean, datetime, null, numeric, reference, and string). Data that is retrieved from a database or is defined in a message model is mapped to one of these basic ESQL types when it is processed in ESQL expressions.

ESQL field reference

A sequence of period-separated values that identify a specific field (which might be a structure) within a message tree or a database table. The path from the root of the information to the specific field is traced using the parent/child relationships. An example of a field reference might be something like `Body.Invoice.InvoiceNo`.

ESQL function

A single ESQL expression that calculates a resultant value from a number of given input values. The function can take input parameters but has no output parameters; it returns the value that results from the implementation of the expression to the caller. The ESQL expression can be a compound expression such as `BEGIN END`.

ESQL module

A sequence of declarations that define MODULE-scope variables and their initialization, and a sequence of subroutine (function and procedure) declarations that define a specific behavior for a message flow node. A module must begin with the `CREATE node_type MODULE` statement and end with an `END MODULE` statement. The `node_type` must be one of `COMPUTE`, `DATABASE`, or `FILTER`. The entry point of the ESQL code is the MODULE scope procedure named `MAIN`.

ESQL procedure

A subroutine that has no return value. It can accept input parameters from and return output parameters to the caller.

ESQL variable

A local temporary field that is used to assist in the processing of a message.

exception list

A list of exceptions that has been generated during the processing of a message, with supporting information.

execution group

A named process or set of processes within a broker in which message flows are executed. The broker is guaranteed to enforce some degree of isolation between message flows in distinct execution groups because it ensures that they execute in separate address spaces, or as unique processes.

extended SQL (ESQL)

A specialized set of SQL functions and statements based on regular SQL, extended with functions and statements unique to WebSphere Message Broker.

extensible markup language (XML)

A standard metalanguage for defining markup languages that was derived from and is a subset of Standard Generalized Markup Language (SGML).

extensible stylesheet language (XSL)

A language for specifying style sheets for XML documents. XSL Transformation (XSLT) is used with XSL to describe how an XML document is transformed into another document.

external security manager (ESM)

In a z/OS environment, a security product that performs security checking on users and resources. RACF is an example of an ESM.

F**field reference**

See ESQL field reference

filter (1) An ESQL expression that is applied to the content of a message in a Filter node to determine how the message is processed.

(2) An ESQL expression that is applied to the content of a publication message to determine if the message matches certain criteria.

flow debugger

A facility to debug message flows that is provided in the Debug perspective in the workbench.

format

The definition of the internal structure of a message, in terms of the fields and the order of those fields. A format can be self-defining, in which case the message is interpreted dynamically when it is read.

G**graphical user interface (GUI)**

A type of computer interface that presents a visual metaphor of a real-world scene, often of a desktop, by combining high-resolution graphics, pointing devices, menu bars and other menus, overlapping windows, icons, and the object-action relationship.

group A list of elements with information about how those elements can appear in a message. Groups can be ordered, unordered, or selective.

GUI See graphical user interface.

I**IBM Runtime Environment for Java**

A subset of the IBM Developer Kit for the Java Platform that contains the core executable files and other files that constitute the standard Java platform. The IBM Runtime Environment includes the Java virtual machine (JVM), core classes, and supporting files.

IBM Software Developer Kit for Java

A software package that can be used to write, compile, debug, and run Java applets and applications.

IDoc domain

The message domain that includes all messages that are exchanged between the broker and SAP R3 clients across the MQSeries link for R/3. Messages in this domain are processed by the IDoc parser. See also BLOB domain, JMS domain, MRM domain, and XML domain.

IDoc parser

A program that interprets a bit stream or tree that represents a message that belongs to the IDoc domain, and generates the corresponding tree from the bit stream on input, or bit stream from the tree on output.

implementation function

A function written for a user-defined node or message parser. Also known as a callback function.

input node

A message flow node that represents a source of messages for a message flow or subflow. See also output node.

installation directory

In a z/OS environment, a file system into which all product data is installed, and from which it is referenced and retrieved during the customization phase.

J**Java Database Connectivity (JDBC)**

An industry standard for database-independent connectivity between the Java platform and a wide range of databases. The JDBC interface provides a call-level API for SQL-based database access. See also Open Database Connectivity.

Java Message Service (JMS)

An application programming interface that provides Java language functions for handling messages. See also Application Messaging Interface (AMI) and Message Queue Interface (MQI). Applications using JMS connect to the broker using either WebSphere MQ Real-time Transport or WebSphere MQ Multicast Transport.

JCL See Job Control Language

JDBC See Java Database Connectivity.

JMS See Java Message Service.

JMS domain

The message domain that includes all messages that are produced by the WebSphere MQ implementation of the Java Message Service standard. These messages, which have a message type of either JMSMap or JMSStream, are supported in the same way as messages in the XML domain and are parsed by the XML parser. See also BLOB domain, IDoc domain, MRM domain, and XML domain.

Job Control Language

Job Control Language (JCL) comprises a set of Job Control Statements, which are used to define work requests called jobs. JCL tells the operating system what program to execute and defines its inputs and outputs.

L

LIL See loadable implementation library.

loadable implementation library (LIL)

The implementation module for a node or parser written in C. This is implemented in the same way as a dynamic link library, but has a file extension of .lil rather than .dll.

local environment

A structure within the message tree that contains broker and, optionally, user information associated with a message while it is being processed by a message flow. In previous releases, the local environment structure was known as the Destination list; the latter term is retained for compatibility.

local error log

A generic term that refers to the logs to which WebSphere Message Broker writes records on the local system. Also known as system log.

M

map (1) A complete transformation that has sources and targets that define the structure of the inputs and the outputs, respectively. A map is represented as a msgmap file.

(2) To associate a source to a target by creating a target value expression.

mapping

A target value expression.

mapping cardinality

The granularity of the way in which message elements are mapped from message source to message target. For example:

- One-to-one: associates a single source with a single target
- One-sided: associates a value with a target
- Many-to-one: associates multiple sources with a single target

message

A communication sent from a person or program to another person or program. In WebSphere Message Broker, a message can be modeled by a message definition which describes the structure and content of the message. Messages must have a structure and format which is agreed by the sending and receiving applications.

message broker

See broker.

Message Brokers Toolkit

The WebSphere Message Broker development environment that integrates with IBM Rational Application Developer which is based on the IBM WebSphere Eclipse Platform. Also known as the workbench.

message definition

A logical description of a message. A message definition is a structured collection of simple elements.

message definition file

A file that contains the messages, elements, types, and groups that make up a message set.

message dictionary

A data structure that describes all of the messages in a message set in a form suitable for deployment to a broker.

message domain

A grouping of messages that share certain characteristics. A message

domain has an associated parser that interprets messages that are received and generated by a broker. WebSphere Message Broker supports messages in the BLOB domain, IDoc domain, JMS domain, MIME domain, MRM domain, and XML domain. You can create additional parsers known as user-defined parsers to support messages that do not conform to the supported domains.

message element aggregation

A message element aggregation occurs when all the repeatable elements in one instance are mapped to another instance; it is not possible to map the repeatable elements themselves, only the instances. This aggregation is useful when mapping all possible inputs to one or more outputs, and can be used for copying an array, or for assigning a scalar, such as a summation. Use message element aggregation when the following conditions are met:

- a single source and target are selected
- source and target are of simple numeric type
- the source repeats

message flow

A sequence of processing steps that execute in the broker when an input message is received. A message flow is created in the workbench by including a number of message flow nodes that each represents a set of actions that define a processing step. The connections in the flow determine which processing steps are carried out, in which order, and under which conditions. A message flow must include an input node that provides the source of the messages that are processed. Message flows are then ready to deploy to a broker for execution. See also subflow.

message flow node

A processing step in a message flow. A message flow node can be either a built-in node, a user-defined node, or a subflow node. Also called message processing node.

message flow node connection

An entity that connects the output terminal of one message flow node to the input terminal of another. A message flow node connection represents the flow of control and data between two message flow nodes.

message model

A definition of a message format that is used by applications. You define a message model in the workbench.

message parser

A program that interprets the bit stream of an incoming message and creates an internal representation of the message in a tree structure, and that regenerates a bit stream for an outgoing message from the internal representation.

message processing node

See message flow node.

Message Queue Interface (MQI)

The programming interface provided by WebSphere MQ queue managers. The programming interface allows application programs to access message queuing services. See also Application Messaging Interface (AMI) and Java Message Service (JMS). Applications using the MQI connect to the broker using WebSphere MQ Enterprise Transport.

message set

A container; a logical grouping of messages and associated message resources (elements, types, groups).

message set documentation

A human-readable form of message definitions that you have created in the workbench.

message set project

A specialized container in which you create and maintain all the resources associated with one message set.

message template

A named and managed entity that represents the format of a particular message. Message templates represent a business asset of an organization.

message tree

The logical tree structure that represents the content and structure of a message in the broker. The message tree is created by a message parser from the input message received by a message flow.

message type

The logical structure of the data within a message; for example, the number and location of character strings.

metadata

The data that describes the characteristic of stored data.

MIME See Multipurpose Internet Mail Extensions.

MIME domain

The message domain that includes all messages that conform to the MIME standard.

MIME parser

A program that interprets a bit stream or tree that represents a message that belongs to the MIME domain, and generates the corresponding tree from the bit stream on input, or bit stream from the tree on output.

MQI See Message Queue Interface.

MQIsdp

See SCADA device protocol.

MQRFH

An architected message header that is used to provide metadata for the processing of a message. This header is supported by the MQSeries Publish/Subscribe SupportPac.

MQRFH2

An extended version of MQRFH, providing enhanced function in message processing.

MRM The name given to the domain and parser associated with messages that are modeled in the workbench. MRM stands for Message Repository Manager and is used only to identify the MRM parser and MRM domain.

MRM domain

The message domain that includes all messages that are modeled in the workbench. Message models can be created to represent a wide range of message types, with one or more optional physical formats. Messages in this domain are processed by the MRM parser. See also BLOB domain, IDoc domain, JMS domain and XML domain.

MRM parser

A program that interprets a bit stream or tree that represents a message that belongs to the MRM domain, and generates the corresponding tree from the bit stream on input, or bit stream from the tree on output. Its interpretation depends on the physical format that you have associated with the input or output message.

multilevel wild card

A wild card that can be specified in subscriptions to match any number of levels in a topic.

multipart message

A message that contains one or more other messages within its structure. The contained message is sometimes referred to as an embedded message.

Multipurpose Internet Mail Extensions

An Internet standard that allows different forms of data including video, audio, or binary data to be attached to e-mail without requiring translation into ASCII text.

N**namespace**

In XML, a uniform resource identifier (URI) that provides a unique name to associate with all the elements and type definitions in a schema. XML instance documents and XML schemas can make use of namespaces.

node (1) An endpoint or junction used in a message flow. See message flow node.

(2) An element in a message mapping tree. See tree node.

O**ODBC**

See Open Database Connectivity.

Open Database Connectivity (ODBC)

A standard application programming interface (API) for accessing data in both relational and non-relational database management systems. Using this API, database applications can access data stored in database management systems on a variety of computers even if each database management system uses a different data storage format and programming interface. ODBC is based on the call level interface (CLI) specification of the X/Open SQL Access Group.

output node

A message flow node that represents a point at which messages leave the message flow or subflow. See also input node.

P

parser See message parser.

partitioned data set (PDS, PDSE)

In a z/OS environment, a data set in direct-access storage that is divided into partitions, which are called members. A partitioned data set (extended) (PDSE) is an extension to a PDS that contains an indexed directory in addition to the members.

PDS, PDSE

See partitioned data set.

perspective

A group of views that show various aspects of the resources in the workbench. The user can switch perspectives, depending on the task at hand, and customize the layout of views and editors within the perspective. See also view.

physical format

The physical representation of a message within the bit stream. The supported physical formats are Custom Wire Format, XML Wire Format, and Tagged/Delimited String Format.

point-to-point

A style of messaging application in which the sending application knows the destination of the message. Contrast with publish/subscribe.

predefined element and message

An element or message for which a matching definition exists in the message model with an appropriate set of properties and in the correct context. See also self-defining element and message.

principal

An individual user ID (for example, a login ID) or a group. A group can contain individual user IDs and other groups, to the level of nesting supported by the underlying facility.

property

A characteristic that, as one of a set of characteristics, defines the values and behaviors of objects in the workbench. For example, message flow nodes and deployed message flows have properties.

publication

A piece of information about a specified topic that is available to a broker in a publish/subscribe system.

publication node

An end point of a specific path through a message flow to which a client application subscribes, identified to the client by its subscription point.

publisher

An application that makes information about a specified topic available to a broker in a publish/subscribe system.

publish/subscribe

A style of messaging application in which the providers of information (publishers) are de-coupled from the consumers of that information (subscribers) using a broker. See also topic. Contrast with point-to-point messaging.

publish/subscribe topology

The brokers, the collectives, and the connections between them, that support publish/subscribe applications in the broker domain.

Q

queue A WebSphere MQ object to which message queuing applications can put messages, and from which they can get messages.

queue manager

A system program that provides queuing services to applications. It provides an application programming interface (the MQI) to enable programs to access messages on the queues that the queue manager owns.

R

request/reply

A type of messaging application in which a request message is used to request a reply from another application. See also datagram.

resource

A file of any type that exists in the workbench. You can view and edit a resource in the Resource Navigator view in the workbench.

Resource Recovery Services (RRS)

A z/OS facility that provides two-phase sync point support across participating resource managers.

retained publication

A published message that is kept at the broker for propagation to clients that subscribe at some point in the future.

RRS See Resource Recovery Services.

S

SCADA

See Supervisory, Control, And Data Acquisition.

SCADA device protocol (MQIsdp)

A protocol that implements the WebSphere MQ Telemetry Transport to connect SCADA devices to the broker.

schema

See XML Schema.

self-defining element and message

An element or message for which no matching definition exists in the message model. For example, a message coded in XML is self-defining. See also predefined element and message.

send-and-forget

See datagram.

simple element

A field in a message that is based on a simple type. A simple element can repeat, and it can define a default or a fixed value. See also complex element.

simple type

A characteristic of a simple element that defines the type of data within a message (for example, string, integer, or float). A simple type can have value constraints which place limits on the values of any simple elements based on that simple type. See also complex type.

single-level wild card

A wild card that can be specified in subscriptions to match a single level in a topic.

SQL See Structured Query Language.

stream

A method of topic partitioning that is used by applications that connect to MQSeries Publish/Subscribe SupportPac brokers.

Structured Query Language (SQL)

A standardized programming language that is used to define and

manipulate data in a relational database. ESQL, the language used by WebSphere Message Broker, is based on SQL, and has many similar constructs.

style sheet

A specification of formatting instructions that, when applied to structured information, provides a particular rendering of that information (for example, online or printed). Different style sheets can be applied to the same piece of structured information to produce different presentations of the information.

subflow

A sequence of processing steps, implemented by message flow nodes, that is designed to be embedded in a message flow or in another subflow. A subflow must include at least one Input or Output node. A subflow can be executed by a broker only as part of the message flow in which it is embedded, and therefore cannot be deployed.

subflow node

A message flow node that represents a subflow.

subscriber

An application that requests information about a specified topic from a publish/subscribe broker.

subscription

A record that contains the information that a subscriber passes to its local broker to describe the publications that it wants to receive.

subscription filter

A predicate that specifies a subset of messages that are to be delivered to a particular subscriber.

subscription point

The name that a subscriber uses to request publications from a particular set of publication nodes. It is the property of a publication node that differentiates that publication node from other publication nodes in the same message flow.

substitution group

An XML Schema feature that provides a means of substituting one element for another in an XML message. A substitution group contains a list of global elements that can appear in place of another global element, called the head element.

Supervisory, Control, And Data Acquisition (SCADA)

A term used to describe any form of remote telemetry system that is used to gather data from remote sensor devices (for example, flow rate meters on an oil pipeline) and for the near real time control of remote equipment (for example, pipeline valves). These devices communicate with the broker using the SCADA device protocol (MQIsdp).

system log

See local error log.

T**Tagged/Delimited String (TDS) Format**

The physical representation of a message in the MRM domain that has a number of data elements separated by tags and delimiters.

TDS Format

See Tagged/Delimited String Format.

terminal

The point at which one node in a message flow is connected to another node. You can connect terminals to control the route that a message takes, dependent on the outcome of the operation performed by the node on that message.

topic A character string that describes the nature of the data that is published in a publish/subscribe system.

topic based subscription

A subscription specified by a subscribing application that includes a topic for filtering of publications.

topic security

The application of ACLs to one or more topics to control subscriber access to published messages.

topology

See publish/subscribe topology.

transform

A defined way in which a message of one format is converted into one or more messages of another format.

tree node

An element in a mapping tree. A container for the mapping type such as an MRM message, RDB table, a column, or a basic element.

type A characteristic of an element that describes its data content. See also simple type and complex type.

U**Unicode Transformation Format, 8-bit encoding form (UTF-8)**

A transformation format that is designed for ease of use with existing ASCII-based systems. UTF-8 is an encoding of Unicode character strings that optimizes the encoding of ASCII characters in support of text-based communication.

uniform resource identifier (URI)

An encoded address that represents any resource, such as an HTML document, image, video clip, or program, on the Web. As opposed to a Uniform resource locator or a Uniform resource name, which are concrete entities, a URI is an abstract superclass.

uniform resource locator (URL)

A sequence of characters that represent information resources on a computer or in a network such as the Internet. This sequence of characters includes (a) the abbreviated name of the protocol used to access the information resource and (b) the information used by the protocol to locate the information resource. A Web server typically maps the request portion of the URL to a path and file name. Also known as universal resource locator.

uniform resource name (URN)

A name that uniquely identifies a Web service to a client.

URI See Uniform resource identifier.

URL See Uniform resource locator.

URN See Uniform resource name.

user-defined node

An extension to the broker that provides a new message flow node in addition to those supplied with the product. See also implementation function and utility function.

user-defined parser

An extension to the broker that provides a new message parser in addition to those supplied with the product. See also implementation function and utility function.

User Name Server

A component that interfaces with operating system facilities to determine valid users and groups.

UTF-8 See Unicode Transformation Format.

utility function

A function provided by the broker that can be used by developers who write user-defined nodes or parsers.

V

value constraint

A limit that sets a restriction on the values that a simple type can represent.

view A display area in the workbench in which you can navigate and edit your information and resources. For example, the Resource Navigator view enables you to view and edit your project files. See also perspective.

W

warehouse

A persistent, historical data store for events (or messages). The Warehouse node within a message flow supports the recording of information in a database for subsequent retrieval and processing by other applications.

Web service

A modular application that performs specific tasks and is accessible through open protocols like HTTP and SOAP.

Web Services Description Language (WSDL)

An XML-based specification for describing networked services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. A WSDL document enables a Web services client to invoke a Web service using the messages defined in a message definition.

WebSphere MQ Enterprise Transport

A transport protocol supported by WebSphere Message Broker that enables WebSphere MQ application clients to connect to brokers.

WebSphere MQ Everyplace

A generally available WebSphere MQ product that provides proven WebSphere MQ reliability and security for mobile and wireless devices. WebSphere MQ Everyplace applications connect to the broker using WebSphere MQ Mobile Transport.

WebSphere MQ Mobile Transport

A transport protocol supported by WebSphere Message Broker that enables WebSphere MQ Everyplace application clients to connect to brokers.

WebSphere MQ Multicast Transport

A transport protocol supported by WebSphere Message Broker that enables dedicated JMS application clients to connect to brokers. This protocol is optimized for high volume, one-to-many publish/subscribe topologies.

WebSphere MQ Real-time Transport

A transport protocol supported by WebSphere Message Broker that enables dedicated JMS application clients to connect to brokers.

WebSphere MQ Telemetry Transport

A transport protocol supported by WebSphere Message Broker that enables SCADA devices to connect to brokers. This protocol is a lightweight publish/subscribe protocol that flows over TCP/IP that uses a subset of UTF-8.

WebSphere MQ Web Services Transport

A transport protocol supported by WebSphere Message Broker that enables HTTP compliant application clients to connect to brokers.

wild card

A character that can be specified in subscriptions to match a range of topics. See also multilevel wild card and single-level wild card.

workbench

See Message Brokers Toolkit

World Wide Web Consortium (W3C)

An international industry consortium set up to develop common protocols to promote evolution and interoperability of the World Wide Web.

WSDL

See Web Services Description Language.

W3C See World Wide Web Consortium.

X

XML See extensible markup language.

XML domain

The message domain that includes all messages that conform to the W3C XML standard. The XMLNS domain is an extension of the XML domain and contains messages that conform to the same standard and that can exploit the namespaces feature of the XML specification. Messages in this domain are processed by the XML parser. See also BLOB domain, IDoc domain, JMS domain, and MRM domain.

XML parser

A program that interprets a bit stream or tree that represents a message that belongs to the XML domain and JMS domains, and generates the corresponding tree from the bit stream on input, or bit stream from the tree on output. The bit stream is a representation of an XML file.

XML Path Language

An XSL sublanguage designed to uniquely identify or address parts of a source XML document, for use with XSLT. XPath also provides basic facilities for manipulation of strings, numbers, and Boolean values.

XML schema

An international standard that defines a language for describing the structure of XML documents. An XML schema formally describes and constrains the content of XML documents by indicating which elements are allowed and in which combinations. (An XML Schema is an alternative to a document type definition (DTD), and can be used to extend functionality in the areas of data typing, inheritance, and presentation.) The XML Schema language is ideally suited to describing the messages which flow between business applications, and it is widely used in the business community for this purpose.

XML Wire Format

The physical representation of a message in the MRM domain that can be parsed as XML.

XPath See XML Path Language.

XSL See extensible stylesheet language.

Part 3. Appendixes

Appendix. Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this information in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this information. The furnishing of this information does not give you any license to these patents. You can send license inquiries, in writing, to:

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

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

*IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032,
Japan*

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the information. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this information at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licenses of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

*IBM United Kingdom Laboratories,
Mail Point 151,
Hursley Park,
Winchester,
Hampshire,
England
SO21 2JN*

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Programming License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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 and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information includes examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not

been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

(C) *(your company name) (year)*. Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. *_enter the year or years_*. All rights reserved.

Trademarks

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

AIX	CICS	Cloudscape
DB2	DB2 Connect	DB2 Universal Database
developerWorks	Domino	
Everyplace	FFST	First Failure Support Technology
IBM	IBMLink	IMS
IMS/ESA	iSeries	Language Environment
Lotus	MQSeries	MVS
NetView	OS/400	OS/390
pSeries	RACF	Rational
Redbooks	RETAIN	RS/6000
SupportPac	Tivoli	VisualAge
WebSphere	xSeries	z/OS
zSeries		

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel and Pentium are trademarks or registered trademark of Intel Corporation or its subsidiaries in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Index

A

- access control lists
 - User Name Server 30
- accessibility 33
 - keyboard shortcuts 68
- administration 24
 - Broker Administration perspective 37
- API, Configuration Manager 26
- archive
 - Broker Archive editor 47

B

- bar files
 - editor 47
- Broker Administration perspective 37
- Broker Application Development perspective 39
- Broker Archive editor 47
- broker archive files
 - editor 47
- broker domains 25
 - diagram 24
- Broker Topology editor 48
- brokers 28
 - broker domain 25
 - configuration details 25
 - execution groups 30
 - managing 48
 - system management 29
- business integration, WebSphere MQ 3

C

- changes in this release 5
 - Version 5.0 12
 - Version 5.0 fix packs 15
 - Version 6.0 5
 - Version 6.0 fix packs 10
- CICS 3
- client environment 19
- code pages
 - editor preferences 47
- components
 - broker domains 25
 - brokers 28
 - Configuration Manager 25
 - execution groups 30
 - User Name Server 30
- Configuration Manager 25
 - API 26
- Configuration Manager Proxy 26
- context-sensitive help 19
- conventions
 - resource names 46

D

- Debug perspective 42
- development
 - Broker Application Development perspective 39
 - Plug-in Development perspective 43
- development repository 24
- display
 - high-contrast 33

E

- Eclipse 19
- editor preferences, localized settings 47
- editors 21
 - Broker Archive 47
 - Broker Topology 48
 - ESQL 49
 - Event Log 52
 - localized settings 47
 - Message Category 53
 - Message Definition 54
 - Message Flow 58
 - Message Mapping 64
 - Message Node 65
 - Message set 65
 - preferences 47
 - Subscriptions Query 66
 - Topics Hierarchy 67
- environments
 - client 19
 - runtime 24
- ESQL
 - files
 - sharing 47
- ESQL editor 49
- Event Log editor 52
- execution groups 30

F

- files 22
 - message flows 44
 - message sets 44
 - plug-in development 45
 - sharing 47
- folders 22

H

- help, context-sensitive 19
- high contrast
 - display 33

I

- IMS/ESA 3
- infopops 19
- introduction 3

- introduction (*continued*)
 - scenarios 31
 - technical 16
 - WebSphere Message Brokers 3

K

- keyboard shortcuts 68

L

- linking objects by name 23
- Lotus Domino 3

M

- mappings
 - Message Mapping editor 64
- Message Brokers Toolkit 19
- Message Category editor 53
- message category files
 - editing 53
- Message Definition editor 54
 - Outline view 56
 - Overview editor 57
 - Properties editor 57
- message definition files
 - Message Definition editor 54
- Message Flow editor 58
- message flows
 - Broker Application Development perspective 39
 - Debug perspective 42
 - execution groups 30
 - Mapping editor 64
 - projects and files 44
- Message Mapping editor 64
- Message Node editor 65
- Message set editor 65
- message set files
 - creating 65
 - editing 65
- message sets
 - Broker Application Development perspective 39
 - projects and files 44
- Microsoft Exchange 3

N

- name, linking by 23
- naming resources 46
- new function and capabilities 5
 - Version 5.0 12
 - Version 5.0 fix packs 15
 - Version 6.0 5
 - Version 6.0 fix packs 10

O

objects, linking by name 23
overview 16
 scenarios 31

P

perspectives 20
 Broker Administration 37
 Broker Application Development 39
 Debug 42
 overview 37
 Plug-in Development 43
 switching 20
plug-in development
 projects and files 45
Plug-in Development perspective 43
preferences
 editors 47
projects 22
 message flows
 resource files 44
 message sets
 resource files 44
 plug-in development 45
 reference property 24
publish/subscribe
 User Name Server 30

Q

queues 28

R

reference, project 24
repository, development 24
resources 22
 brokers 28
 Configuration Manager 25
 message flows 44
 message sets 44
 naming rules 46
 plug-in development 45
 types 43
 User Name Server 30
rules
 resource names 46
runtime
 components 24
 environment 24

S

SAP/R3 3
scenarios 31
 WebSphere Message Brokers 31
shortcuts, keyboard 68
Subscriptions Query editor 66
Supply Chain Management 3
system management interfaces 29

T

text-only Quick Tour 16
Toolkit 19
Topics Hierarchy editor 67
topology
 broker 48
trademarks 93

U

User Name Server 30
user-defined nodes, editing 65
UTF-8 47

W

WebSphere MQ 3
 AMI 3
 business integration 3
 MQI 3
 WebSphere MQ messaging 3
workbench 19
 Broker Administration perspective 37
 Broker Application Development
 perspective 39
 Debug perspective 42
 Plug-in Development perspective 43



Printed in USA